

# Exploring Sparse, Unstructured Video Collections of Places

*James Henri Tompkin*

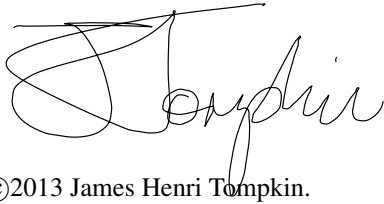
A dissertation submitted in partial fulfilment  
of the requirements for the degree of  
**Doctor of Engineering**  
of the  
**University College London.**

Department of Computer Science  
University College London

11<sup>th</sup> February 2013

'I, James Henri Tompkin, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the thesis.'

Signed:

A handwritten signature in black ink, appearing to read 'J. Tompkin', written in a cursive style.

©2013 James Henri Tompkin.



# Abstract

The abundance of mobile devices and digital cameras with video capture makes it easy to obtain large collections of video clips that contain the same location, environment, or event. However, such an unstructured collection is difficult to comprehend and explore. We propose a system that analyses collections of unstructured but related video data to create a *Videoscape*: a data structure that enables interactive exploration of video collections by visually navigating — spatially and/or temporally — between different clips. We automatically identify transition opportunities, or *portals*. From these portals, we construct the Videoscape, a graph whose edges are video clips and whose nodes are portals between clips. Now structured, the videos can be interactively explored by walking the graph or by geographic map. Given this system, we gauge preference for different video transition styles in a user study, and generate heuristics that automatically choose an appropriate transition style. We evaluate our system using three further user studies, which allows us to conclude that Videoscapes provides significant benefits over related methods. Our system leads to previously unseen ways of interactive spatio-temporal exploration of casually captured videos, and we demonstrate this on several video collections.

# Acknowledgements

To Jan Kautz, my doctoral supervisor and academic mentor, whose unflinching support and easy-going attitude made no problem seem large even through my extended study. To his patience in waiting for my thesis and putting up with my side projects and endeavours.

To Gabriel Brostow and Simon Prince, my second supervisors, who provided valuable points of reflection and career support.

To Oliver Grau, my industrial supervisor at the BBC, who provided time, resources, and a critical eye on my plans and work.

To Christian Theobalt, Kwang In Kim, and Miguel Granados, who welcomed me into collaborations with the Max-Planck-Institut für Informatik, and who were always ready with support when asked.

To Wojciech Matusik, who supervised me as an intern at Disney Research Boston and welcomed me into CSAIL at MIT.

To my co-authors: Abel Maciel, Samuel Muff, Feng Xu, Beste F. Yuksel, Simon Heinzle, Henrik Lieng, Philippe Levieux, Fabrizio Pece, Kartic Subr, Carsten Stoll, Gaurav Bharaj, Yebin Liu, Jim McCann, Jennifer G. Sheridan, Stanislav Jakushevskij, Marc Alexa, Michael Donnerer, George Roussos, Martin Theobald, Qionghai Dai, and Hans-Peter Seidel.

To my colleagues and friends at UCL: Sadia Rahman Zain, Min H. Kim, Jeren Chen, Andrew Cox, Laura Panagiotaki, Harsha Sri-Narayana, Davide Vercelli, Fotios Tzellos, Alastair Moore, Christopher Leung, and all those in the Kautzonauts and VECG groups.

To my family: the Tompkins, the Puckrins, the Howards, and the Jolys.

To HSLB.

*For my parents, Kendrick and Dominique.*

# Contents

<b>1</b>	<b>Introduction</b>	<b>13</b>
1.1	Approach . . . . .	14
1.2	Contributions . . . . .	15
1.3	Thesis Outline . . . . .	17
1.4	Miscellanea . . . . .	17
<b>2</b>	<b>Background</b>	<b>18</b>
2.1	Introduction . . . . .	18
2.2	The Breadth of Image-based Rendering . . . . .	19
2.3	A Brief History of Image-based Environments . . . . .	20
2.4	Video Collections . . . . .	23
2.5	Commercial Examples . . . . .	24
2.5.1	Image-based Rendering and Image-based Environments . . . . .	24
2.5.2	Video Collections . . . . .	26
2.6	Conclusion . . . . .	27
<b>3</b>	<b>Literature Review</b>	<b>29</b>
3.1	Introduction . . . . .	29
3.2	Related Work . . . . .	29
3.2.1	Content-based Retrieval . . . . .	29
3.2.2	Structuring Media Collections . . . . .	30
3.2.3	Rendering and Exploring Media Collections . . . . .	32
3.2.4	Summary . . . . .	33
3.3	Key Papers in Detail . . . . .	34
3.3.1	RealityFlythrough: A System for Ubiquitous Video . . . . .	34
3.3.2	Creating Map-based Storyboards for Browsing Tour Videos . . . . .	37
3.3.3	Unstructured Video-based Rendering . . . . .	41
3.4	Discussion . . . . .	43
3.5	Conclusion . . . . .	45

<b>4</b>	<b>Approach Overview</b>	<b>46</b>
4.1	Introduction . . . . .	46
4.2	Definitions . . . . .	46
4.3	System Overview . . . . .	47
4.4	Meeting the Recommendations . . . . .	49
4.5	Scope of the Solution . . . . .	51
4.6	Experimental Databases . . . . .	51
4.6.1	London . . . . .	51
4.6.2	South Bank . . . . .	52
4.6.3	Campus Bike . . . . .	52
4.7	Conclusion . . . . .	52
<b>5</b>	<b>Identifying Portals</b>	<b>53</b>
5.1	Introduction . . . . .	53
5.2	Filtering . . . . .	54
5.2.1	Discussion . . . . .	55
5.3	Holistic Matching and Feature Matching . . . . .	57
5.4	Context Refinement . . . . .	58
5.4.1	Discussion . . . . .	59
5.5	Portal Selection . . . . .	61
5.5.1	Support Sets . . . . .	62
5.6	Synchronization . . . . .	63
5.6.1	Sensor Data . . . . .	63
5.6.2	Time . . . . .	64
5.7	Pipeline . . . . .	64
5.8	Experiment: Context Refinement and Portal Identification . . . . .	64
5.8.1	Context Refinement . . . . .	64
5.8.2	Portal Identification . . . . .	66
5.9	Conclusion . . . . .	67
<b>6</b>	<b>Video Transitions</b>	<b>68</b>
6.1	Introduction . . . . .	68
6.2	Experiment Design . . . . .	69
6.2.1	Transition Choice . . . . .	70
6.2.2	Clip Choice . . . . .	72
6.3	Experiment: Towards Automatic Transition Type Choice . . . . .	77
6.3.1	Hypothesis . . . . .	77
6.3.2	Method . . . . .	77
6.3.3	Results . . . . .	78

6.3.4	Discussion	79
6.3.5	Individual Transition Types	83
6.3.6	Individual Sets	87
6.3.7	Outcomes	87
6.4	Summary	94
<b>7</b>	<b>Videoscape Exploration</b>	<b>96</b>
7.1	Introduction	96
7.2	Videoscapes Interfaces	97
7.2.1	Interactive Exploration Mode	97
7.2.2	Overview Modes	100
7.2.3	Fast Geographical Browsing	103
7.2.4	Workflow Switching Animations	103
7.3	Videoscape Labelling, Propagation, and Search	106
7.4	Experiment: Interface Evaluation	108
7.4.1	Spatial Awareness Experiment	108
7.4.2	Video Tour Experiment	111
7.4.3	Video Browsing Experiment	114
7.5	Summary	122
<b>8</b>	<b>Discussion</b>	<b>124</b>
8.1	Portal Finding	124
8.2	Transitions	125
8.3	Interfaces	127
8.4	Databases	128
<b>9</b>	<b>Conclusion</b>	<b>131</b>
	<b>Appendices</b>	<b>134</b>
<b>A</b>	<b>Transition Types in Detail</b>	<b>134</b>
A.1	Cut	134
A.2	Dissolve	137
A.3	Warp	138
A.4	Plane	143
A.5	Ambient Point Clouds	146
A.6	Full 3D	149
A.7	Transition Issues in Common	152
A.7.1	Geometry Recovery and Processing	152
A.7.2	Video Registration	156
A.7.3	Camera Interpolation	159

A.7.4 Empty Areas and Inpainting . . . . .	160
A.7.5 Transition Timing Differences . . . . .	162
A.8 Video Stabilization . . . . .	163
A.9 Transition Feature/Artefact Table . . . . .	164
<b>B Individual Set Perceptual Score Analysis</b>	<b>166</b>
<b>C Transition Experiment Material</b>	<b>179</b>
<b>D Spatial Awareness Experiment Material</b>	<b>184</b>
<b>E Video Tour Experiment Material</b>	<b>187</b>
<b>F Video Browsing Experiment Material</b>	<b>190</b>
<b>Bibliography</b>	<b>190</b>

# List of Figures

2.1	Image-based rendering hierarchy. . . . .	21
2.2	Aspen Movie-map interface. . . . .	22
2.3	Google Maps Street View and Microsoft Street Side. . . . .	25
2.4	Quiksee. . . . .	26
2.5	Switchcam and Vyclone. . . . .	27
3.1	RealityFlythrough session. . . . .	34
3.2	RealityFlythrough transitions. . . . .	35
3.3	RealityFlythrough transition experiments 3 and 4. . . . .	36
3.4	Pongnumkul et al. system viewing interface. . . . .	38
3.5	Pongnumkul et al. advanced viewing modes. . . . .	39
3.6	Ballan et al. example transition. . . . .	41
3.7	Ballan et al. interface modes. . . . .	42
3.8	Ballan et al. experiment. . . . .	42
4.1	Videoscapes overview. . . . .	47
4.2	Videoscapes geometry reconstruction overview. . . . .	48
5.1	An example of a mistakenly found portal. . . . .	58
5.2	An example of a hypothetical local connectivity-based confidence assignment. . . . .	60
5.3	An example graphical embedding of frames and their connections. . . . .	61
5.4	Example portal frame pairs. . . . .	62
6.1	Slight view change transition example. . . . .	74
6.2	Considerable view change transition example. . . . .	74
6.3	All scenes with slight and considerable view changes. . . . .	76
6.4	All scene transition type experiment results. . . . .	78
6.5	Horizontal pan effects diagrams for full 3D dynamic transitions. . . . .	90
6.6	Horizontal pan effects diagrams for full 3D static transitions. . . . .	92
6.7	Zoom effects on empty areas diagrams for transitions. . . . .	93
7.1	Interactive exploration portal choice example. . . . .	99

7.2	Different options for representing videos that are currently used in map-based image and video exploration softwares. . . . .	101
7.3	Overview mode with eye icons. . . . .	102
7.4	Early prototype overview mode . . . . .	104
7.5	Path planning and video browsing workflows. . . . .	105
7.6	Label propagation example. . . . .	107
7.7	Spatial awareness experiment steps. . . . .	109
7.8	Spatial awareness experiment results. . . . .	110
7.9	Video browsing comparison interfaces. . . . .	115
7.10	Completion time results for video browsing experiment. . . . .	117
7.11	Error results for video browsing experiment. . . . .	118
A.1	Jump cut transition example. . . . .	134
A.2	180° and 30° cutting rules. . . . .	135
A.3	180° rule break. . . . .	136
A.4	Dissolve transition example. . . . .	137
A.5	Warp without flow or inpainting. . . . .	140
A.6	Warp with flow correction. . . . .	142
A.7	Warp artefacts. . . . .	143
A.8	Plane transition results. . . . .	145
A.9	Plane artefacts. . . . .	145
A.10	Ambient Point Clouds transition. . . . .	147
A.11	Ambient Point Cloud artefacts. . . . .	149
A.12	Full 3D dynamic transition. . . . .	150
A.13	Full 3D static transition. . . . .	150
A.14	Full 3D static transition. . . . .	151
A.15	Recovered point cloud geometry cleaning. . . . .	155
A.16	Point cloud surface reconstruction. . . . .	156
A.17	Camera interpolation diagram. . . . .	161
B.1	All scenes with slight and considerable view changes. . . . .	168
B.2	Set 1 transition type experiment results. . . . .	169
B.3	Set 2 transition type experiment results. . . . .	170
B.4	Set 3 transition type experiment results. . . . .	171
B.5	Set 4 transition type experiment results. . . . .	172
B.6	Set 5 transition type experiment results. . . . .	173
B.7	Set 6 transition type experiment results. . . . .	174
B.8	Set 7 transition type experiment results. . . . .	175
B.9	Set 8 transition type experiment results. . . . .	176



B.10	Set 9 transition type experiment results. . . . .	177
B.11	Set 10 transition type experiment results. . . . .	178
C.1	Experiment description webpage for transition ranking experiment. . . . .	180
C.2	Video ranking webpage for transition ranking experiment. . . . .	181
C.3	Example transition experiment result sheet. . . . .	183
D.1	Spatial awareness experiment website. . . . .	184
D.2	Spatial awareness experiment website. . . . .	185
D.3	Example spatial awareness experiment result sheet. . . . .	186
E.1	Video tour summarization experiment website. . . . .	188
E.2	Example video tour summarization experiment result sheet. . . . .	189
F.1	Video browsing experiment questionnaire website. . . . .	190
F.2	Example video browsing experiment result sheet. . . . .	191

# List of Tables

5.1	Number of filtered frames for each database. . . . .	57
5.2	Performance of spectral and local connectivity analyses. . . . .	66
5.3	Performance of portal identification. . . . .	67
6.1	Transition feature/artefact table. . . . .	73
6.2	Experiment scenes with features. . . . .	75
6.3	All sets t-test significance. . . . .	78
6.4	All sets perceptual scaling values. . . . .	79
6.5	Slight view change sets t-test significance. . . . .	79
6.6	Considerable view change sets t-test significance. . . . .	80
6.7	Transition positive and negative comments. . . . .	80
6.8	Transition feature/artefact comments table. . . . .	81
6.9	Horizontal pan effects on empty areas in full 3D dynamic transitions. . . . .	89
6.10	Horizontal pan effects on empty areas in full 3D static transitions. . . . .	91
6.11	Zoom effects on empty areas in transitions. . . . .	91
7.1	Spatial awareness experiment questionnaire results, 1. . . . .	111
7.2	Spatial awareness experiment questionnaire results, 2. . . . .	111
7.3	Spatial awareness experiment participant comments. . . . .	112
7.4	Video summarization experiment questionnaire results. . . . .	113
7.5	Video tour summarization experiment participant comments. . . . .	114
7.6	Video browsing experiment questionnaire results, 1. . . . .	116
7.7	System use desire experiment results. . . . .	117
7.8	Video browsing experiment questionnaire results, 2. . . . .	119
7.9	Video browsing experiment participant comments about iMovie. . . . .	120
7.10	Video browsing experiment participant comments about Pongnumkul et al. . . . .	121
7.11	Video browsing experiment participant comments about Videoscapes. . . . .	122
A.1	Transition feature/artefact table. . . . .	165
B.1	Experiment scenes with features. . . . .	167
C.1	Corresponding score attributed to ranking for Figure C.3. . . . .	183

## Chapter 1

# Introduction

With the ubiquity of video capture devices, it is very easy to form video collections. Online, there are staggeringly large video collections. However, the interfaces to these video collections are often simply lists of text-ranked videos which do not exploit the visual content relationships between videos, nor other implicit relationships such as spatial or geographical relationships. Finding content relationships between arbitrary videos is difficult, and the field of multimedia retrieval tries to address these problems. However, we want to provide better interfaces to exploit content relationships for the specific subset of videos which capture places. This important subset of videos shows the dynamics and liveliness of a place and events or performances within those places. Videos of places may be found in online video collections, or could be captured specifically to demonstrate a particular place and the activities that happen within it.

Imagine a theme park. The owners specifically capture professional video of the park to show off the lively environment and the rides and attractions. Similarly, visitors to the park also capture their own videos of their experiences in the theme park. However, currently there is no automatic way to find similar content between these videos and allow people to explore the interesting connections that are within the collection. For instance, potential visitors to the parks could explore the collection as an advertising tool or to plan their trip, or existing visitors could relive their experiences and see their own videos integrated into a wider collection.

## Hypothesis

If we can automatically find visual content relationships between sparse, casually captured videos of places in a collection, then we can provide qualitative and quantitative improvements to video collection exploration through novel interfaces.

Our goal is to build a system to explore connections within video collections of places, to show with examples that there are compelling use cases for novel interfaces which allow connection exploration, and to experimentally evaluate this system against existing interfaces for video collection exploration to provide evidence which supports the thesis hypothesis.

## 1.1 Approach

In recent years, the research community has started to harvest the immense amount of data from community photo collections, and has developed tools to estimate the spatial relationships between photographs, or to reconstruct 3D geometry of certain landmarks if a sufficiently dense set of photos is available [SSS06, GSC<sup>+</sup>07, ASS<sup>+</sup>09, FGG<sup>+</sup>10]. With these tools, we can interactively explore locations by viewing the reconstructed 3D models or spatially transitioning between photographs. Navigation tools like Google Street View or Bing Maps also use this exploration paradigm and reconstruct entire street networks through alignment of purposefully captured imagery via additionally recorded localization and depth sensor data.

These photo exploration tools are ideal for viewing and navigating static landmarks but cannot convey the dynamics, liveliness, and spatio-temporal relationships of a place or the events within that place. Additionally, there are no comparable browsing experiences for casually captured videos and how to generate these experiences is still an open challenge. It may be tempting to think that videos are simply series of images, so straightforward extensions of image-based approaches should serve the purpose and enable video collection exploration. However, in reality, the nature of casually captured video is different from photos and prevents such a simple extension. Casually captured video collections are usually sparse and largely unstructured, unlike the dense photo collections used in the approaches mentioned above. This precludes a dense reconstruction or registration of all frames.

Furthermore, the exploration interface should reflect the dynamic and temporal nature of video. This major data difference causes problems for existing image-based approaches. Current interfaces to video collections expect videos to be isolated and cannot handle the expression of connections within the collection. Existing spatial or geographical video browsing techniques do not extend to sparse, casually captured video collections, and typically either handle single videos or require complicated capture setups.

In this thesis, we propose a system to explore unstructured video collections in an immersive and visually compelling way. Given a sparse video collection of a certain (possibly large) area, e.g., the inner city of London, the user can tour through the video collection by following videos and transitioning between them at corresponding views. While our system cannot provide directions from location A to B, as sparse video collections may not contain sufficient input, it does provide the spatial arrangement of landmarks contained within a video collection (distinct from the geolocations of video captures). Unlike tours through images, our system conveys a sense of place, dynamics and liveliness while still maintaining seamless browsing with video transitions. The challenge is to build a set of techniques to analyse such video collections, and to provide a set of interfaces to exploit the derived structure.

To this end, we compute a *Videoscape* graph structure from a collection of videos. The edges of the Videoscape are video segments and the nodes mark possible transition points, or *portals*, between videos. We automatically identify portals from an appropriate subset of the video frames as there is often great redundancy in videos, and process the portals (and the corresponding video frames) to enable smooth transitions between videos. The Videoscape can be explored interactively by playing video clips

and transitioning to other clips when a portal arises. When temporal context is relevant, our system provides temporal awareness of an event by offering correctly ordered transitions between temporally aligned videos. This yields a meaningful spatio-temporal viewing experience of large, unstructured video collections. With GPS and orientation data, a map-based mode lets the user choose start and end views of content within the collection, from which the system automatically finds a path of videos and transitions to join them. Furthermore, images can be given to the system, from which a path through the Videoscape graph between the closest matching portals is formed. To enhance the experience when transitioning through a portal, we develop different video transition modes, with appropriate transitions selected based on the preference of participants in a user study. Finally, we evaluate the Videoscape system with three further comparative user studies which address spatial awareness, video tour summarization, and video browsing.

## 1.2 Contributions

The contributions of this thesis are:

- **Videoscape graph:** A graph capturing the semantic links within a video collection. Edges are video clips and nodes are *portals*, which represent transition points between videos.
- **Videoscape construction:** An effective filtering strategy for portal candidates, and the adaption of holistic and feature-based matching strategies to video frame matching. The system also includes a graph-based spectral refinement strategy which, when placed into our coarse-to-fine graph construction strategy, enables us to automatically find portals with 98% precision and 53% recall. Parts of this work were completed with colleague Kwang In Kim; see Chapter 5 for full details.
- **Transition construction:** A practical demonstration of how to generate various different image- and geometry-based transition types, including combining geometry reconstruction, tracking and stabilization to generate dynamic transitions with 3D geometry.
- **Transition preference:** A user study analysing preferred transition types across scene and view-point changes, and heuristics for their appropriate use. A detailed analysis of transition artefacts, leading to further heuristics which rank the relative importance of different artefacts.
- **Videoscape exploration:** An explorer application that enables intuitive and seamless spatio-temporal exploration of the Videoscape, based on several novel exploration paradigms.
- **Videoscape evaluation:** Three user studies providing data comparing Videoscapes to existing systems. The first experiment quantitatively and qualitatively assesses transitions for their ability to improve spatial awareness when switching between two videos in map-based interfaces. The second experiment qualitatively assesses Videoscapes tours as summarization tools. The third experiment both quantitatively and qualitatively assesses Videoscapes as a tool for finding content within video collections. We also measure participant preference for different interface elements and for the system as a whole.

Much of the content for this thesis is derived from the following paper, though here this content is significantly expanded:

- James Tompkin, Kwang In Kim, Jan Kautz, and Christian Theobalt. Videoscapes: Exploring Sparse, Unstructured Video Collections. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 31(4), 2012

During the doctoral study for this thesis, the candidate student also contributed to the following peer-reviewed publications and juried exhibitions:

- Philippe Levieux, James Tompkin, and Jan Kautz. Interactive Viewpoint Video Textures. In *Visual Media Production (CVMP), 2012 Conference on*, 2012.
- Kwang In Kim, James Tompkin, Martin Theobald, Jan Kautz, and Christian Theobalt. Match Graph Construction for Large Image Databases. In *European Conference on Computer Vision (ECCV)*, 2012.
- Miguel Granados, Kwang In Kim, James Tompkin, Jan Kautz, and Christian Theobalt. Background Inpainting for Videos with Dynamic Objects and a Free-moving Camera. In *European Conference on Computer Vision (ECCV)*, 2012.
- James Tompkin, Samuel Muff, Stanislav Jakushevskij, Jim Mccann, Jan Kautz, Marc Alexa, and Wojciech Matusik. Interactive Light Field Painting. In *SIGGRAPH 2012 Emerging Technologies*, 2012.
- Miguel Granados, James Tompkin, Kwang In Kim, Oliver Grau, Jan Kautz, and Christian Theobalt. How Not to Be Seen - Object Removal from Videos of Crowded Scenes. *Computer Graphics Forum*, 31(2pt1):219–228, May 2012.
- Henrik Lieng, James Tompkin, and Jan Kautz. Interactive Multi-perspective Imagery from Photos and Videos. *Computer Graphics Forum*, 31(2pt1):285–293, May 2012.
- James Tompkin, Fabrizio Pece, Kartic Subr, and Jan Kautz. Towards Moment Imagery: Automatic Cinemagraphs. *Visual Media Production (CVMP), 2011 Conference on*, 2011.
- Feng Xu, Yebin Liu, Carsten Stoll, James Tompkin, Gaurav Bharaj, Qionghai Dai, Hans-Peter Seidel, Jan Kautz, and Christian Theobalt. Video-based Characters - Creating New Human Performances from a Multi-view Video Database. In *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 2011.
- Beste F. Yuksel, Michael Donnerer, James Tompkin, and Anthony Steed. Novel P300 BCI Interfaces to Directly Select Physical and Virtual Objects. In *5th International Brain-Computer Interface Conference*, pages 5–8, 2011.
- Beste F. Yuksel, Michael Donnerer, James Tompkin, and Anthony Steed. A Novel Brain-computer Interface using a Multi-touch Surface. *Proceedings of the 28th International Conference on Human Factors in Computing Systems - CHI '10*, page 855, 2010.

- Jennifer G. Sheridan, James Tompkin, Abel Maciel, and George Roussos. DIY Design Process for Interactive Surfaces. *Proceedings of 23rd Conference on Human Computer Interaction*, 2009.

## 1.3 Thesis Outline

Chapter 2 presents a background review of the literature of image-based rendering, image-based environments, and video collections. It also reviews commercial examples which exploit these approaches to provide relevant interfaces to videos and video collections. In Chapter 3, we present work which directly relates to our problem of structuring and exploring video collections. It identifies three key papers which most closely relate to our problem, and through analysis finds recommendations for a video collection system. Chapter 4 describes our implementation of a system to meet these recommendations, and defines the scope of the implementation. With the broad introductions over, Chapter 5 begins the detailed description of the system with the off-line component. We explain how to find portals and their support sets from very large collections of video frames. Chapter 6 explains how to generate transitions from the Videoscape graph. It describes in detail an experiment to measure participant preference for different transition types, and analyses the artefacts within these transitions. With a Videoscape graph and a set of transitions, Chapter 7 details the various exploration interfaces we have designed. It explains three user studies to test our system and presents their findings. Chapter 8 reflects upon the system in discussion, covering limitations and potential future work. Chapter 9 concludes this thesis.

Appendix A contains a detailed explanation of the background, generation, and artefacts of each transition type. Appendix B provides additional scene analysis from the preference experiment in Chapter 6. Appendices C, D, E, and F list experimental interfaces and data.

## 1.4 Miscellanea

Throughout this document, we reference a part of our London database that includes videos of the Palace of Westminster and the clock tower commonly known as Big Ben. Where we reference Big Ben, we appreciate that we do so incorrectly: that Big Ben is the heaviest bell of five in the tower, and that the tower is officially named the Elizabeth Tower (and prior to 12<sup>th</sup> September 2012, named simply as the Clock Tower of the Palace of Westminster).

Certain image results may have been auto-levelled for better contrast, especially in print. Where it is important for the interpretation of a result this will be mentioned explicitly in the figure caption, but otherwise it will not be mentioned.

## Chapter 2

# Background

## 2.1 Introduction

Image- and video-based environments are media-driven representations of places and events which allow interactive explorations through space and time. Image-based environments create photo-realistic display imagery but cannot represent accurate world dynamics; video-based environments further attempt to display these dynamics. Such techniques are most commonly used to represent real-world places and events as captured by cameras, rather than virtual places and events.

Computer graphics techniques are more suitable for virtually representing places and events where flexibility or world interactivity are important. Video games make use of world interactivity to allow fluid and immersive player interactivity, and both video games and movies use the visual freedom that computer graphics provides to create novel, otherworldly aesthetics. The type of environment to be represented and the interface/application goals are key to deciding which approach to use: real-time rendering is not necessarily realistic, visual effects rendering cannot provide interactivity, and both approaches often require man-years of content creation. Alternatively, image- and video-based approaches are most useful for applications where accurate or realistic representation of the real world is important, where limited interactivity is acceptable, and where man power is limited. The difficulty with image- and video-based environments comes with turning the captured images into an interactive experience.

The development of video-based environment techniques has been, in part, dependent upon exponential improvements in computer storage and compression. Over the past 20 years, this has made feasible the collation of large digital video collections with thousands of hours of video on a single disk. More recently, concurrent exponential growth in the speed of telecommunications has created online repositories to which years of video are uploaded daily<sup>1</sup>. Methods to exploit video collections to create video-based environments would have access to more video than ever previously. However, new techniques to cope with these ever-increasing stores need to be developed and, given these, new interfaces need to automatically exploit similar content and provide semantic connections between videos within video-based environments.

To provide context to this thesis, and to ground the literature review in Chapter 3, we present a back-

---

<sup>1</sup>As of 15th June 2012, YouTube receives 8 years of content uploads per day [You12], though this number may include duplicates and forbidden content.



ground to image-based rendering and video-based collections and environments. Image-based rendering is a broad topic spanning both computer vision and computer graphics. To place video-based collections and environments within this spectrum, Section 2.2 demonstrates the range of problems that image-based rendering has attempted to solve over the past 15 years. We then focus on the specific history of image- and video-based environments to learn about previous approaches to tackle the problem, their scope, and where they have succeeded (Section 2.3). A brief history of video collections then follows in Section 2.4. Finally, a discussion of recent commercial applications of video-based environments presents the current state of consumer interfaces to video collections (Section 2.5).

## 2.2 The Breadth of Image-based Rendering

Image- and video-based rendering as a field is broad. Its youth also adds to its breadth: many conventions are yet to be formed. Not only does the term IBR apply to the creation of the final result, but it also applies to many of the processes along the pipeline from capture to result. This pipeline often includes many other image-based techniques that could not be called rendering, but are an intrinsic part of IBR linked by their use of camera captured images. IBR research frequently includes work on infrastructure, calibration and capture techniques. IBR researchers draw from many different areas of computer graphics, computer vision, and photography.

Image capture is often the first problem faced by the IBR researcher. IBR techniques often use cameras in various different arrangements: singularly, mounted on mobile platforms [AFYC03], in small clusters mimicking an omnidirectional camera [UCK<sup>+</sup>04], along a short arc focused on an object [ZKU<sup>+</sup>04], or even surrounding an object as completely as possible, either sparsely [KSV98] or densely [CEJ<sup>+</sup>06]. Cameras are often used with ultra wide-angle (or ‘fish-eye’) lenses [XT97], or take pictures of mirrored curved surfaces, often spheres, to create a similar effect [AFYC03]. High dynamic range camera techniques may also be used [UCK<sup>+</sup>04]. Almost all IBR techniques require accurate calibration of camera geometry and often calibration of colour reproduction — this is especially important when using multiple synchronous cameras.

Many image-based techniques strive to reconstruct the geometry of a scene, be it a static object on a pedestal, a dynamic human actor in a studio, or a real world environment with both static and dynamic objects. Given the geometry, the captured images can be applied as textures to obtain a photo-realistic representation that is difficult and laborious to obtain with from-the-ground-up computer graphics. Segmentation is usually necessary to isolate the object of interest. In a studio, this is frequently performed with chroma-keying [Gra04] but difference keying is also used in less controlled environments [IBL00, HGK<sup>+</sup>11]. Volume reconstruction then usually starts with visual hull computation [Lau94, SSH02, Gra11]. Optimization strategies exist to improve reconstruction [MSH06, GH10] but other approaches exist, such as using model-based templates [Sta03, CTMS03]. Structured light may be used to increase the accuracy of the reconstruction [WWC<sup>+</sup>05], as may other cues such as shading and shadow [ALS07]. When attempting to capture dynamic scenes, often operations are performed frame to frame, leaving temporal inconsistency artefacts. Space/time coherence algorithms fuse these time instances and form a consistent spatio-temporal representation [BZS<sup>+</sup>07].

Combining the texture information stored in the captured image with the reconstructed geometry allows us to view a realistic depiction of the object. However, correct texture mapping is not a trivial task. View-dependent texture mapping [DYB98] allows fast texturing for arbitrary viewpoints from a sparse set of cameras. Isolating specular reflections is necessary to reproduce the diffuse component of a surface [TI05]. This allows the object to be relit more accurately for a new scene [Gra06]. Full BRDF estimation is desirable, and some techniques attempt this via iterative image comparison [BG01]. Relighting is possible once surface properties are known [ATS07, TAL<sup>+</sup>07]. More ambitious capture rigs, called light stages, allow for extremely accurate relighting [CEJ<sup>+</sup>06, GFT11] though convincing results are possible with simple equipment and modest constraints [PTMD07].

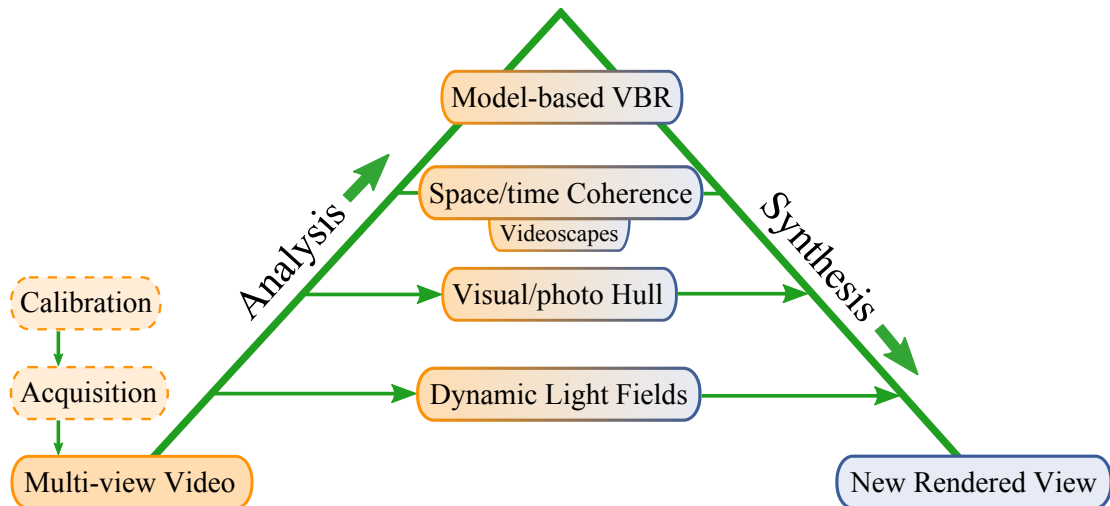
For certain IBR applications, full scene geometry is not required, but some depth information is necessary. Depth from stereo is a commonly used technique in multi-camera systems [HS04, ZKU<sup>+</sup>04]. More exotic depth techniques, such as depth from defocus [MNB07] or depth from a coded aperture [LFDF07], are applicable for single-camera systems. View interpolation is one application of IBR that generally does not use geometry [CW93]. Here, novel frames are generated from the input images such that a virtual camera is placed along the path between two real cameras. Often, optical flow is calculated to aid the interpolation [HS81, HS93, BBP04]. Depth can be used as an alternative to geometry [ZKU<sup>+</sup>04]. Light fields are data structures which capture all rays of light coming in to and leaving from an area, for a single instant or dynamically over time [LH96, GGSC96]. Once this information is captured, we can form new views of the scene by selecting only the rays of interest.

Some image-based rendering techniques attempt to image across ultra wide field of views, up to 360° horizontally and vertically. Here, captured omnidirectional images are reprojected to allow the viewer to observe the environment with a more natural perspective. Other techniques employ user assisted tools and geometry proxies to create virtual environments from normal and panoramic images [OCDD01, HAA97, KS02, Aut08]. The user is capable of moving into the picture as the geometry proxies simulate parallax and depth movement. Other techniques remain purely image-based [AFYC03, UCK<sup>+</sup>03, SFP10]. These techniques require dense sampling and restrict movement, but retain a consistently higher-quality output than geometry-based techniques as less interpolation takes place.

Figure 2.1 shows a hierarchy demonstrating the relationship between common IBR techniques.

## 2.3 A Brief History of Image-based Environments

One of the first uses of IBR appeared in 1980, when Lippman presented a ‘Movie-map’ of Aspen, Colorado [Lip80]. Movie-maps exploited then new optical videodisc technology to store video of streets taken from a moving vehicle. The system allows the user to interactively navigate a graph representing the streets. A still image is presented at street junctions, which are nodes in the graph (Figure 2.2). Lippman also discusses the use of a spherical lens to afford the viewer a greater field of view. The spherical image is reprojected onto a conical mirror such that it can be viewed undistorted. This pioneering work at the MIT Media Laboratory was continued by Naimark [Nai91] in imaging Karlsruhe, Germany and the Madelaine area of Paris, France. Further work captured stereoscopic panoramas in the



**Figure 2.1:** Image- and video-based rendering hierarchy as presented at the SIGGRAPH 2005 course ‘Video-based Rendering’. Videoscapes portal reconstructions most appropriately fit into the ‘Space/time Coherence’ category of multi-view video analysis/synthesis. Adapted with permission from [MPC<sup>+</sup>05].

See Banff! [Nai94] and Be Here Now [Nai96] projects.

Lippman writes in his conclusion that Movie-maps (and by implication VBR in general) “changes the attitudes of graphics from one of making single images well, to making multiple images better and more efficiently”. This change did not start to occur for many years, and IBR research did not take off until the mid 1990s when digital imaging, storage and computation became affordable.

In 1995, McMillan and Bishop presented ‘Plenoptic Modelling’ [MB95]. They propose a problem definition for IBR as a sampling of Adelson and Bergman’s *plenoptic function* [AB91]. This 7D function describes all light information for a position (3D), direction (2D), wavelength band (1D), and time instance (1D). McMillan and Bishop go on to describe a method for generating a panoramic image by sampling this function. In the same year, Chen et al. presented ‘QuickTime VR’ [Che95]. This system allowed for multiple images to be stitched together, creating a horizontal and vertical 360° panorama. This panorama is viewed by a perspective projection where look direction is controlled interactively. Both methods produce similar outcomes: 2D samplings of the plenoptic function by a panoramic image which is interactively viewed with a perspective correct reprojection.

More adventurous samplings of the plenoptic function soon followed. Two papers in 1996 describe and implement similar structures: the light field [LH96] and the lumigraph [GGSC96], though the idea is much older [Ger39]. Both structures represent 5D sampling of the plenoptic function (i.e., position and direction) in 4D by parametrising rays in free space between two planes. In the case of the light field, display is accomplished by resampling a 2D slice of lines from the 4D light field. This can be accomplished in real-time to allow the user to observe a photo-realistic object/environment from limited angles with correct world effects such as specularities. The lumigraph system additionally makes use of approximate geometry, computed from silhouette, to improve the efficiency of the representation and the quality of the results.



**Figure 2.2:** A user experiences a Movie-map of Aspen, Colorado, circa 1980. Touch screens displaying map (left) and aerial views (right) allow access to additional multimedia material. Image courtesy of Michael Naimark and Bob Mohl; reproduced with permission from Andrew Lippman.

1997 brought the first impactful demonstration of image-based rendering, though not directly of interactive image-based environments. Debevec et al. produced ‘The Campanile Movie’ [Deb97] and its sister paper publications [DTM96, DYB98], to show that simple geometry proxies from image-based modelling and view-dependent texturing could produce convincing depictions of real-world environments. This approach, of using simple geometry proxies for transitions between real and virtual cameras, formed the basis for many research, artistic, and commercial endeavours over the next 10 years (including such films as ‘The Matrix’ [WW99], software tools such as Autodesk ImageModeler [Aut08], and sports broadcast analysis software such as BBC / Red Bee Media ‘Piero’ [Red06]).

In a series of papers starting in 2001, Aliaga et al. demonstrate a 4D sampling of the plenoptic function that is apt for interactive walkthroughs. This collection of work demonstrates the many fields that IBR can encompass, introducing new work in the fields of optics and calibration (catadiotropic systems), tracking (feature globalisation), geometric algorithms (fiducial planning), and compression (spatial hierarchies for images). Culminating in a paper titled ‘Sea of Images’ [AFYC03], Aliaga realises a photo-realistic walkthrough of a static scene using very dense capture sampling. A motorized imaging platform takes omnidirectional images every inch around a real environment. Markers are placed throughout the environment to provide reliable tracking for the platform. A complex pre-fetching and caching system then allows for the massive amounts of data to be viewed interactively.

Kimber et al. introduced the FlyAbout system in 2001 [KF01]. Four small CMOS cameras are arranged in a square to provide horizontal omnidirectional imaging. Video recordings are matched to map data by GPS or by hand. The user is presented with both a map and a look-around window onto the environment. Using the mouse, users can change where they are looking and move through the

environment as the video plays. Alternatively, they can navigate by clicking on the map.

Uyttendaele et al. presented interactive IBR explorations of real-world environments in 2003 [UCK<sup>+</sup>04]. A near-omnidirectional 6-camera system was built for the project by Point Grey Research. The camera is head-mounted and attached to a portable RAID disk array for storage. Users control the exploration of the environment using a joypad, and are given direction choices at manually-set bifurcation points along the path. Dynamic objects, such as fireplaces and televisions, are composited into the photorealistic virtual environment by placing chequerboard markers into the real world during filming. These are replaced during reconstruction by compositing dynamic images generated with traditional computer graphics or with video textures [SSSE00].

McCurdy approached image- and video-based environments from the direction of ubiquitous video, with the RealityFlythrough system [McC07]. This work attempts to situate live 2D video feeds into a 3D space using GPS and orientation sensor data, to provide the user with a sense of how video streams relate to one another spatially. The work introduces and assesses novel interface and visualization techniques for abstracting numerous video streams, and also begins to assess the effect of transitions on switching from one video to another.

Most recently, advances have been made in unstructured video-based rendering. Ballan et al. [BBPP10] present a system which enables smooth blending between different videos which show a spatially confined scene or event. These videos are captured with hand-held cameras, with no prior setup or in-scene calibration. The examples shown demonstrate scenes in which 3 to 5 cameras capture a person performing an action against a natural and potentially cluttered background, for instance, a rock climber or a juggler in a town square. While this work makes large strides in removing the need for image-based environments to use specialized equipment, it only applies to single scenes or events.

## 2.4 Video Collections

Large video collections have long existed as rooms full of film reels and video tapes, and before digitization they provided much slower access times and search facilities. Such video archives, like the expansive BBC archive currently being digitized [IBR<sup>+</sup>09], typically hold *programming* content intended for delivery on television or in cinemas.

With ever-increasing-capacity digital storage mediums and ever-increasing-bandwidth Internet connections, video viewing and sharing services such as YouTube or Vimeo are an inevitable part of the modern Web. These provide instant metadata search results and non-linear access within videos, making it possible to find content much faster than with physical collections. Here, and as in the physical case, the quality and detail of the metadata and index determines how successful a search will be. The content held in online video collections varies greatly: they similarly include programming and music videos; however, with the explosion of mobile devices capable of capturing video, they also contains amateur videos of almost anything and everything. Mobile devices are also sensor platforms, able to capture location and orientation data among other things, and some video softwares are beginning to exploit this additional data in basic ways by geolocating video [Con11].

The current primary way of exploring a video collection is by searching through metadata such

as name, description, rating, date, or popularity. This search is often improved by ranking algorithms. Very few services provide content-based video collection exploration. While this is perfectly functional for finding music videos and clips from named shows, it is much less functional when wanting to find video of a place or an event where the search term is typically less descriptive. In this case, metadata searches are often slow if an exact match is not found, requiring the user to scroll through pages of video and watch or scrub through each video. These searches do not exploit content similarities or useful additional data from sensors. At a fundamental level, collection exploration without content similarities is more limited than with content similarities: There are many possible new interfaces and applications for exploring video collections if we can successfully exploit content-based similarities.

The difficulty in providing content-based similarity interfaces is highlighted by work in the multimedia retrieval and indexing communities (at venues such as ACM Multimedia and ACM International Conference on Multimedia Retrieval). While some work in this field deals with interfaces to video collections [GS11, SB11], it mainly concentrates on the algorithmic efforts of retrieval. Since 2001, the Text Retrieval Conference (TREC) spin-off track, TRECVID, has been collating large test collections and defining uniform scoring procedures for evaluation of research into automatic segmentation, indexing, event detection, and content-based retrieval [NIS12]. These collections and tasks focus on programming content, surveillance, and Web video collections. Creating a representative Web video database that is more than a toy example is difficult [OAS<sup>+</sup>09], and more specialized databases are needed for specific research into sub-problems of generalized content-based retrieval such as for interfaces to video collections of a place or event.

Video Google [SZ03] was one of the first systems to enable retrieval of specific video content from a collection (in this case, a single feature-length movie, though it can be thought of as a collection of video clips or scenes). This system quickly identifies regions of interest, each adapted based on the local contexts of images, such that the resulting feature descriptors represent objects in a viewpoint invariant way. Temporal consistency within video clips is used to track regions and reject unstable regions. While this work demonstrates image-based querying of videos, it does not attempt to provide novel interfaces to video collections which exploit spatial or temporal context.

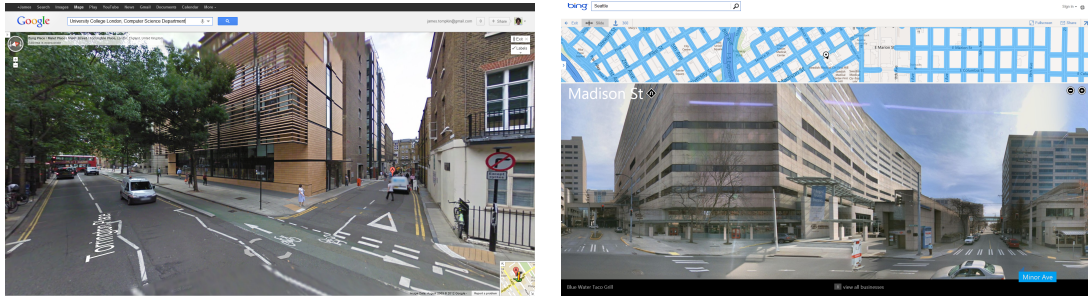
One promising example is presented by Kennedy et al. [KN09], which attempts to automatically organize community-contributed collections of concert videos. They find amateur videos of music concerts and make semantic connections between them by analysing and matching the audio content. However, no such interface exists for video content similarities of places, or of events which may be organized visually and not audially.

## 2.5 Commercial Examples

### 2.5.1 Image-based Rendering and Image-based Environments

Prior to 2005, the two main implementers of IBR research were the games and visual effects industries. Due to its inherent data intensity and inflexibility, IBR is not broadly useful for modern games, though some games in the 1990s and 2000s exploited image-based techniques to great commercial ef-





**Figure 2.3:** Left: Street View was added to Google Maps in summer 2007. Similar views are available from other major search providers. The presented view is a perspective projection of an omnidirectional image, stitched from many camera views. Users can navigate the streets by sparse nodes at tens of metre intervals. Right: Microsoft Street Side allows users to drag the street left and right, viewing a multi-perspective ‘pushbroom’ composite image. Images courtesy of Google [Goo07] and Microsoft [Mic11].

fect [Cya93], including 360° panoramas [Pre01]. IBR techniques have been extensively used in visual effects for over 10 years to integrate real and virtual scene elements. With the notable exception of Quicktime VR, image-based environments have had little commercial success outside of these realms prior to 2007. Quicktime VR’s commercial success was still hindered by the expense of early digital cameras and the bandwidth restrictions of pre-broadband communication, and it became a differentiator for estate agents, travel destinations and exhibitors. However, recent commercial interests in mapping have brought image-based environments back into the public eye.

The broadly termed ‘search’ industries have now ventured into omnidirectional imagery for mapping. While niche applications such as EveryScape and SuperTour Travel [Eve09, Mok06] have existed since 2006 offering such services (both derived from research by Oh in 2001 [OCDD01]), companies such as Google [Goo07] and Microsoft [KCS10, Mic11] offer street-level views from their online mapping applications (Figure 2.3). Currently, users navigate a sparse set of nodes, each of which holds a panoramic image, much in the same way that Lippman proposed in 1980 [Lip80]. Capture is typically handled by a multi-camera system attached to a car which drives through streets. Imagery is captured at intervals of tens of metres for display to the user. Newer systems have also attempted this with video data, notably the GlobalVision system covering parts of Switzerland [Glo09].

Other services have allowed individual users and businesses to create video environments with consumer hardware. Quiksee [Goo10] launched a system which asked users to create node and edge video graphs, with a 360° horizontal pan at each node and bi-directional walking videos between each node as edges. Users would then manually mark portals from nodes to edges, creating the graph (Figure 2.4). Quiksee was purchased by Google in 2010 [The10] and their system was pulled from the Web and has yet to be integrated into Google Maps.

Sports coverage on television now often includes replay analysis which exploits image- and video-based rendering [Tho07, HGK<sup>+</sup>11, Tho12]. Fixed and pan/zoom cameras are calibrated against play or equipment features, such as white lines on a football pitch, to find the relative positions and orientations between cameras. A virtual view may then be created to interpolate between cameras, or to pause the



**Figure 2.4:** Quiksee allowed users to generate video environments by manually mapping out node and edge graphs of places with a single handheld camera. Images courtesy of Quiksee [Goo10].

action for analysis at a novel camera location, for instance, to see whether a player was off-side. This also allows analysis graphics to be overlaid onto video by using simple proxy geometry. Various media companies and software vendors produce competing systems for broadcast use [Spo98, Red06, Haw07, Lib07]; however, the leap to allow home users to download a sports event and control the viewpoint has yet to be made.

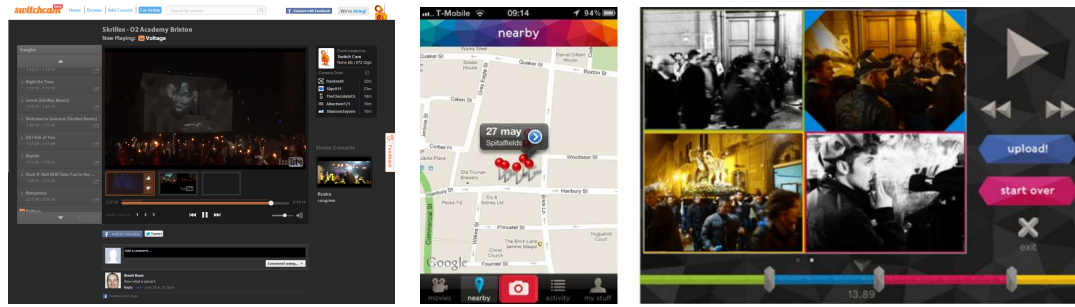
### 2.5.2 Video Collections

Before Web video collections, interfaces for browsing video collections usually resided in video editing softwares. Here, videos are often presented as icons or as rows of thumbnails. Moving the mouse over the thumbnail often scrubs through the video, allowing the user to see all the available content. Professional editing softwares provide a complete data management system, handling time codes, synchronous footage from multiple angle, and stereoscopic data [App12, Avi12, Ado12a]. Consumer tools provide browsing interfaces which are much the same, though usually with less tools and simpler interfaces. However, some support for content browsing is being integrated into video editing softwares. For example, the iMovie ‘People Finder’ feature uses face detection and recognition to allow a collection to be sorted by the people that are present within it [App11].

Another way that consumer editing softwares have tried to provide video collection analysis is with automatic video collection summarizations. These features are born out of research that often exploits spoken audio and closed captioning to provide better summarizations [Chr06]. For consumers, so-called ‘InstantMovies’ [Ado12b] or ‘AutoMovies’ [Mic12] generally try to find high-quality or interesting footage from a collection of videos to automatically create a montage. This is a challenging task, and understandably the results cannot maintain a sense of story; however, they do provide a one-click button for users who have no patience for video editing.

More recently, Web-based interfaces to video collections have allowed text searches through meta-data. Websites such as YouTube [Goo12b] or Vimeo [Vim12] collate millions of videos from almost every possible source and contain many different kinds of content. Text-based search tools parse video titles and descriptions to provide instant access to vast collections. Videos which are related by similar-





**Figure 2.5:** Left: Switchcam website. Up to 3 different synchronized user videos can be selected from underneath the main video window, with a separate audio channel selection from any of the 3 videos. Centre: The Vyclone iOS application attempts to cut together montages of video from synchronized user video cameras. Nearby Vyclone users are identified so that group video recording sessions can be started. Right: If a user does not like the automatic montage, they can montage manually. Images courtesy of Switchcam, The Register [The12], and Vyclone.

ities in these text data are ranked and presented as alternative choices in much the same way as general purpose search engines.

Research similar to Video Google [SZ03] and other multimedia retrieval works have been commercialized in VideoSurf [Vid06], which provides Web-based video content search tools. Undisclosed computer vision algorithms are combined with text searches to augment a more typical Web video presentation interface. Videos may be searched or refined by person and by broad categorical type (exclude slideshows, TV episodes, etc.) and automatic visual summaries of videos are provided as hyperlinked thumbnails for each video. However, VideoSurf does not provide novel interfaces for content based on a specific place or event.

The work of Kennedy et al. [KN09] has been commercialized as Switchcam [SRS11], which launched in November 2011. Switchcam uses videos from YouTube to match the audio tracks of amateur concert videos in hopes of providing whole concerts with multi-angle presentation. Vyclone [Vyc12] attempts something similar, and produces an automatically or manually edited video with changing view-points through cut transitions. Nearby users are highlighted in the Vyclone iOS application, as Vyclone users in the same area band together to start a group recording session. With accurate time codes and central server organization, such an application does not require any audio or video content processing; however, exactly how Vyclone works is not public knowledge. Figure 2.5 shows both of these systems.

## 2.6 Conclusion

Video-based environments provide little flexibility compared to traditional computer graphics. The user's travel is often restricted to the camera's travel, or some subset of cameras which does not break the assumptions of the image-based rendering technique and still produces pleasing results. It is generally hard to edit video-based environments. Simple editing operations in computer graphics, such as relighting, become very difficult to perform on video-based environments. Computer graphics techniques also allow the distortion of reality through simulation or imagination. The breadth of possible content able to be created and displayed by computer graphics is far larger than that which can be captured and presented

in a video-based environment.

However, video-based environments provide a compelling alternative to traditional computer graphics for many applications. Their results are formed from real-world imagery and often provide a vivid experience. In many cases, no visual abstraction is necessary by the viewer and this is a desirable property for many applications which wish to show the real world. Video-based environments suffer fewer problems than traditional computer graphics in presenting real-world phenomenon, such as complex motion, light interactions and volumetric effects. Video-based environments are a very fast method of reproducing a real-world environment and, depending on the data, can often be captured and processed in minutes. Video-based environments also provide a compelling alternative to pure video solutions [Lip80]. The exploratory interaction creates an immersive experience and stimulates a sense of place.

Digital video collections are ever increasing in size and scope. Google Street View-like video experiences already exist in dedicated video collections as these are captured densely and rigorously with specific equipment. Their exploration is paired with specific interfaces to exploit the spatial relationships between video clips. However, collections of Web videos typically provide no content-based interfaces at all. Exploration of these vast databases is by metadata searches, which provide only general information about whole videos and is often inaccurate and incomplete. While some new commercial ventures are exploiting computer vision to provide novel applications for specific capture scenarios, they do not provide interfaces to explore the general relationships between content. Importantly, no existing techniques provide interfaces to explore sparse, casually captured video of content joined by place or time.

In Chapter 3, we explore the state-of-the-art in video-based environments and video collection interfaces to identify opportunities to provide better video collection interfaces.

## Chapter 3

# Literature Review

### 3.1 Introduction

Existing image- and video-based rendering techniques have not yet been extended to work with video collections. Recent work has attempted to construct novel video interfaces for subsets of this problem: a few videos of the same scene [BBPP10], a few low-framerate videos registered with sensor data [NW05], and single tour videos [PWC08]. However, there has been little work thus far which attempts to provide novel interfaces for videos of places or events more similar to what may be found in Web video collections. In the same context, there has been little work to provide novel video-based rendering interfaces for collections with hundreds or thousands of sparse, unstructured videos of a place or event, let alone millions of videos.

Building on the work in Chapter 2, we broadly review work in the relevant fields of content-based retrieval, media collection structuring, and rendering and exploring media collections in Section 3.2. From this, in Section 3.3 we identify key works which provide novel interfaces for videos and collections of videos. We critically assess these works to see whether their assumptions and solutions hold for our scenario, and discuss what requirements can be deduced for the system we will implement (Section 3.4). Finally, this chapter concludes with a summary of the major points discovered (Section 3.5).

### 3.2 Related Work

In this section, we will review representative work from the most closely related fields of research. Any system built to tackle the problem in this thesis will use key techniques include a) feature extraction, matching, and content-based retrieval, b) graph and geometric media collection association and structuring, and c) video-based rendering and video applications and interfaces. This review provides an overview as following sections will detail specific works that are most relevant.

#### 3.2.1 Content-based Retrieval

Finding content correspondence between videos relates to content-based image and video retrieval from an off-line database or a Web database, see Datta et al. [DJLW08] for a survey. To recover these connections, we have to solve a content-based retrieval problem to match individual video frames against a database of video frames. Content-based image and video retrieval [Kat92, FSN<sup>+</sup>95] has received

significantly increasing interest over the last two decades. There are a variety of techniques for organizing, annotating, and retrieving photographs and videos from an off-line database or from the Web [SWS<sup>+</sup>00, CCMV07, SZ03].

Two recent works on video-based retrieval and annotation are noteworthy. *Video Google* [SZ03] is one of the first systems that enables video retrieval. It can robustly detect and recognize objects from different viewpoints and so provides image-based retrieval of contents in a video database. There has also been research into retrieving and annotating geographic locations or spatial landmarks. For instance, Toyama et al. [TLRA03] inferred the spatial location of photographs from metadata such as time stamps, owner labels, and GPS location stamps. Photographs can then be connected to a map and queried based on spatial cues.

Often, visual content is much more useful than metadata content, especially when the latter are not always available. Kennedy and Naaman [KN08] used visual features, metadata, and user-tags for clustering and annotating photographs. The underlying idea is to exploit the metadata and user tags to quickly generate a set of candidates and then to refine the results using visual features. Based on measures of the coherence and the connectivity of the clusters, representative photographs are identified and presented as a summary of a location. Additional related work in image retrieval and annotation can be found in [ZZS<sup>+</sup>09, SMV09].

Our approach, outlined in Chapter 4, is different from the above-mentioned methods in that, except for the videos themselves, no additional information is required. However, when GPS and orientation information are available, we can further embed our video collection onto a map, see Chapter 7. The goal of our work is not pure content retrieval; instead, we want to structure video data such that it can be explored intuitively and seamlessly.

Robust key-point matching could be used for content correspondence identification. This approach has been used in recent work on content-based geolocation of images [BKC<sup>+</sup>10, ZS10, LWZ<sup>+</sup>08]. To increase retrieval performance, Li et al. [LWZ<sup>+</sup>08] build a graph structure — the iconic scene graph — which relates images of a landmark and only contains a sparse set of representative images. In Chapter 5, through spectral refinement we also filter out erroneous portals in our graph, which is related in spirit to identifying iconic images. However, our setting is different since our graph models entire video collections covering many landmarks, and our filtering and matching technique are adapted specifically to our sparse video data.

### 3.2.2 Structuring Media Collections

Since casually captured community photo and video collections stem largely from unconstrained environments, analysing their connections and the spatial arrangement of cameras is a challenging problem. However, the rewards are great, leading to synthesized novel views of locations. Fortunately, we can benefit from the massive amount of media data that is nowadays available for many locations on Earth in community Web platforms.

In their *Photo Tourism* work, Snavely et al. [SSS06] took on this challenge: Given a set of photographs showing the same spatial location (e.g., images of ‘Notre Dame de Paris’), they performed

structure-from-motion to estimate cameras and sparse 3D scene geometry. The set of images is arranged in space such that *spatially confined* locations can be interactively navigated. Recent work has extended this approach to create multi-view stereo geometry reconstructions from photo tourism data [GSC<sup>+</sup>07], to use graph constraints to disambiguate similar visual relations [ZKP10], to find paths through images taken from the same location [SGSS08, GBQV09], and to apply cloud computing to enable significant speed-up of reconstruction from community photo collections [ASS<sup>+</sup>09]. Other work finds novel strategies to scale the basic concepts to larger image sets for reconstruction [FGG<sup>+</sup>10], including reconstructing geometry from frames of videos captured from the roof of a vehicle with additional position and orientation sensors [FPL<sup>+</sup>10].

While some of these problems are parallel to ours, transfer of their approaches to casually captured videos is non-trivial. For instance, a naive application of [FGG<sup>+</sup>10] on a sparse, unstructured video collection cannot yield a full 3D reconstruction of the depicted environment from the video data: we could hardly reconstruct a dense geometry of a wide area of London from a moderate set of casually captured videos. Even with dense video sampling, photogrammetric geometry reconstruction cannot currently deal with real-world scenes with dynamic objects and specularities.

In contrast to previous systems, which attempt to reconstruct a dense geometry for a confined location, our defined approach in Chapter 4 aims to recover and navigate the linkage structure of videos covering a much larger area. As video coverage is sporadic, we reconstruct scene and camera geometry only for specific locations of content correspondence.

Kennedy et al. [KN09] used audio data to align video clips that are known to have been recorded at the same event by different people, e.g., a concert. However, they do not go farther and automatically link networks of videos from unknown locations, nor do they use vision and video-based rendering techniques to compute immersive 3D transitions.

Recently, advances have been made in analysing and representing the connectivity of images as a graph. Philbin et al. [PSZ11] proposed geometric latent Dirichlet allocation, which exploits the geometrical collocation structure of objects in images and thereby enables accurate image matching for specific landmarks. Weyand and Leibe [WL11] proposed an algorithm to select favourite views of an object based on the analysis of how views of it overlap. These algorithms focus on improving pairwise image matching or constructing representative views of image collections. As we will see in Chapter 5, they can all benefit from our analysis of global context in the graph structure.

*Image Webs* [HGO<sup>+</sup>10] constructs and visualizes a graph structure reflecting the large-scale connectivity of images. The system first builds a sparsely connected graph by performing feature-based matching, which is then made incrementally denser via connectivity analysis. In Chapter 5, our graph construction scheme also relies on key point matching followed by connectivity analysis based on the graph Laplacian. However, as opposed to *Image Webs*, we want to filter out unreliable matches rather than to increase graph connectivity.

### 3.2.3 Rendering and Exploring Media Collections

Image- and video-based rendering methods synthesize new views from photos and videos of a scene [Shu07]. This is an important component of an exploration interface as synthesis of intermediate views is required to display a smooth transition between two video views. For more detailed reviews of image-based rendering, readers are referred to the literature [SK00, Zha04, CSDI11].

Image-based rendering methods have been applied to generate interactive 3D walkthroughs from image and video data, often captured with specially fitted camera setups. The pioneering work of Lippman [Lip80] realized one of the first systems for interactive navigation through a database of images. Subsequent research attempted to automate this process. For instance, Kimber et al.’s *FlyAbout* [KF01] captured panoramic videos by moving a 360° camera along continuous paths and synthesized novel views by mosaicking. Users chose a path through a constrained set of automatically pre-computed branching points, and novel view synthesis is required only at these points. We describe heuristics, investigated through a user study, to select appropriate transition rendering styles beyond mosaicking (Section 6.3).

In a *telepresence* context, McCurdy and Griswold’s *RealityFlythrough* [NW05] establishes connections between videos from mobile devices based on GPS information and provides a simple transition between overlapping videos in a manner similar to Snavely et al. [SSS06]. At transitions, videos are projected onto their respective image planes. The view synthesis problem is simplified by allowing the users to choose a path from a constrained set of pre-specified branching points, and by registering the images with a geographical map.

Aliaga et al.’s *Sea of Images* [AFYC03] requires a special robotic acquisition platform and fiducials placed into the scene. As a consequence, the system operates in a spatially confined environment (e.g., a library) where a dense set of views can be easily captured with standard cameras. The extrinsic parameters of the cameras are calibrated with image-plane fiducial locations and bundle adjustment [HZ04]. Novel views can be efficiently generated by sampling from the dense set of input views. Further related approaches exist for navigating through real scenes captured in photographs and videos [DTM96, SFP10]. However, these methods rely on a constrained capture environment (e.g., special hardware or confined spatial locations), which facilitates processing and rendering. In contrast, in Chapters 5 and 6, we exploit vision techniques to automatically find the connections between videos captured under less constrained conditions.

Free-viewpoint video-based rendering research attempts to allow user control over the view onto a scene. This scene is usually captured in a studio, and so is typically a human actor [CTMS03, MHS05, SH07, TAL<sup>+</sup>07, SGdA<sup>+</sup>10]. Tens of calibrated cameras compute shape from silhouette to reconstruct the 3D geometry and texture of the performance. These works have been extended to apply to sports scenarios, where the blue or green screen used for chroma keying is replaced with the green grass playing field [Tho07, HGK<sup>+</sup>11, Tho12]. Other works use denser sets of cameras with narrow baselines to try and provide photo-real view interpolation [ZKU<sup>+</sup>04, LLB<sup>+</sup>10]. However, none of these approaches are appropriate for our sparse, casually captured data as all require specific setup or calibration.

The video browsing system proposed by Pongnumkul et al. [PWC08] provides an interface to create a geographical storyboard from a single continuous video by manually connecting frames to map landmarks. Similarly, Zhang et al. [ZLC<sup>+</sup>10] manually register a single video to a georeferenced 3D model of a city to provide map-based browsing and tracked video annotations. In Chapter 7, our system improves upon these methods by automatically identifying connections between many videos, joining them with visual transitions, and providing video annotations. We also exploit sensor data to provide a richer viewing interface.

The technique proposed by Ballan et al. [BBPP10] enables blending between different videos showing a single spatially confined scene or event. They assume a scene model with a billboard in the foreground and 3D geometry in the background. The background is reconstructed from additional community photos of the scene, and the video cameras are calibrated with respect to the background model. The viewer can transition between the videos of the scene, at which point the proxy foreground geometry is used to find the best possible time to cut to the other video. The system is state of the art, but is tailored to spatially confined sets of videos that all observe the same event at the same time from converging camera angles. It requires dense photo sets as additional input, needs some user interaction, and is streamlined to handle scenes with a clear foreground. In contrast, our system operates with a video collection that shows a variety of general scenes filmed from a much less constrained set of camera positions at different times.

### 3.2.4 Summary

The papers presented do not tackle the problem of creating interfaces for video collections which contain sparse, unstructured imagery of places. Some of the papers discussed are simply computationally unaffordable when applied to video collections, or assume priors on content which we do not. There are papers in each field which solve similar sub-problems of our larger problem; however, each technique must be adapted to fit within an end-to-end system which is capable of automatically structuring a video collection for novel interface exploration. Chapters 5, 6, and 7 will discuss these adaptations in detail as necessary. However, there are some problems in such an end-to-end system for video collections for which no presented solution exist, and many of these problems lie in the exploration of media collections.

As such, from the reviewed works, we identify 3 key papers to be explored in more detail in Section 3.3:

1. McCurdy and Griswold [McC07], *RealityFlythrough: A System for Ubiquitous Video*: This research tries to locate and present live video streams in a 3D space. The work explores the effect of transitions upon users when switching between video streams.
2. Pongnumkul et al. [PWC08], *Creating Map-based Storyboards for Browsing Tour Videos*: This work attempts to provide map-based interface tools for single tour videos. Many of these interface elements may be useful for our problem, and we will see if their ideas extrapolate to multiple videos in a collection.
3. Ballan et al. [BBPP10], *Unstructured Video-based Rendering: Interactive Exploration of Casually*





**Figure 3.1:** The current live video stream is shown to the left, with a geographic map showing camera paths to the right. Each camera is represented by one colour: arrows show the position and orientation of historical views, and view frusta show the position and orientation of the live streams. Image reproduced with permission from [McC07].

*Captured Videos:* This paper provides insights into how vision and graphics techniques may be applied to a few videos of a single event. These techniques are highly relevant for video collection interfaces, but must be assessed with care when applying them to hundreds of videos.

### 3.3 Key Papers in Detail

#### 3.3.1 RealityFlythrough: A System for Ubiquitous Video

McCurdy imagines a system to ‘tap into’ multiple live video feeds to remotely explore the world in real time, and introduces the RealityFlythrough system as a way to accomplish his vision. He focuses on providing the user with a sense of how 2D video streams relate to one another by situating them in a 3D space and by providing transitions when switching between streams. McCurdy uses the compelling example of teams of workers relaying live video back to a control room, for policing, disaster recovery, or remote monitoring. Under these situations, McCurdy assumes limited bandwidth of either IEEE 802.11b or EV-DO wireless protocols in mesh networking setups, for multiple video streams, leaving an effective bandwidth of 100Kbps in a typical 3-camera deployment. As such, the video resolution is CIF (352x288) or QCIF (176x144) and the framerate frequency of the video is at most 11 Hz but is commonly 0.67 Hz or 1 Hz.

The research uses sensors to locate video streams in a common 3D space — each camera is paired with a GPS receiver and a MEMS compass. The data from these sensors provides sufficient information both to allow camera interpolations in a rendered 3D space and to provide a 2D top-down map of all current video streams (assuming a map is provided). McCurdy takes snapshots from the low temporal sampling to progressively add more views of the real world to the map. Transitions use these old images, with added sepia tone or age-indicator bars, as intermediate frames to join spatially separated live streams. Figure 3.1 shows the system in action. Arrows on the map show potential old views, with view frusta showing current live views.

SIFT matching [Low04] to robustly align views is also tested, but this system was too compu-





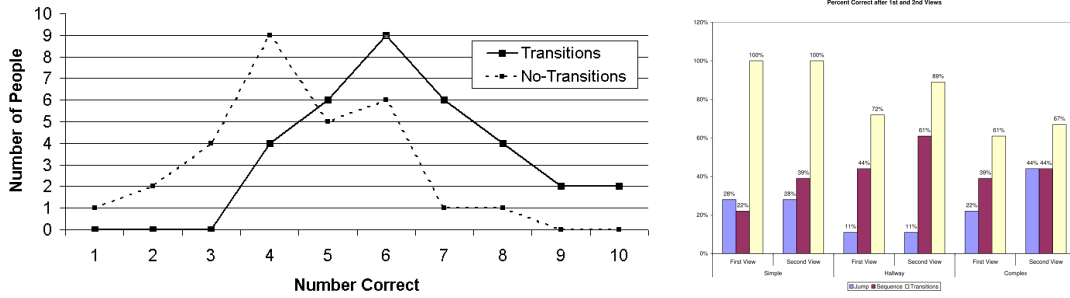
**Figure 3.2:** Top: A transition which exploits scene-to-scene closure, as no intermediate views exist. The motion in 3D space helps users infer the spatial relationship between views even though empty space is visible between the two images. Middle: A transition sequence with intermediate views, using only sensor data. Here, scene elements are misaligned. Bottom: The same transition sequence with SIFT feature matching for robust alignment. The two intermediate views now form a single canvas. Images reproduced with permission from [McC07].

tationally expensive at the time to use in live sessions. However, McCurdy does express the visual improvement that occurs with correctly registered views during a transition. When insufficient intermediary frames exist for a transition, or there is no match, the user sees empty 3D space when switching between video streams. McCurdy suggests that Gestalt closure explains why this effect is not distracting, and that the motion of one frame sliding out of view provides enough sense of motion to produce scene-to-scene closure (as defined by McCloud [McC94]). Figure 3.2 shows this kind of transition, and the improvement that robust feature matching can make to the appearance.

McCurdy describes five experiments which use this system:

**Experiment 1** [McC07, p. 121]: An experiment to assess preference for various video encoding parameters, specifically framerate variations. While not strictly relevant to our situation, the 14 participants generally preferred low framerate video (0.67 Hz and 1 Hz) over middle framerate video (5 Hz) as it was less jerky, but preferred ‘high’ framerate video (11 Hz) over both (see [Wan95, p. 223] for an explanation of this phenomenon). We assume that full framerate (30+ Hz) video is available at all times and that bandwidth is not a restriction.

**Experiment 2** [McC07, p. 148]: Spatial awareness with and without sensor-only transitions is assessed. 11 participants were asked to draw on a paper map as many objects as they could recall from a RealityFlythrough of a house with 3 rooms. 31 spatially located still images represented the rooms virtually, and participants explored the space for 2 minutes. While subjectively assessed, McCurdy notes that the participants who saw transitions when moving about the virtual house covered more space than those who did not see transitions, and so concludes that participants believed they had understood the space more quickly with transitions as they did not linger in any one place. McCurdy also notes that rotation transitions were easier to understand than translation transitions



**Figure 3.3:** Left: *Experiment 3* results, suggesting some improvement in the transition case. Right: *Experiment 4* results, suggesting improvement for the transition case. Figure reproduced with permission [McC07].

given this particular density of images.

**Experiment 3** [McC07, p. 152]: This sensor-only transition experiment asks 30 subjects to choose between 4 different options for the camera configuration of a particular transition, in both familiar (university food court) and unfamiliar (disaster scene) locations. Figure 3.3 shows the results. McCurdy concludes that transitions provided participants with additional information that is beneficial in determining the spatial relationships between cameras.

**Experiment 4** [McC07, p. 159]: More complex transitions are introduced in this experiment. 18 participants assess 3 transitions: a cut, a sequences of dissolved images which are not spatially located, and a 3D sensor-only transition. Three scenes are used: *simple*, a large outdoor space with only camera rotations; *hallway*, a walk down an L-shaped corridor full of people, in a disaster setting; and *complex*, within one room, but containing 180° rotations and some walking, again in a disaster setting. Participants assessed 3 videos for each scene and transition type. The task was to describe aurally how to travel from the first camera to the second camera, and to provide a confidence rating on a Likert scale. Figure 3.3 shows the results. Accuracy and confidence improved slightly with transitions, but it is not known whether this was a statistically significant result [McC07, p. 168]. This experiment did not include an additional top-down map, and so McCurdy concludes that transitions viewed in isolation provide good spatial understanding.

**Experiment 5** [McC07, p. 171]: The final experiment asks 7 pairs of participants to use the RealityFlythrough system to analyse a complicated disaster scene scenario. Participants were asked to answer questions such as ‘how many people are injured and how severe are their injuries?’, or ‘what is the status of potential escape routes?’. This experiment is holistic and presents only anecdotal conclusions about how participants seemed to use the system. McCurdy states that “after spending only 5 minutes in such an environment [participants] can answer detailed questions about what they saw and they can describe with incredible detail what they experienced. It was almost as if they were there, but in fact their experience really went *beyond being there* [HS92]”, as they had gained extra awareness of the environment.

Experiments 2, 3, and 4 provide some evidence which, while not statistically verified, suggests that visual

transitions between imagery are a powerful tool to increase spatial awareness and scene comprehension. Experiment 4 in particular suggests that transitions by themselves are sufficient to express the spatial relationship between video streams, and that a map overview is not essential.

## Analysis

It is clear from McCurdy's presented experiments and experience that the idea of spatially locating video feeds is a useful one, at least for control room/remote worker scenarios. Transitions improve spatial localization and scene understanding, and this is also born out in more rigorous experiments by Morvan et al. [MO09], and Veas et al. [VMK<sup>+</sup>10]. It is also suggested in Experiment 4 that comprehension improved as the fidelity of the transition improved, though McCurdy does not attempt any more complicated transitions involving geometry proxies or full geometric scene representations.

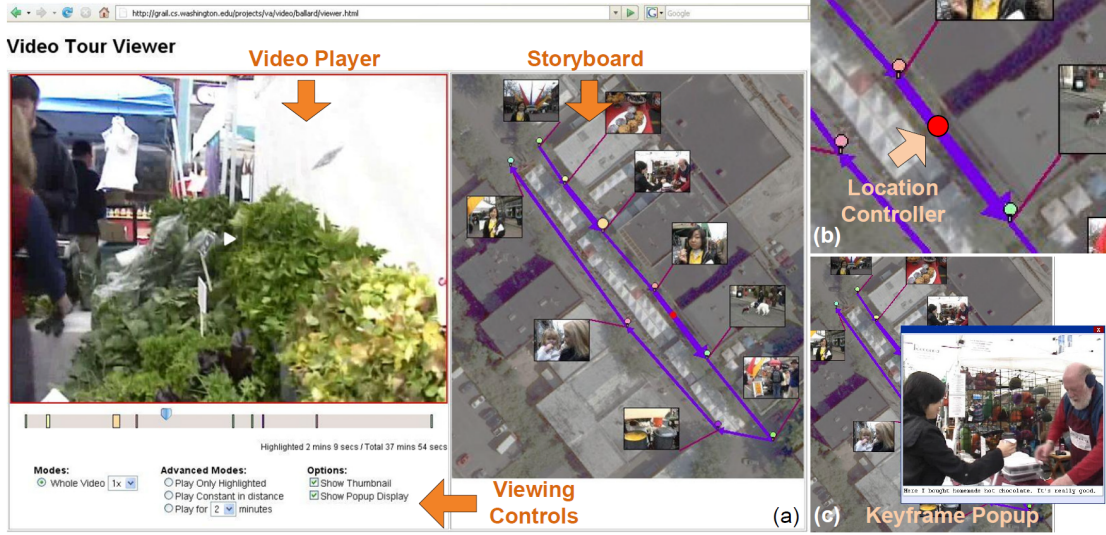
McCurdy's thesis assumes that video must be delivered through a low bandwidth wireless network connection, and so the RealityFlythrough system samples sparsely in time. Our situation is different and we makes no such assumptions, as we wish to work with large collections of full framerate videos. However, an important anecdote is stated [McC07, p. 119]: the presentation of video streams at 1 Hz snapshots was preferred by controllers to higher framerate 5 Hz video, as it was less distracting for decision making. This kind of abstraction is useful when designing interfaces that might present multiple video streams (see Chapter 7).

Some of the techniques presented are directly applicable to existing video collections (for example, SIFT matching for registration) while others require us to posit that GPS and orientation sensor data will be available in future video collections. Given the popularity of smartphones and the potential uses of video with embedded sensor data, we believe it is probable that such collections will exist in the future; however, there will always be a significant amount of video that has no such metadata (largely, almost all currently recorded video) and many situations where sensor positioning is unlikely to work (for example, with video captured indoors). As such, any solution should attempt to build an interface which works with and without sensor data.

For an interface for video collections without position and orientation sensor data, McCurdy's result suggesting that transitions by themselves, without a map, are sufficient to express the spatial relationship between two video streams is an important one. Were this not the case, we would have to try and build a complete spatial representation of a sparse, unstructured video collection using only the visual data within the video collection. While automated mapping techniques (e.g., SLAM) have made great strides in the past 20 years, this is a task outside the scope of this thesis as currently no techniques are able to provide this complete spatial representation from such unorganized and unreliable input.

### 3.3.2 Creating Map-based Storyboards for Browsing Tour Videos

Pongnumkul et al. [PWC08] tackle the problem of tour videos, where a camera moves through a real environment to capture the essence of a place. Often, unedited tour videos are boring, over-long, and less informative than they could be. Typically, hand-held cameras with narrow fields of view cannot give a good perspective onto the spatial layout of a place, and often the tour is unplanned and haphazard. Pongnumkul et al. solve this problem by augmenting the video with a map, by providing map-based tools



**Figure 3.4:** Pongnumkul et al. viewing tool. a) The layout of the UI. b) The pin which identifies the location of the current scene. It either updates automatically when playing the video, or can be controlled by the user for location-based navigation. c) When hovering over a thumbnail on the storyboard, a pop-up window shows a larger keyframe image along with additional text description. Image licensed for reproduction from [PWC08].

to browse the video, by analysing the video for coherent, high-quality shots, and by providing interface tools to fast-forward or skip through low-quality shots.

The technique begins by pre-processing the tour video. Feature points [Low04, MS04, MCUP04] are extracted for 1 frame in every 2 seconds of video, and a SIFT descriptor is extracted for each point. In frames which are sharp, the SIFT response will likely be large as strong image edges will likely be present; conversely, if the video is blurry or out of focus then the response will be small. As such, the quality  $Q_t$  of a frame at time  $t$  is defined as:

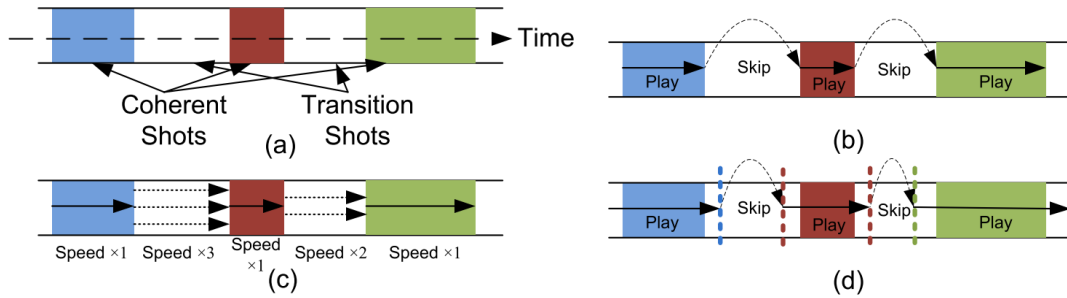
$$Q_t = \frac{\max F - |F_t|}{\max F - \min F} \quad (3.1)$$

where  $F$  is the set of extracted features for all frames, and min/max operate on the number of features per frame in  $F$ .  $Q$  can then be thresholded to find shots of relative quality.

Shot coherence is computed by looking at the number of feature points which match between extracted frames. A 2D coherence matrix is formed, whereby the number of matches between all pairs of frames is computed. All matches in the matrix are projected to the diagonal, and regions along the diagonal with relatively high numbers of matching features are kept as coherent shots. The frame which most strongly correlates to all other frames within a shot becomes the keyframe for that shot. The quality of a shot is defined as the mean quality of its constituent frames.

To geolocate the video over time, users pin shots and their keyframes onto a map by hand. Arrows are automatically added between pins to approximate the path of travel within the video. The size of the pin is proportional to the length of the shot, and the width of the arrow between the pins is proportional to the time interval between shots. Figure 3.4 shows the viewing interface.

The system provides a variety of viewing modes which exploit this information. First, the system



**Figure 3.5:** Pongnumkul et al. advanced viewing modes. a) Coherent shot segmentation. b) High-quality summarization. c) Intelligent fast forward. d) Defined-length video summarization. Figure licensed for reproduction from [PWC08].

can present a high-quality summarization of the tour video by playing only high-quality, coherent shots. Second, the system can provide such a summarization to a required time length, by increasing and decreasing the length of each coherent shot out from its keyframe. Third, the user can employ an *intelligent fast forward*, whereby low-quality, incoherent shots are sped up. In this mode, the sped up shots play such that the speed is spatially consistent, leading to a uniform speed along arrows in the map. Figure 3.5 shows these playback modes diagrammatically.

The paper concludes by presenting an exploratory user study. 13 participants spent an average of 45 minutes using the system, including familiarization and tutorial time, viewing two different tour videos of 38 and 10 minutes for 5 minutes each. Interface controls were instrumented to collect usage data.

Of all interface features, three were used much more than others. In order of most frequent use: playing highlights, timeline slider scrubbing, location controller scrubbing. No significances are provided, but these features were used at least twice as frequently as the next closest feature [PWC08, Figure 8]. Upon asking participants which advanced viewing modes they preferred, the top two in order were playing highlights and map navigation. Pongnumkul et al. postulate that map navigation was not higher because of the limited amount of time in the experiment. Participants suggested that map navigation would be more useful when there was more time.

The more complicated of the two map ‘storyboards’ was found to be confusing, as it was multi-layered to represent different floors of a building. Also, participants did not frequently use the thumbnails to navigate the video as, they postulate, the interface gave instant non-linear access to the whole video. This kind of access was not common for Web videos at the time of publication, though now with progressive streaming this is less of an issue. If users had to wait for video to buffer, cached thumbnails may be used more frequently to explore the video. Some participants also suggested that animations or a 3D layout might be easier to follow. However, in general, Pongnumkul et al. conclude that the feedback about the interface was very positive.

## Analysis

This work has two obvious peripheral limitations that are easily fixed. The first is not having any automatic geolocation. Once location metadata is embedded in a video stream, it should be easy to simplify

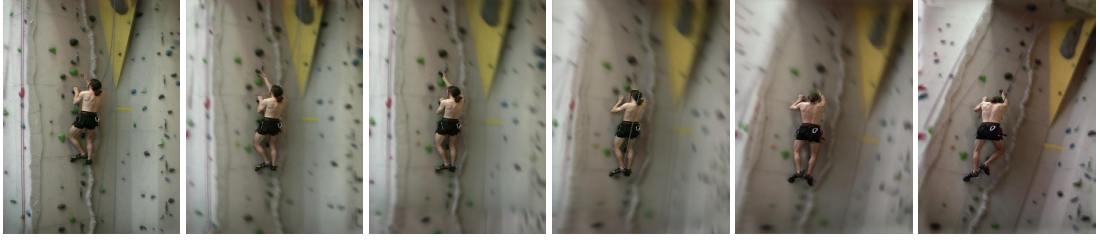
the raw data to show an appropriate path for display. Arrow thickness between pins currently relates to time — this mapping could carry over to the thickness of a real path. The second is that the map storyboard itself also needs to be provided, and with location metadata this could be automatically generated. Admittedly, there is still creative value in allowing the user to specify both pin location and map storyboards themselves.

Conceptually, the metrics for determining coherent and high-quality shots are questionable. These metrics only work well if the content is of a particular type, where low-quality parts of the video also happen to suffer blur or contain few features. This is certainly the case in some footage, but it is certainly not the case in all footage. These metrics become a measure of interestingness or relevance when placed within the intelligent fast forward advanced viewing mode, and we believe this does not hold in the general case. Interestingness is an abstract, subjective measure and does not directly relate to video quality or coherence as defined in this work. It may be possible to deduce better metrics for interestingness from video collections, by assuming that interesting or relevant content appears frequently (or infrequently) among the collection.

Pongnumkul et al. suggest that, for future work, different tour videos captured by different people at the same place could be coupled together on the same map storyboard so that viewers can experience a more complete virtual tour of the area. However, they do not suggest how to overcome any of the problems that this would bring:

1. If we directly apply their method, there would be no content correspondence between different videos within the collection. Parts of videos that contain the same visual information would not be identified, and we posit that this is an important feature of any video collection system.
2. The preprocessing is quite computationally expensive, requiring twice as much time as the video is long. For large video collections, this preprocessing needs to be carefully considered.
3. What does a summarization of high quality, coherent shots look like in a video collection? Is it simply a concatenation of all such shots in the whole collection? Should such shots be montaged by a similarity measure? Should a coherence measure be applied across the whole collection?
4. The pins/arrows representation is an abstraction of the true camera path in the videos. Where are pins placed? From where the video was taken, or at where the video is looking? With multiple videos taken from the same place, or looking at the same content, this abstraction breaks down and can no longer uniquely abstract content in multiple videos.
5. The density of information presented would become large very rapidly, with many thumbnails and coherent shots vying for the same screen space. There needs to be a way to prioritize which information is important across videos in the collection. Further, they also suggest that, in future work, thumbnails could be replaced with video clips. However, with the number of thumbnails present, this runs counter to McCurdy's suggestion that many high-framerate video clips presented at once are confusing. Simple interface elements can solve this problem, like only displaying dynamic thumbnails on mouse hovering, or by providing scrubbable thumbnails.





**Figure 3.6:** Ballan et al. example transition frame subset. The background climbing wall geometry is blurred to accentuate the virtual camera motion, but the foreground climber is not. The foreground climber cuts between the two video views between the 4th and 5th images. Images licensed for reproduction from [BBPP10].

While the naive extension of simply duplicating all existing user interface elements for each video is simple and possible, we feel it would rapidly become an unusable and confusing system.

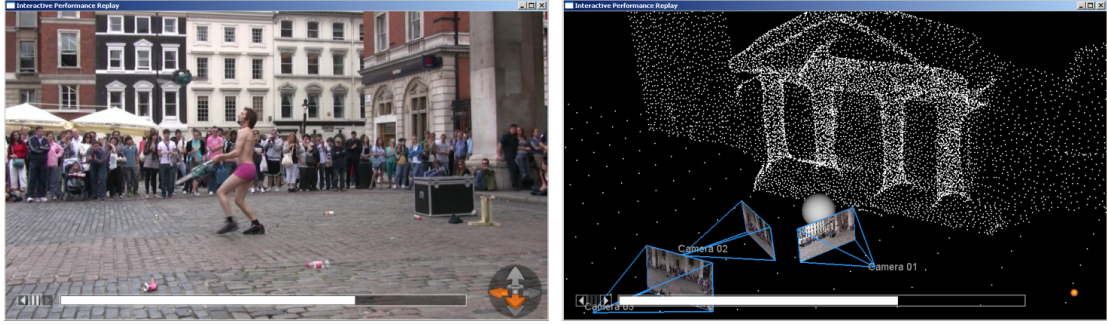
### 3.3.3 Unstructured Video-based Rendering: Interactive Exploration of Casually Captured Videos

Ballan et al. [BBPP10] try to take video-based rendering (VBR) techniques out of the studio and extend them to casually captured, real world, hand-held footage. Starting with a few video streams of a performance, as well as a collection of photographs of the environment, the approach separately models the scene background and a performer in the foreground. The background is modelled as geometry and view-dependent textures, and the foreground as a video sprite on a billboard. A real-time interface inserts VBR transitions when switching between real camera views, taking special care to ensure the performer is centred as much as possible in the virtual camera frame. Figure 3.6 shows a generated transition.

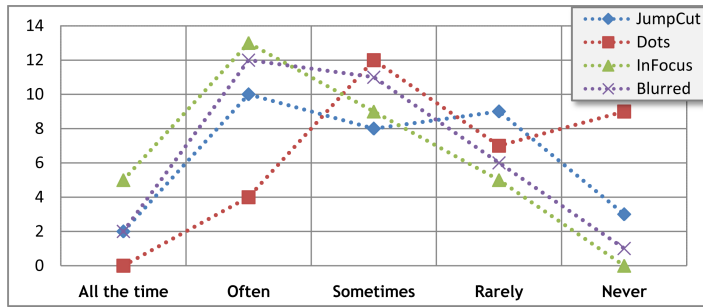
The paper describes an end-to-end system for casually captured videos of a performance, and so presents technical solutions to many problems along the pipeline. First, and as in [HRT<sup>+</sup>09], all videos are synchronized by their audio tracks. Next, the background scene geometry and texture must be created, and the camera poses for each frame of each video recovered. The collection of photographs of the environment are passed through the same structure from motion pipeline as in [SSS06], depth maps are computed using multi-view plane sweep based on normalized cross-correlation, a mesh surface is extracted using range image fusion [ZPB07], and the mesh is textured from the static photographs.

Second, camera poses are estimated by matching SIFT feature points [Low04] in the first stage to SIFT features in each video frame. The 2D to 3D mesh point correspondence found when recovering the geometry is then used to solve the intrinsic and extrinsic camera parameter linear system with the direct linear transform [HZ04]. However, after this stage, camera poses are not sufficiently accurate as similar reprojection errors can lead to a variety of different poses. Ballan et al. make an important point that, for this application, it is only important that the camera pose produces a registration against the geometry that *looks* correct, even if it is inaccurate. As such, a particle filter optimizes the camera pose by minimizing the video frame sum of square difference from the textured mesh.

Further parts of the paper discuss in detail how to segment and matte the foreground object in each video, how to render the foreground object in virtual views, how to constrain the motion of the foreground object to a line in the virtual view, and how to compute the optimal foreground transition cut.



**Figure 3.7:** Ballan et al. interface modes. Left: Regular mode looking down one of the cameras. Arrows in orange allow switching between cameras. Right: Orbit mode looking down upon three spatio-temporally aligned videos. Images licensed for reproduction from [BBPP10].



**Figure 3.8:** Ballan et al. experiment results. Number of users with each preference: 32 users reported how often they would use each transition type. Dots, InFocus, and Blurred refer to the 3 styles used to render the background geometry while transitioning. Figure licensed for reproduction from [BBPP10].

These parts of the paper are not relevant to our scenario as we choose not to model foreground objects at all as we do not assume contemporaneous capture. Finally, the paper describes how to render all parts together during video transitions.

The system presents two interfaces to the set of video: a regular mode, where the viewer looks down the lens of the camera at all times except when during transitions, and can move spatially between videos with arrow icons; and an orbit mode, where the viewer sees the whole scene from a free-floating virtual camera (see Figure 3.7).

Ballan et al. conduct an experiment to discover how often users would switch between videos using various transition types, and to discover whether users liked the two interface modes. 32 participants were asked to view 3 scenes. 4 minutes of instruction were given, and users were asked to fill in a questionnaire after using the system. Figure 3.8 shows results for various transition types, with no clear distinction between transition modes (though dots transitions are less preferred). From this we should consider that many types of transition, even cuts, may be appropriate. Of the two navigation modes, 4 participants preferred regular mode, 3 preferred orbit mode, and 25 liked both. Again, this shows no clear trend.



## Analysis

Ballan et al. demonstrate the first high-quality example of VBR for casually captured videos. Many of the techniques employed will be useful for video collections; however, there are fundamental differences which mean we cannot apply this approach whole.

First, for video transitions, the paper rightly states that accurate camera pose registration is not important, so long as the resulting transition is visually pleasing. However, the approach of improving pose registration by comparing each video frame against the recovered geometry in image space is only possible if the recovered geometry reliably covers all of the video frame. In this paper, the background geometry is recovered in a separate step which does not involve any of the video footage. Photographs are specifically collected for geometry recovery, ensuring conservative coverage. In the general case, this approach will fail: automatic geometry reconstruction cannot yet recover the shape of many structures, for instance, structures made of specular or transparent material such as glass. Equally, requiring special capture of the scene with photographs is infeasible for all places that a video collection might cover. While it would be possible to crowd-source photographs of many places, in general this cannot be assured. As a consequence, orbit mode is not possible in the general case, as it assumes recovered geometry and pose estimation. An approach which provides general interfaces to video collections of places must be able to cope with these failure cases.

Second, in general video collections, there is no implicit assumption of synchronicity. The content in the videos may share a similar background, but not necessarily a similar foreground. For videos in a collection which do share the same time instance (as could be defined by the ability to synchronize their audio tracks), the foreground modelling approach in this work is only applicable in “somewhat” cluttered scenes, where the object or person of interest is giving a performance and is separated in colour or depth from the background. It also requires a small amount of manual labelling per video clip. We cannot expect such a foreground modelling method to work for general video collections.

The experiment conducted in this work draws no strong conclusions as to which transitions are better than others, or which interface is better than the other. From this, we can infer that different transitions may be appropriate at different times and for different video clips. A stylized dot rendering seems to be least preferred among all options, and a simple cut seems as effective as rendered transitions.

## 3.4 Discussion

It is clear that none of these three key papers can be directly applied to video collections of places, even though each of them contains useful constituent parts. We summarize the literature review by defining recommendations for a system providing more general interfaces to video collections.

### 1. Content Correspondence

- (a) SIFT feature matching is robust but expensive, and cannot be directly applied to match all pairs of frames within a video collection ([PWC08], preprocessing). More advanced methods are necessary to deal with hundreds of videos.

- (b) SIFT features and their analysis cannot necessarily find interesting content within a video, though they can find sharp, feature-full video frames ([PWC08], quality/coherence metrics). Content correspondence across videos should be investigated to improve this feature.
- (c) Time synchronization can be performed using the audio tracks of the videos. However, in general, this will only apply to a subset of videos in the collection ([BBPP10], synchronization).

## 2. Transitions

- (a) Spatiotemporally located transitions are liked and can improve spatial awareness ([McC07], experiments 2-4), even though they may not be used all the time ([BBPP10], experiment 1). A system to automatically generate VBR transitions should be included.
- (b) We should not assume that geometry recovery is possible to the extent that it covers all parts of all videos in a collection, so fall-back transitions must exist. However, geometry recovery should be possible in some parts of some videos by exploiting the different views of a place across videos in a collection ([BBPP10], our analysis).
- (c) Foreground segmentation is likely unachievable in the general video collection case ([BBPP10], our analysis).
- (d) We should not assume any additional input data to the system other than the video collection itself. It is likely that, with the proliferation of smartphones, sensor data may be available in the future for videos collections, but this should not be assumed and fall-back interfaces should exist ([McC07] assumes sensors, and [PWC08] geolocates manually). Likewise, we should not assume that an appropriate image collection exists for all places in the video to provide good geometry reconstruction ([BBPP10], reconstruction technique). This will make our approach as applicable as possible for existing and archived video collections.
- (e) Transition comprehension may improve with fidelity ([McC07], experiment 4), and many types of transitions may be appropriate — including cuts ([BBPP10], experiment 1).

## 3. User Interfaces

- (a) Presenting many video streams at once may be confusing ([McC07], experiment 1 and anecdotal evidence with experts).
- (b) Map-based browsing is liked, is useful, and helps scene comprehension ([McC07] and [PWC08], various experiments).
- (c) Pins are not a good way of abstracting the location and orientation of a video as it can only either identify the position of the camera or the position of the content viewed ([PWC08], our interface analysis).
- (d) Any representation of video frames placed onto a map must be density aware. A video collection will contain thousands of scenes or shots, and their representation must be ordered by some measure of importance ([PWC08], our interface analysis).

- (e) Users most preferred watching summarizations of a tour video ([PWC08], experiment), so some equivalent function for video collections may also be preferred.

In the following chapter, we will explain how our proposed solution adheres to these recommendations as it provides interfaces for exploring more general video collections of places and the events within than are presented in existing work.

## 3.5 Conclusion

This chapter presented a review of existing work in the areas of content-based retrieval, structuring media collections, and rendering and exploring media collections (Section 3.2). From this high-level review, we identify and review three key papers in detail, each dealing with end-to-end systems which provide novel interfaces for subsets of our problem (Section 3.3). The first, RealityFlythrough [McC07], locates videos in 3D space with sensor data, and investigates various transitions for scene comprehension. The second, for browsing tour videos [PWC08], finds high-quality, coherent shots to allow automatic summarization. The third, for unstructured VBR [BBPP10], provides high-quality transitions with separate foreground and background elements for a handful of unsynchronized cameras observing the same performance.

Specific techniques from each approach are assessed for their suitability for more general video collections. From this assessment, recommendations are devised for such a system and its user interfaces (Section 3.4). The chapter following will propose a system and discuss how it meets these requirements.

## Chapter 4

# Approach Overview

### 4.1 Introduction

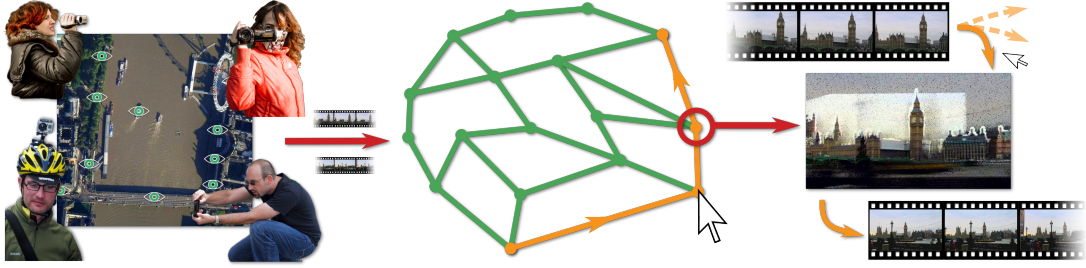
In Chapter 2, we surveyed video-based rendering, video-based environments, video collections, and current commercial uses of these techniques, to identify a opportunity to provide interfaces to explore sparse, casually captured video collections of places or events. In Chapter 3, we reviewed work most relevant to the field on structuring and exploring media collections. We analysed three key papers and drew up a list of recommendations for any system which creates interfaces for such video collections.

In this chapter, we outline our approach to solve some of the problems with existing work and to create a novel system for structuring and exploring sparse, casually captured video collections. We call this system *Videoscapes*. We first overview our entire system in Section 4.3, then explain how each of the recommendations are met by this system in Section 4.4. Next, we explain the system scope (Section 4.5) before finally describing the test databases collected for our system in Section 4.6. The chapter concludes by describing the specific implementation details covered in the next three chapters.

### 4.2 Definitions

We begin by defining some key terms:

1. *Videoscapes* is the end-to-end system that takes as input a video collection and creates both a data structure and ways of navigating that data structure through various user interfaces.
2. A *Videoscape* is the created data structure: a graph capturing the semantic links within the video collection. Edges are video clips and nodes are *portals*. The graph can be directed or undirected, allowing video clips to play backwards.
3. A *portal* is a collection of video frame spans, usually but not necessarily from different videos, which share similar visual content and viewpoints. It represents a potential spatial, temporal, or spatio-temporal transition between one or more video clips.
4. A *support set* is a larger collection of video frames which also share similar content to a portal. However, these frames may vary in viewpoint much more than those of a specific portal, but share enough similarity to *support* the key portal frames during geometry reconstruction and so aid in creating a more complete reconstruction of the content.



**Figure 4.1:** A Videoscape formed from casually captured videos and an interactively-formed path through it, consisting of individual videos and automatically generated transitions. A video frame from one such transition is shown here: a 3D reconstruction of Big Ben automatically formed from the frames across videos, viewed from a point in space between cameras and projected with video frames.

5. The *portal geometry* is the automatically recovered 3D geometry of the scene depicted in the frames at a portal, including frames in the support set.

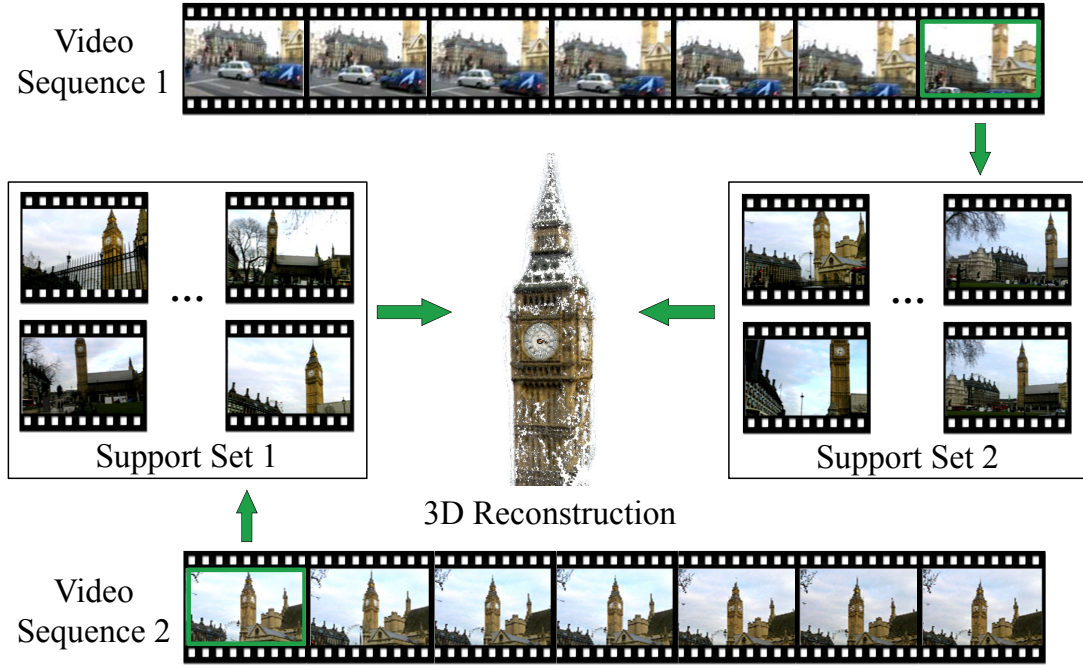
### 4.3 System Overview

Our system has both online and offline components. Chapter 5 describes the offline component which constructs the Videoscape: the graph capturing the semantic links within a database of casually captured videos (Figure 4.1). If necessary, the Videoscape can maintain temporal consistency by only allowing edges to portals that lie forwards in time during walks. The graph can also include portals that join a single video at different times (a loop within a video). Along with portal nodes, we add graph nodes representing the start and end of each input video. This ensures that all video content in the collection is connected to the graph and is navigable. Our approach is suitable for indoor and outdoor scenes.

Input to our system is a database of videos in which each video may contain many different shots of several locations. We expect most videos to have at least one shot that shows a similar location to at least one other video. Here, we intuit that people will naturally choose to capture prominent features in a place, such as landmark buildings in a city.

Videoscape construction commences by identifying possible portals between all pairs of video clips (Chapter 5). A portal is a collection of spans of video frames from any number of videos that shows the same visual content, possibly filmed from different viewpoints and at different times. In practice, we represent the portal by a single frame from each span, forming a set of visual transitions between videos. Long videos, which may contain shots of several scenes, are masked during graph construction into a series of shorter 30 second video clips to provide portal opportunities at regular intervals. This also allows videos to match themselves at different times. In addition to portals, we also identify all frames across all videos which broadly match portal frames. This produces clusters of frames around visual targets, known as the support set, and enables 3D reconstruction of the portal geometry (Figure 4.2).

After a portal and its corresponding supporting set have been identified, the portal geometry is reconstructed as a 3D model of the environment, see Figure 4.2. Video clips in temporal windows around each portal are tracked to find camera poses, and these tracks are combined with the reconstructed geom-



**Figure 4.2:** Overview of Videoscape computation: a portal (green rectangles) between two videos is established as the best frame correspondence, and a 3D geometric model is reconstructed for each portal based on all frames from the database in the support set of the portal. From this, a video transition can be generated as a 3D camera sweep combining the two videos (e.g., Figure 4.1, right).

entry into one coordinate system combining recovered 3D geometry and camera poses. This provides the ability to render dynamic 3D transitions around portals. Chapter 6 contains details of these operations.

Once the offline construction of the Videoscape has finished, it can be interactively navigated in our *Videoscape Explorer*, explained in Chapter 7. This online component provides interfaces to navigate the Videoscape by watching videos and rendering transitions between them at portals.

The explorer provides three modes. The interactive exploration mode allows casual exploration of the database by playing one video and transitioning to other videos at portals. These are automatically identified as they approach in time, and can be selected to initialize a transition. In the overview mode, the Videoscape is visualized from the graph structure formed by the portals. If GPS data is available, the graph can be embedded into a geographical map indicating the spatial arrangements of the Videoscape (see Figure 4.1, left). A tour through the graph can be manually specified by selecting views from the map, or by browsing edges as real-world travelled paths. These tours can be thought of as a geographical summarization of the video collection. A third mode is available, in which images of desirable views are presented to the system (personal photos or image from the Web). Our system matches these against the Videoscape and generates a graph path which encompasses the views. Once the path is found, a corresponding new video is assembled with transitions at portals. Other functions round out the interface, such as label and image searches and fast path-based geographical browsing.

## 4.4 Meeting the Recommendations

Section 3.4 identified 13 recommendations for a system which provides interfaces to video collections. Videoscapes attempts to meet these recommendations:

### 1. Content Correspondence

- (a) *More advanced correspondence methods are necessary to deal with hundreds of videos.* We implement a state-of-the-art coarse to fine matching strategy. It includes filtering for viable frames to match, holistic matching with SIFT bag of words histograms, robust SIFT matching with F-matrix consistency checks by RANSAC, and a final novel graph-based context refinement post-process.
- (b) *SIFT features and their analysis cannot necessarily find interesting content within a video - content correspondence across videos should be investigated to improve this feature.* We identify portals between videos in a collection, where a portal may join many videos which share content. We suggest that interesting content may be captured more frequently, and if so, that portals naturally represent interesting content. The number of videos that a portal joins can be used as a measure of interestingness for that content.
- (c) *Time synchronization can be performed using audio tracks for only a subset of videos in the collection.* Where available, we exploit GPS and orientation sensor data to find viable videos to later match by their audio tracks.

### 2. Transitions

- (a) *A system to automatically generate VBR transitions should be included, even though they may not be used all the time.* We create a geometry reconstruction pipeline which exploits the Videoscape structure by creating support sets. We allow a range of VBR transitions, and experimentally test these for preference to define heuristics for appropriate transitions.
- (b) *We should not assume that geometry recovery is possible to the extent that it covers all parts of all videos in a collection, so fall-back transitions must exist.* Our node and edge scheme for portals and videos is specifically designed to overcome this problem. Where reconstruction is possible, at portals, we allow video switching and provide 3D transitions. Where reconstruction is unlikely in areas of poor coverage, we display only video. Instead of attempting to reconstruct the entire geography, we maintain only the local linkage structure present in the video collection.
- (c) *Foreground segmentation is likely unachievable in the general video collection case.* We define explicit foreground handling as out of scope; however, in experiments, we find anecdotal evidence from participant comments that, for our case and with our data, this not a major factor in transition preference.
- (d) *We should not assume any additional input data to the system other than the video collection itself.* In our baseline system, we make no assumptions about data other than that which



is in the video collection, and can provide automatic collection structuring and exploration interfaces. However, with the proliferation of smartphones, we try to exploit additional sensor data where possible to speed up portal finding and to provide novel interfaces for the Videoscape.

- (e) *Transition comprehension may improve with fidelity. Many types of transitions may be appropriate, even cuts.* We implement and test a variety of different transitions, each either providing a visual variation or improving visual fidelity. We experimentally verify this conjecture of McCurdy.

### 3. User Interfaces

- (a) *Presenting many video streams at once may be confusing.* We create interfaces which do not present multiple playing video streams at once to the user. However, we use scrubbable thumbnails to provide access to every connected video frame when necessary.
- (b) *Map-based browsing is liked, is useful, and helps scene comprehension.* Where possible, when GPS sensor data is present, we provide various map-based video collection exploration tools. These include overview mode browsing, inset mini-maps, geolocated portals, travelled path lines, and view frusta.
- (c) *Pins are not a good way of abstracting the location and orientation of a video as it can only either identify the position of the camera or the position of the content viewed.* We solve this problem with portal eye icons on the map-based interfaces, which abstract content and not cameras. We separate the position and orientation of the camera (displayed as a view frustum) from the map-based representation of the visual content of the portal. Portals, representing shared visual content, are placed geographically above the relevant content, and not at the camera position which views it. A separate exclusive thumbnail area shows the specific visual content, and we couple this portal eye system with frusta and trails to provide interface elements which encapsulate all spatio-temporal camera and content information.
- (d) *Any representation of video frames placed onto a map must be density aware.* The Videoscape graph provides us with a way of ordering the importance of portals, and our map-based portal representation does not overwhelm the screen with thumbnails. We dynamically add content where available, and maintain one dedicated screen area for thumbnails.
- (e) *Users most preferred watching summarizations of a tour video, so some equivalent function for video collections may also be preferred.* We include tours around the graph, which can be thought of as geographic summarizations of the video collection. We experimentally verify that these tours are better than existing summarization techniques in being more interesting, giving greater spatial awareness, and providing a better sense of place.

Chapters 5, 6 and 7 explain in detail the implementation of each of the recommendations in these three classes.



## 4.5 Scope of the Solution

The Videoscapes system explicitly deals with sparse, unstructured video of places and the events within those places. As such, there are entire classes of videos and video collections for which it makes much less sense as an exploration system and for which our system is untested. While Web video collections do contain the kinds of video that our system needs as input, these videos are difficult to find as current online video collection search tools are keyword based and not content based. We conducted preliminary experiments to search for videos of a place with the term “London Big Ben”, on both YouTube and Vimeo. Sampling 100 videos from each, we found very low signal-to-noise ratios of approximately 1:10 for appropriate videos of London, i.e., not video clips from television news or at parties or indoors. This contrasts with Flickr searches for appropriate images for geometry reconstruction, where the signal to noise ratio is approximately 1:1.2. Still, there are videos very much like ours in online collections; unfortunately, they are harder to find than in image collections. Our work does not attempt to solve the problem of sorting and categorizing vast online video collections. This is a complicated and challenging problem, which, once solved, could create input sets for our exploration system.

We also do not address the issue of scale. Web video collections are staggeringly large. Our system can only preprocess hundreds of videos within a reasonable amount of time (a few days, see Section 5.8.2). This is clearly many orders of magnitude away from handling real-world databases. However, our system uses similar approaches and algorithms to existing state-of-the-art works [ASS<sup>+</sup>09, FGG<sup>+</sup>10] and so performs approximately comparably ( $< 5\times$ ) given leeway for the focus on speed and engineering efforts of these works. Most of our pipeline is parallelized, and so our approach would apply well to cloud computing environments.

## 4.6 Experimental Databases

During the project we captured various databases to test and demonstrate our method. Here, we provide capture and basic processing information for each database. Our system takes a database of videos of a place as input. To capture our databases, we distributed video cameras to several people and let them move around the place capturing video. We also mounted video cameras to bicycles. The video collections include different conditions such as a wide variety of spatial locations, changes of date, time, weather, camera, and foreground objects.

In Chapter 1, we use the motivating example of a theme park, where the video collection contains both professionally shot video and amateur video from park visitors. While this would be an excellent test of our system, unfortunately for logistical reasons we did not obtain such a database.

### 4.6.1 London

Our first database comprises 196 videos taken around four locations in London: Big Ben and the London Eye, the Tate Modern gallery and St Paul’s Cathedral, the Tower of London and Tower Bridge, and Museum Street and the Albert Hall. The footage also includes general street footage within each specific area, and two 30 minute walking videos joining a) St Paul’s Cathedral and the Tower of London, and b) the Tate Modern gallery and Tower Bridge. Individual videos feature a variety of motions, and include

small casual movement from one location to another and pans and zooms to take in views. The videos vary in location, date and time, viewpoint, and the presence and variety of foreground objects. The videos in this database exhibit stereotypical camera shake as they were captured hand-held, and this shake is especially noticeable when the camera operator is walking. All videos were captured asynchronously with one camera (Sanyo FH1) at a resolution of  $1920 \times 1080$ , and with heterogeneous framerates of either 30Hz progressive, or 60Hz progressive, or 60Hz interlaced.

#### **4.6.2 South Bank**

In the second database, we employed steadycams to reduce locomotion-caused camera shake, though this is not a functional requirement and only improves the presentation of the database. Where employed, our sensor data was captured with smartphones strapped to the cameras, but all video and optional sensor data could be captured with just one smartphone. We anticipate that cameras in the near future will integrate the required MEMS parts to similarly optionally provide orientation data to our system. This data was captured contemporaneously between four operators over an hour, with no explicit synchronization and with heterogeneous cameras, resolutions, and framerates.

#### **4.6.3 Campus Bike**

Our third database simulates a sports event. Two riders travel around a course with bike-mounted cameras, while three spectators capture their actions. The spectators move around the course to different locations and orientations, taking in the riders as they pass along the track. Here, the cameras were also heterogeneous and recording at different resolutions and framerates.

### **4.7 Conclusion**

This chapter provides a high-level overview of our Videoscapes system. We defined key system terms and introduced our node/edge portal/video structure [4.2](#), explained the offline preprocessing and online interface components of our system in [Section 4.3](#), and declared the broad limitations of our system in [Section 4.5](#). Finally, we reviewed the video collections created for testing our system ([Section 4.6](#)).

The next three chapters will describe the three significant components of our system: content correspondence as portal finding in [Chapter 5](#), creating, rendering, and testing transitions in [Chapter 6](#), and designing and testing interfaces to navigate the Videoscape graph in [Chapter 7](#).

## Chapter 5

# Identifying Portals

## 5.1 Introduction

Finding structures or similar contents within data collections is a problem relevant to many disciplines of computer science. In computer vision, this task often manifests as image search, where descriptive image features are computed for each image in a database and matched against features in a target image (see [LSDJ06, DJLW08] for reviews). More specifically, recent work has tried to cluster images within large image collections to reconstruct geometry of famous places to allow free or guided navigation of image collections [SSS06, GSC<sup>+</sup>07, SGSS08, SSS08, ASS<sup>+</sup>09]. With collections specifically captured for this purpose, it is possible to reconstruct sparse representations of city streets or districts [FCSS10, FPL<sup>+</sup>10] so long as they uniformly conform to well-textured and non-specular assumptions.

It might seem that video collections are a simple extension of image collections; however, there are important differences which make difficult the application of existing techniques for finding similar content, for reconstructing geometry, and for providing access to all content within a video collection. For instance, a naive application of Frahm et al. [FGG<sup>+</sup>10] on a sparse casually captured video collection will be very unlikely to yield a full 3D reconstruction of the depicted environment: the video data simply does not contain enough example shots with sufficient baselines to ensure geometric coverage between landmark buildings. In contrast to previous systems, which attempt to reconstruct a dense geometry for a confined location, our approach aims to recover the *local linkage structure* of videos covering a much larger area, while reconstructing scene and camera geometry only for specific locations at portals.

Other works in structuring media collections improve the accessibility of image collections by exploiting novel connectivity algorithms on a graph of images. Recent approaches also begin with feature-based matching, but then later find favourite views of an object or landmark in a collection [WL11], improve accuracy via geometrical collocation [PSZ11], or improve graph density (the number of links between images) via connectivity analysis [HGO<sup>+</sup>10]. Again, while some of our problems are parallel to the ones solved in these works, none are directly applicable. Our primary goal is to maximize the precision of found portals, as we do not want to incorrectly join unrelated content. This directly contrasts with some existing work, which attempts to maximize connectivity [HGO<sup>+</sup>10]. We wish portal finding to be efficient and applicable to as large a video collection as possible [WL11]. Our connectivity analysis should not be restricted to only a handful of landmarks as a video collection may contain many hundreds

or thousands of content similarities. This is in contrast with some current works in the literature which perform complex analysis on only tens of landmarks [PSZ11].

In this chapter, we detail how we find portals within a video collection. Even a small video collection will have millions of frames, and so we must find portals in a computationally-feasible way. We have two goals:

1. To identify candidate portals and to examine each pair of videos to find the best frames to smoothly move between them.
2. To find for each portal the support set to reconstruct a geometric representation of the content with which to render transitions.

The identification of frame connections is performed in four phases, where each phase reduces the set of candidate portal video frames in the collection:

1. In the *filtering* phase, a set of representative frames  $\mathcal{I} = \{I_1, \dots, I_n\}$  is extracted (Section 5.2).
2. In the *holistic matching* phase, we quickly generates a set of candidate matches on  $\mathcal{I}$  (Section 5.3).
3. In the *feature matching* phase, these candidates are verified using a more costly but more robust local matching scheme (Section 5.3).
4. In the *context refinement* phase, the overall connectivity of the resulting graph structure is analysed and spurious matches are removed (Section 5.4).

From the graph, we select the most appropriate portals (Section 5.5) and finally, for each portal, deduce the support set. These portals, the linkage structure which they form, and their support sets are all later used to recover geometry and render transitions between video clips (Chapter 6) and to provide novel video collection interfaces (Chapter 7).

*The work in this chapter was completed in close collaboration with Kwang In Kim of the Max-Planck-Institut für Informatik. Most of the ideas in this chapter were discussed and devised by both the candidate and the collaborator, with some exceptions. Specifically, Section 5.2 is the work of the candidate, Section 5.3 is joint work, Section 5.4 is the work of Kwang In Kim, is included for completeness, and as such should not be assessed, Section 5.5 is joint work, and Section 5.6 is the work of the candidate.*

## 5.2 Filtering

Naively matching all video frames in a database against each other is computationally prohibitive. As such, a method which can quickly find only a small set of potential matches is essential. Our goal in the filtering phase is to remove the redundancy present across video frames and produce a set of frames which samples all visual content in the video collection, thereby reducing the visual matching computation time.

One approach to this is to pick frames from videos at regular time intervals. However, this often finds too many similar candidates from parts of videos where the camera is largely still, and it may miss

content during fast camera motion. An ideal system would select just enough frames per video such that all visual content were represented and all possible transitions were still found. In this, we assume that repeated content which is separated by other content within the same video still needs to be represented, e.g., a pan left and pan back right to the same content does not count as repeated content as we assume that the camera operator intentionally returned to the same content for a scene- or shot-specific reason.

We implement two approaches to approximately select these frames, depending on which data is available:

1. Optical flow analysis [Far03, EDM<sup>+</sup>08] provides a good indication of the camera motion from just the video frames, and allows us to find appropriate video frames that are representative of the visual content. We compute the frame-to-frame mean flow, accumulate, and select one frame every time the cumulative flow in  $x$  (or  $y$ ) exceeds 25% of the width (or height) of the video; that is, whenever the scene has moved 25% of a frame. We perform this on a frame subsampled to one quarter area for speed.
2. Orientation sensor data (*odometry*) can accomplish much of the same task as optical flow (*visual odometry*). For instance, by integrating data from MEMS accelerometer and gyroscope sensors as commonly found in mobile devices and synchronizing this to the video, appropriate frame selection becomes trivial by accumulating angular change and applying the same 25% heuristic. This overcomes the computational cost of visual odometry but adds a hardware cost.

The 25% heuristic is chosen because, in the difficult case of trying to match content from two videos with contrasting pans, it still leaves a 50% overlap in both video frames from which to accurately match content later on in the algorithm. In the flow case, pure zooms can be detected by contrasting the zero average flow direction with the non-zero average flow magnitude, and here a threshold must be picked. Further, this also detects translations into the scene. Zooms combined with camera rotations are detected by the 25% heuristic. In the sensor case, zooms cannot be detected, and here we await camera focal length metadata to be embedded into videos.

With GPS and orientation sensor data provided, we can further cull candidate frames that are unlikely to provide matches. For example, if we consider camera frusta, frames that have physically-close locations and opposite orientation vectors will never produce a geometric match. However, even though we perform sensor fusion with a complementary filter which makes individual position and orientation readings more robust, we must still cull with respect to the sensor error as sensor data is often unreliable. Sensor filtering allows us to process databases approximately  $4\times$  larger for the same computational cost as processing without sensor filtering.

### 5.2.1 Discussion

Filtering reduces unnecessary duplication in still and slow rotating camera shots. The reduction in the number of frames over regular sampling is content dependent, but in our London database filtering selects approximately 30% fewer frames compared to sampling every 50th frame (a moderate trade-off between retaining content and the number of frames). This leads to a 50% reduction in computation time in

subsequent portal-finding stages.

However, do the frames represent the same content? Testing this comprehensively would require comparing thousands of video frames; instead, we perform a smaller subjective experiment on a subset of frames. For a random selection of frames of the same scene from 10 different videos, we compared the number of frames representing each scene for the regular and the optical flow heuristic sampling strategies. On average, for scene overlaps that we judged to be visually equal, the optical flow produces 5 frames, and regular sampling produces 7.5 frames per scene. This indicates that flow filtering extracts frames more economically while maintaining a similar scene content sampling. The 25% sampling parameter could be optimized for each video to minimize this number, but in general it provides sufficient overlap for later visual correspondence-finding stages of the portal finding strategy while significantly reducing computational cost. The quality of matching in later stages is not affected by the filtering as we set our heuristic to conservatively keep examples of all visual content. Further, for fast pans, regular sampling will undersample the content and our approach will not.

Table 5.1 states the results of filtering for each of our databases. Filtering is not guaranteed to reduce the number of frames over regular sampling as, for example, fast panning will produce more frames under motion analysis to represent all present content. However, in all of our databases it produces at least a modest reduction. At what point does the computational overhead of filtering become pointless? Sensor data filtering takes a very short amount of time, e.g., 7 seconds on our South Bank database, and so is almost always worth completing computationally — only toy databases of a handful of images would not benefit. Flow analysis is computationally more expensive: fast optical flow can compute at 30Hz on modern GPUs [EDM<sup>+</sup>08], e.g., approximately 3 hours on our South Bank database.

At what number of input frames in the video collection  $x$  does flow analysis become cost effective? The number of comparisons made  $c_r$  in an  $n$ -to- $n$  matching of portal candidates in the regular sampling case is:

$$c_r = \frac{1}{2} \left( \frac{x(x-1)}{50^2} \right), \quad (5.1)$$

where  $x$  is the input number of frames in all videos. We only need to compute matches in the upper triangular part of the  $n$ -to- $n$  matrix, hence the division by 2, and the  $-1$  removes the diagonal terms in the match matrix. With fast GPU-based feature matching [Wu07, FGG<sup>+</sup>10], currently in the best case we can compute roughly 10 pairwise matches per second [ASS<sup>+</sup>09], meaning that the time in seconds  $t_r$  required to compute these matches is:

$$t_r = \frac{1}{2 \times 10} \left( \frac{x(x-1)}{50^2} \right). \quad (5.2)$$

Let us now assume a 74% reduction in frames over regular sampling. This is the average reduction seen in Table 5.1 for the flow-filtered databases. In this next equation, here we must also include the time for flow computation, which is at 30Hz. The time in seconds  $t_f$  then becomes:

$$t_f = \frac{1}{2 \times 10} \left( \frac{x(x-1)}{50^2 \times 1.74} \right) + \frac{x}{30}. \quad (5.3)$$

If we equate Equations 5.2 and 5.3, flow-based sampling starts to become computationally worthwhile for video collections with approximately 2500 frames in total, or roughly 1.4 minutes of footage

Database	Number of frames			Filter method
	Regularly sampled	Filtered	Total	
London	4,893	3,508	318,734	Optical flow
– Big Ben & London Eye	1,739	667	89,774	Optical flow
– Museum Street & Albert Hall	1,832	1,756	93,330	Optical flow
– Tate Modern & St Paul’s	678	551	69,478	Optical flow
– Tower of London & Bridge	644	534	66,152	Optical flow
British Museum	1,105	417	55,914	Optical flow
Gordon Square	816	400	41,006	Optical flow
South Bank	6,826	2,651	341,550	Sensor data
Campus Bike	5,938	3,771	297,064	Sensor data

**Table 5.1:** A comparison of the number of regularly sampled frames (at 50 frame intervals) versus the number of filtered frames.

at 30fps. These are only approximate calculations and do not include data transfer times, but it is a good indicator that it is almost always advisable to filter beforehand. Additionally, as sensor-based filtering takes only a few seconds to compute as approximate position and orientation data are provided for every frame of video, this filtering is practically always worth performing.

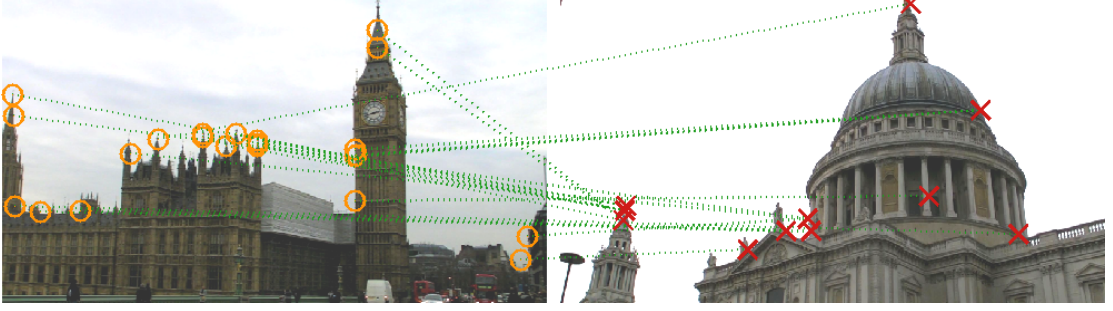
This 74% average reduction in portal candidate frames is only approximate as all motion analysis is content dependent. We can see this in the marked difference in improvement between subsets of our London database, where the Big Ben & London Eye database has many more still shots than the Museum Street & Albert Hall database, and so produces relatively fewer frames with filtering.

### 5.3 Holistic Matching and Feature Matching

In the holistic matching phase, we quickly generates a set of candidate frame matches from the filtered frames. The global structural similarity of frames is examined based on spatial pyramid matching [LSP06]. Here, we use *bag-of-visual-word*-type histograms of SIFT features [CDF<sup>+</sup>04, LM01] with a standard set of parameters: the number of pyramid levels and the size of codebook were fixed at 3 and 200, respectively. The resulting matching score between each pair of frames is compared with a threshold  $T_H$  and pairs with distances higher than  $T_H$  are discarded ( $T_H = 2.2$ , see Section 5.8.2). Performing holistic matching before feature matching has the advantage of reducing the overall time complexity of portal finding, while not severely degrading matching results [HGO<sup>+</sup>10, FPL<sup>+</sup>10, FGG<sup>+</sup>10].

The output from the holistic matching phase is a set of candidate matches (i.e., pairs of frames), some of which may be incorrect. We improve results through *feature matching*, and match local frame context with the SIFT feature detector and descriptor. After running SIFT, we use RANSAC [FB81] to estimate matches that are most consistent according to the fundamental matrix [HZ04], similar to other related methods.





**Figure 5.1:** An example of a mistakenly found portal after the holistic and feature matching phases, with 20 incorrect feature correspondences. Such errors are removed in the context refinement phase. The green lines connecting orange circles and red crosses show the feature correspondences.

The RANSAC algorithm, as well as other approaches for correspondence finding [LH05], has been extensively studied in object detection and recognition research and also in related work on photo collections [SSS06]. However, in general and in our case, it is very difficult to achieve 100% precision in matching images from unconstrained environments.

## 5.4 Context Refinement

The correspondences output from the feature matching stage may still include some false positive matches. Figure 5.1 shows an incorrect match example and demonstrates that these kinds of matches are hard to remove using only the result of pairwise feature matching. When simultaneously examining more than two pairs of frames in preliminary experiments, we observed that correct matches are more consistent with other correct matches than with incorrect matches. For example, when frames  $I_1$  and  $I_2$  correctly match, and frames  $I_2$  and  $I_3$  correctly match, then it is very likely that  $I_1$  also matches  $I_3$ . However, for incorrect matches this is different: if  $I_1$  and  $I_2$  do not match, and  $I_2$  and  $I_3$  do not match, then  $I_1$  and  $I_3$  may still match. Even though incorrect matches may still be correlated, it is less likely that incorrect matches form triangle equalities. We exploit this *context* information and perform a novel graph-based match *refinement* to prune false positives.

This context information can be exploited systematically by applying graph partitioning on the connectivity graph of the matching frames. We first build a graph  $\mathcal{G}(\mathcal{F}, \mathcal{E})$  representing all pairwise matches, where nodes  $\mathcal{F}$  are frames and edges in  $\mathcal{E}$  connect matching frames. Specifically, an edge between two frames  $I$  and  $J$  is added to  $\mathcal{E}$  if holistic matching considers them a valid match. In this case,  $I$  is called a *neighbour* of  $J$ . This graph is different from the Videoscape graph which captures the portal linkage structure (Section 4.2).

Each edge holds a real-valued metric describing how well features of  $I$  and  $J$  match:

$$k(I, J) = \frac{2|\mathcal{M}(I, J)|}{|\mathcal{S}(I)| + |\mathcal{S}(J)|}, \quad (5.4)$$

where  $\mathcal{S}(I)$  is the set of SIFT feature descriptors calculated from a frame  $I$  and  $\mathcal{M}(I, J)$  is the set of feature descriptor matches for frames  $I$  and  $J$ . For pairs  $(I, J)$  filtered after holistic matching, we simply set  $k(I, J) = 0$  instead of performing feature matching. To ensure that the numbers of SIFT descriptors



extracted from any pair of frames ( $I_1$  and  $I_2$ ) are comparable, all frames are scaled such that their heights are identical (480 pixels). Intuitively,  $k(\cdot, \cdot) : \mathcal{F} \times \mathcal{F} \mapsto [0, 1]$  is close to 1 when two input frames contain common features and are *similar*.

Given this metric, we construct the graph Laplacian  $L = D - K$ , with  $[K_{(i,j)}]_{n,n} = k(I_i, I_j)$  and the diagonal matrix  $D_i = \sum_{j=1}^n K_{(i,j)}$ . We perform spectral clustering [Von07] by solving the eigenvalue problem on  $L$ . The first  $m$  eigenvectors that correspond to eigenvalues  $> T_l$ , where  $T_l = 0.1$ , are arranged in a matrix  $G$  and we perform  $k$ -means clustering on the rows of  $G$ . Then, we remove connections between pairs of frames (nodes) that span different clusters. This effectively removes incorrect matches, such as in Figure 5.1, since, intuitively speaking, context-consistent matches will be assigned to the same cluster.

Our graph construction is similar to Heath et al. [HGO<sup>+</sup>10] who used it for the opposite goal of increasing connectivity between matched photographs. Instead, our approach reduces connectivity between clusters by finding incorrect matches. For comparison, 1) [HGO<sup>+</sup>10] uses a binary metric for the construction of the graph Laplacian (1 if a keypoint match is successful, 0 otherwise) while we use a more informative real valued metric (Equation 5.4); 2) in [HGO<sup>+</sup>10], the purpose of building a graph Laplacian  $L'$  is to augment connections, requiring iteration between updating  $L'$  and calculating the corresponding eigenvectors. Our algorithm does not need to iterate.

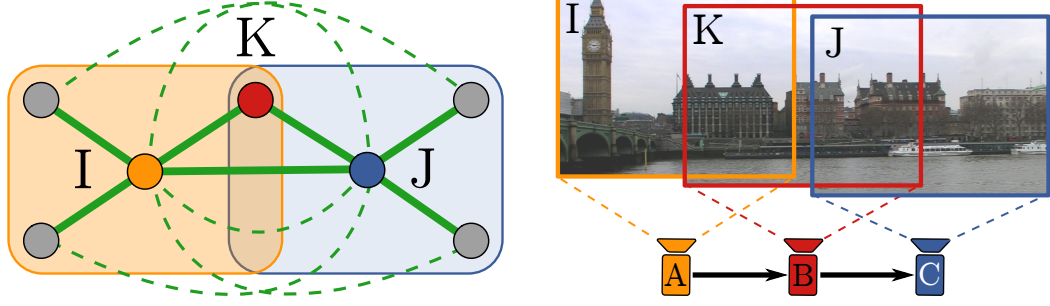
### 5.4.1 Discussion

A naive context-based filtering approach would assign a local context-dependent confidence to an edge  $(I, J)$  and remove it when the confidence is lower than a threshold. For instance, we could define the confidence of  $(I, J)$  as  $\Gamma$ , the degree of overlap of the neighbourhoods  $\mathcal{N}_G(I)$  and  $\mathcal{N}_G(J)$  of  $I$  and  $J$ , respectively:

$$\Gamma(I, J) = \frac{|\mathcal{N}_G(I) \cap \mathcal{N}_G(J)|}{|\mathcal{N}_G(I) \cup \mathcal{N}_G(J)|} \quad (5.5)$$

For instance, if  $I$  is neighbouring  $J$  and  $K$ , it is likely that  $J$  and  $K$  are each other neighbours (see Figure 5.2, left). While this approach may be reasonable when the neighbours of  $I$  and  $J$  consist of frames in a spatially localized scene, it may mistakenly disconnect  $I$  and  $J$  if the camera viewpoints are starkly different. For example, a camera operator walks along a path and takes a panning shot from location A, through to location B, and finally to location C. The footage taken from A and B may contain the same landmark. Now consider location C. The footage from B and C overlaps while the footage from A and C does not (see Figure 5.2, right). In this case, A and B should not be disconnected just because the subgraph composed of A, B, and C shows low connectivity. The same reasoning continues to cases with more than three nodes.

This specific example can be dealt with by adopting a small threshold value for  $\Gamma(I, J)$ . However, this may leave incorrect matches in high-density regions. Furthermore, for edges joining nodes in regions with the same density, we could still distinguish correct matches from incorrect ones depending on how these edges are geometrically collocated. In our A, B, C example, the edges joining the nodes are aligned with the same orientation. This orientation consistency and the variations in local density can be used



**Figure 5.2:** An example of a hypothetical local connectivity-based confidence assignment (not used in the current system). The diagram on the left shows the subgraph of  $\mathcal{G}$  consisting of neighbours of  $I$  and  $J$  respectively. Solid lines correspond to existing edges, while dashed lines show missing edges which would have supported the edge  $(I, J)$ . The corresponding confidence value is  $\frac{2}{6}$ . The diagram on the right shows a case where this confidence assignment would not be applicable.

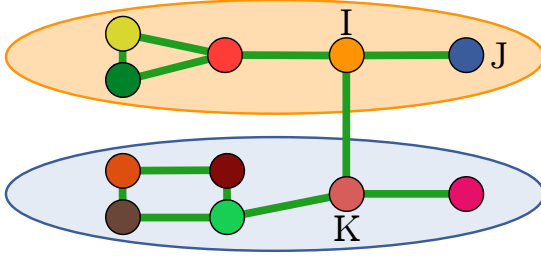
as clues for verifying given connections. To better illustrate this property, let's assume that frames are embedded in a vector space  $\mathcal{X}$  which has a metric structure and an underlying probability distribution  $P$ . Suppose that distribution  $P$  is elongated along a specific axis in  $\mathcal{X}$ . In this case, an edge parallel to that axis should be more likely to be a correct match than ones oriented orthogonally (Figure 5.3). In general, the lengths and orientations of edges do not have to be directly related to real geographical locations and camera orientations as in our example in Figure 5.2.

Given this context, we motivate the use of spectral clustering as follows: given the semi-norm of a vector  $f \in \mathbb{R}^n$ , whose elements represent the assignment of a cluster index (as a real value, before the quantization by  $k$ -means) to each data point:

$$\begin{aligned} \|f\|_L &:= f^\top L f \\ &= \frac{1}{2} \sum_{i,j=1}^n k(I_i, I_j) (f^i - f^j)^2. \end{aligned} \quad (5.6)$$

This norm penalizes the first order variation of  $f$  across the set of frames, weighted by  $k$ . If we assume that  $k$  is inversely proportional to a distance in a space embedding, the frames  $\mathcal{F}$ ,  $\|\cdot\|_L$  can be understood as a measure of the first order variation weighted by the density of  $\mathcal{F}$  in that space. Then, minimizing  $\|f\|_L$  tends to place two points  $I$  and  $J$  in the same cluster (i.e.,  $|f_I - f_J| \sim 0$ ) if there is at least one high-density path connecting them (e.g., nodes lying in the first (upper) cluster in Figure 5.3). Furthermore, when the number of images  $n \rightarrow \infty$ ,  $L$  converges to the Laplace-Beltrami operator on a compact manifold  $M$  in which the data resides [Von07], which is the diffusion generator on  $M$ . The previously mentioned orientation consistency can be understood in the context of diffusion flow. The corresponding smallest eigenvectors span a subspace of vectors which represent the least penalization by  $\|\cdot\|_L$ .

In general, the function  $k$  is not positive definite and does not lead to a distance measure. However, the elements of the matrix  $K$  are positive and, empirically, the corresponding diagonal elements mostly dominate (i.e.,  $\sum_i K_{(i,i)} \geq 2 \sum_{i \neq j} K_{(i,j)}$ ). Accordingly, all  $K$ 's in our experiments were positive definite. When this is not the case, we could instead take its exponential  $e^{\beta K} = \lim_{n \rightarrow \infty} (I + \frac{\beta K}{n})^n$  with a positive constant  $\beta$ , which is always positive definite (see [KL02] for details).



**Figure 5.3:** Example graphical embedding of frames and their connections. Even though  $(I, J)$  and  $(I, K)$  show the same local connectivity,  $(I, J)$  is more likely to be a correct match than  $(I, K)$  since the former is in accordance with the flow direction (elongatedness) of the distribution while the latter is not. The underlying distributions  $P$  (displayed as ellipses) are not known and we must estimate them from the video frames.

The clusters from spectral clustering cannot be used by themselves to identify portals or support sets of frames matching portals: By design, a cluster contains spatially distinct data points. This is not desirable for identifying portals or for identifying sets of appropriate frames to use for the corresponding portal geometry reconstruction. In our A, B, C example, the frames of scene A might not be necessary for the reconstruction of scene C (Figure 5.2, right), but they might be in the same cluster. Sections 5.5 and 5.5.1 address portal and support set selection.

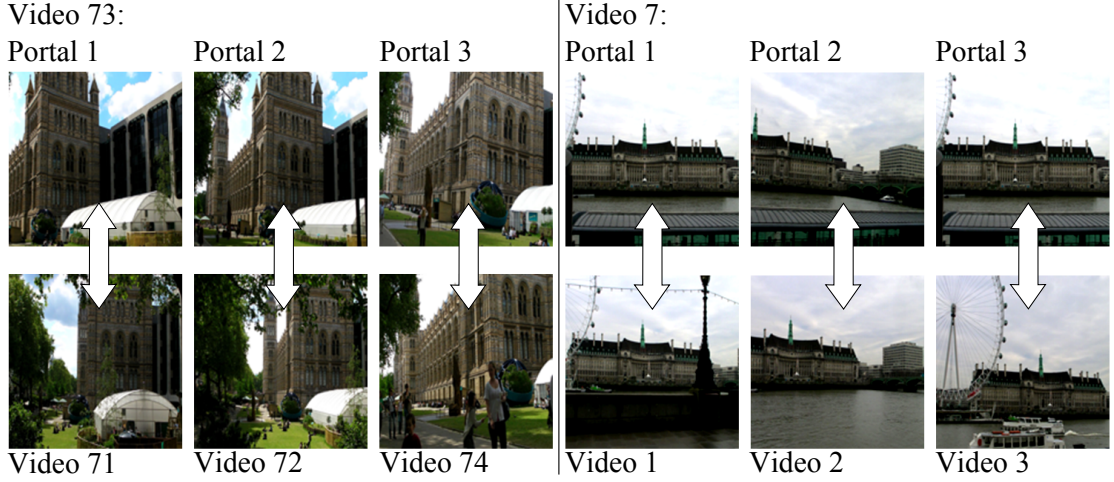
## 5.5 Portal Selection

The matching and refinement phases may produce multiple matching portal frames  $(I_i, I_j)$  between two videos. If the database contains many videos, the resulting quantity of matches makes visualizing and exploring the graph structure very difficult (see Section 7.2.2 for more discussion). We side-step this problem by retaining only the best available portals between a pair of video clips. This might seem to be too few portals joining videos as each video may be long and take in many different scenes. However, we mask each input video into 30 second clips so that there will be portals at approximately regular intervals in time in each video.

A good portal should have many visual feature matches and allow for a transition which maintains spatial awareness between videos. This is more likely for frame pairs shot from similar camera views, i.e., when there are only *small* image-space displacements between matched features. To this end, we enhance the metric from Equation 5.4 to favour such small displacements and define the best portal as the frame pair  $(I_i, I_j)$  that maximizes the following score:

$$Q(I_i, I_j) = \gamma k(I_i, I_j) + \frac{\left( \max(\mathcal{D}(I_i), \mathcal{D}(I_j)) - \frac{\|M(I_i, I_j)\|_F}{|\mathcal{M}(I_i, I_j)|} \right)}{\max(\mathcal{D}(I_i), \mathcal{D}(I_j))}, \quad (5.7)$$

where  $k$  is from Equation 5.4,  $\mathcal{D}(\cdot)$  is the diagonal size of a frame,  $\max(\mathcal{D}(I_i), \mathcal{D}(I_j))$  is the largest possible displacement between two corresponding features,  $\mathcal{M}(\cdot, \cdot)$  is the set of matching features,  $M$  is a matrix whose rows correspond to feature displacement vectors of verified matches,  $\|\cdot\|_F$  is the Frobenius norm, and  $\gamma$  is the ratio of the standard deviations of the second and first summands (excluding  $\gamma$ ), and balances the contributions of the two terms. The intuition behind this score is that, for a given fixed number of matches (the first summand), the score should be inversely proportional to the mean norm of matching vectors, i.e., the displacements between matching features in two frames should be small. This



**Figure 5.4:** Examples of portal frame pairs: the first row shows portal frames extracted from two different videos in the database (7 and 73), while the second row shows corresponding matching portal frames from other videos. The number below each frame shows the index of the source video in the database.

satisfies the criterion that any video transition through this portal should have small displacements to be visually more plausible. Figure 5.4 shows examples of identified portals (see Section 5.8.2 for experimental setup details). Videos with no portals are not included in the Videoscape, though these could be included for completeness as separate graphs with start- and end-frame portals and one edge.

In practice, Equation 5.7 finds portals that are the most-similar camera poses between candidate frames from two video clips. On average, these portals cover view changes of approximately  $35^\circ$ , but this is very content dependent. Some portals have no angular baseline, others have only a zoom difference, and many have approximately  $10^\circ$  or less baselines. As we always pick the best matching portal between two video clips, there are extreme cases where the match is under a wide baseline of over  $60^\circ$ . The performance of portal finding is evaluated in Section 5.8.

### 5.5.1 Support Sets

We define a support set of frames for each portal to be used for geometry reconstruction. The support set for a portal node contains all portal frames from neighbouring nodes in the Videoscapes graph which also belong to the same spectral cluster as the portal, as discovered during context refinement (Section 5.4). For our London database, the average size of portal support sets is 20 frames.

Support sets can be augmented by including neighbours of neighbours. Recursively extending the support set in this way tends to fill in reconstruction detail for objects surrounding the portal landmark. For instance, if a portal showed different views of Big Ben, then recursively extending the support set would add more detail to the final reconstruction of the neighbouring Palace of Westminster building simply because videos which typically cover this popular area are highly interconnected. This helps to cover more of the virtual view with recovered geometry in the final rendering. Including neighbourhood support sets two edges away from a portal in the graph of support set frames increased the average size to 45 after removing duplicates, while including neighbourhood support sets up to three edges away increased it to 70.

The larger the support set used for geometry reconstruction, the longer the process; however, in our experience, often the geometry reconstruction improves as more frames are added. Additional frames which are very similar in pose do not improve the reconstructions. In our case, due to our filtering steps in Section 5.2, very similar frames are removed and so usually additional frames added in neighbourhood augmentations are of different poses. Including all neighbourhoods recursively does not produce a complete geometry reconstruction of the video database due to varying video coverage. Instead, the graph linkage structure maintains global navigability. We choose to use support sets extended by two neighbourhoods, as this was a good compromise between computation speed and reconstruction extent for our London database.

## 5.6 Synchronization

### 5.6.1 Sensor Data

For our London database (Section 4.6), we captured GPS position data with a mobile phone along with video from a camcorder. The internal clock of the video camera was manually synchronized to GPS time before capture. This produced sensor synchronization to within a second which, when coupled with the GPS sensor update rate of 1Hz, confers to a maximum two-second synchronization error. This database was captured on foot: it is unlikely for the position data to be significantly in error due to synchronization errors because it is hard to move very far in two seconds on foot. Of course, GPS is not an infallible positioning system, especially on a mobile phone: this position data is only broadly correct being accurate to 5-20 metres, and might give inaccurate readings in areas with many tall buildings. We do not attempt to tackle these problems; in principle, the Videoscape graph could be exploited at nodes to jointly estimate the geographical camera pose and 3D structure, rather than just the local spatial camera pose and 3D structure of the videos, but we leave this for future work.

For our South Bank and Campus Bike databases, we additionally captured orientation sensor data from gyroscope and accelerometer MEMS chips on a mobile phone. This data is harder to synchronize because rotation has a lower tolerance for error than position (a human can feasibly rotate  $360^\circ$  with a camera in one second), and so frame-exact synchronization is required. We hand-synchronized position/orientation data to video frames: position/orientation data is shown on a map side-by-side with the video, and sliders allow changing the time offset between video and sensor data. Synchronization is most easily achieved by matching the zero-velocity points of rotation arcs (similar to the equilibrium position of a pendulum). We expect that, in the future, this synchronization will be unnecessary as either a) MEMS circuits will be integrated into video cameras and position/orientation data will be provided as metadata to videos (or existing sensor circuitry currently used for optical stabilization will gain consumer/developer interfaces), or b) mobile phone video cameras will improve. Recent advances by Nokia with their PureView cameras [ADS12] show that b) is possible, and over the next 5 years this technology is likely to become commonplace and make mobile phones both capable video capture devices and capable sensor platforms.

A frame-exact synchronization of mobile phone sensor platform and video camera could be

achieved using audio. While no software currently exists on mobile marketplaces to do this (and we did not write one), it would be possible for the phone to generate a loud beep at a certain known time, which is then picked up by the microphone on the video camera. This would provide a short (frame-accurate) reference for synchronization. Alternatively, exact time from GPS could be recorded for every video frame. As this is accurate to 10 nanoseconds [Phy], it should be possible to perform frame-exact video synchronization. For our application, the device which captured video was not the device which received GPS signals, and so this synchronization was only loose.

### 5.6.2 Time

To provide temporal navigation, we perform frame-exact time synchronization between videos in the collection. We group video candidates by timestamp and GPS location if available, and try to synchronize their audio tracks similar to Kennedy et al. [KN09] and Hasler et al. [HRT<sup>+</sup>09] using off-the-shelf software [Sin11]. Videos which are positively matched by their audio tracks are aligned accurately to a global clock (defined from one video at random); hence, portals between these videos create spatial transitions where time does not change (similar to those from Ballan et al. [BBPP10]). Videos which are not matched by their audio tracks can only be aligned loosely from their timestamps, and hence create spatio-temporal transitions. This information will be used later on to optionally enforce temporal coherence among generated tours and to indicate spatial-only and spatio-temporal transition possibilities to the user (Section 7.2.1).

## 5.7 Pipeline

The pipeline for portal identification, with optional stages included, is described in Algorithm 1.

## 5.8 Experiment: Context Refinement and Portal Identification

We captured many databases to demonstrate our method (see Section 4.6); however, here, we provide a detailed analysis of portal identification on only the London database. The processes used for each database are virtually identical, with all parameters kept the same and the only difference being which filtering method is used.

### 5.8.1 Context Refinement

We investigated the performance of the graph Laplacian-based connectivity analysis method by comparing it to the local analysis approach described in Section 5.4.1. For this algorithm, we randomly sampled 100,000 edges, measured their scores, and removed them when the scores were smaller than a threshold. The threshold was set at 0.4 to result in recalls comparable with our proposed method. Since the order of visiting edges can affect the results, we performed the same experiment 20 times and averaged the error rates. Table 5.2 shows the results. The precision of local analysis shows an improvement over the results obtained without any connectivity analysis. However, this is still a lower precision and recall than that of our Laplacian-based graph method.

---

**Algorithm 1:** Pipeline for portal identification including optional stages and default values for our databases.

---

**Data:** A video collection.

**Result:** A Videoscape graph (Section 4.2).

**foreach** *video* **do**

    Synchronize sensor data by Section 5.6.1;

    Time synchronize by Section 5.6.2;

    Mask video into 30s clips;

    Filter video frames (25% frame diff.) to find portal candidates with flow by Section 5.2;

*If present* filter with sensor data;

**foreach** *pair of portal candidates* **do**

    Perform holistic matching with SIFT bag of words,  $T_H = 2.2$ , by Section 5.3;

**foreach** *pair of holistic matched candidates* **do**

    Perform feature matching with RANSAC/F-matrix by Section 5.3;

Perform context refinement,  $T_I = 0.1$  by Section 5.4;

Perform portal selection for  $n$ -to- $n$  clips by Section 5.5;

**foreach** *portal* **do**

    Compute supporting set by Section 5.5.1;

---



Phase	Recall	Precision
Spectral analysis	0.53	0.98
Local analysis	0.51	0.95
No connectivity analysis	0.58	0.92

**Table 5.2:** Performance of spectral and local connectivity analyses. ‘No connectivity analysis’ corresponds to just the holistic and feature matching phases (see Table 5.3). Our spectral analysis reduces the portal match errors seen by users from 1 in 11 to 1 in 50.

### 5.8.2 Portal Identification

To gain insight into the performance of portal identification, we measured the precision and recall for a random subset of our London database. A comprehensive evaluation of an entire database is infeasible since it requires manually labelling  $O(n^2)$  pairs of images.

Precision was measured from all identified portals connecting to 30 randomly selected videos. The corresponding frame matches were visually inspected and portals were labelled as ‘correct’ when matching frames represented the same scene. It is not always possible to deterministically evaluate the correctness of a match, for instance, when one image contains a wide view of Big Ben while the corresponding matched image contains only a small portion of it. Accordingly, we introduced an *undetermined* case which is not counted during evaluation, though in the described instance it is arguable whether it really matters at all — visually, it is still a good match, even if it may lead to inconsistent virtual camera motions during transitions (or in other cases, e.g., where frames containing a repeatedly symmetrical terrace-like building are confused, the virtual camera motion could be a pan that is too short/long).

To calculate recall, 435 randomly selected pairs of video clips were visually inspected to see if their scene content overlapped. Again, ground truth portals were identified as ‘found’ when there was a corresponding automatically identified portal. Table 5.3 proves the importance of each phase of portal finding (the threshold for the holistic phase was fixed to  $T_H = 2.2$ , see Section 5.3). Using only holistic matching, a high recall can be reached but precision is low. When using holistic matching only, we cannot use the score from Equation 5.7 for choosing the best frames because it expects fundamental-matrix verified feature matches to compute feature displacements; instead, the pyramid matching score is used. Adding feature matching leads to a drastic increase in precision (holistic & feature matching 1). Finally, all phases together ( $T_I = 0.1$ ) yields a precision of 98% and a recall rate of 53%. Even though combining all steps leads to a slight reduction in recall, this serves our purpose as our objective is to minimize false positives.

For comparison, it is possible to achieve the same precision with feature matching (holistic & feature matching 2) by simply thresholding the number of key correspondences. However, this lowers the recall considerably, indicating the reduction of the size of the support sets and hence reducing the ability to reconstruct 3D models needed for some transitions.

All these parameters can influence the accuracy and computation time of portal identification. How-



Phase	Recall	Precision
Holistic matching only	0.84	0.14
Holistic & feature matching 1	0.58	0.92
Holistic & feature matching 2	0.42	0.98
All (holistic & features & context)	0.53	0.98

**Table 5.3:** *Performance of portal identification.*

ever, for a fixed  $n$ ,  $T_H$  can roughly be regarded as both an accuracy control and a computation time control (with these two properties being inversely proportional), while  $T_I$  can be regarded as a control to trade between precision and recall.

Reaching 100% precision with automatic methods is nearly impossible and even analysing context information through graph-based refinement cannot completely rule out these errors. For these rare cases, the virtual tourist can manually flag the remaining incorrect portals in the interactive viewer. False matches due to symmetric buildings could be further disambiguated with recent advances in vision [ZKP10, HS12], but even here there will still be errors.

On our London database of 196 videos, all portal identification steps took approximately four days on one Xeon X5560 2.66GHz (using one core). Most of this time is spent in accurate feature matching, and spectral refinement takes only a few minutes. Using filtering instead of regular sampling saves two days of computation. 232 portals were found. Except for the first phase, specifically the codebook generation, the off-line procedure could be executed in parallel.

## 5.9 Conclusion

The chapter describes the stages necessary to transform a collection of videos into a Videoscape graph. First, we filter video frames to isolate individual camera motions, usually pans and still sections. From this, we ensure that we sample all video content as sparsely as necessary, and so produce a set of candidate portals. Next, we perform a holistic matching upon these candidates to further reduce potential content matches, before a more accurate (and more computationally expensive) geometric feature match. This stage still leaves errors, so we develop a graph-based context refinement method to remove incorrect matches. Finally, we select Videoscape portal frames by defining a joint context-and-correspondence score. We experimentally assess our approach against existing methods and find that we achieve high precision with better rates of recall than existing methods.

With the Videoscape graph defined, we can now walk the graph and cut between videos that join at portals. However, the portal frames are still sufficiently different that there is a harsh discontinuity when switching clips. The next chapter attempts to remove this harsh switch by using vision and graphics techniques to render intermediate video transitions.

## Chapter 6

# Video Transitions

## 6.1 Introduction

For over a hundred years audiences watching movies have become familiar with the effects of placing video clips in sequence. The switch between clips is called a transition and, while this is most commonly an instant transition or *cut*, various transitions exist to represent information and create effects in the mind of the viewer. Transitions were an artistic introduction which allowed movies to transcend the restrictions of time and location placed upon theatre (as well as, though unimportantly, the physical restrictions of the amount of film in a reel). Since then, advances in physical and digital visual effects have given movie-makers creative freedom over both discontinuous and continuous or *seamless* transitions.

It would be incorrect to assume that movie transitions are suitable for joining video clips in a sparse, unstructured video collection of a place. For movies, the most significant goals of a transition are to drive emotion and story, whereas the least significant goals are to maintain the two- and three-dimensional *space of action* [Mur01, p. 18]. We consider it important to seamlessly maintain the sense of orientation in the viewer when transitioning between video clips of a place by relating the two- and three-dimensional space of action, else the viewer will become lost in the environment.

Cut transitions represent a change of context and are effective when visual displacement is great [Mur01, p. 6], but rules for their use do not focus on maintaining the space of action [Mur01, p. 18][McC07, p.101]. *Dissolve* transitions suggest a change of place or the passage of time [Dmy84, ch. 13] and likewise do not intentionally maintain the space of action. *Hidden cut* transitions maintain the space of action and disguise the change of clips in pans or zooms across featureless surfaces<sup>1</sup>. At other times, movie-makers create seamless transitions by employing computer generated visual effects to create physically implausible camera moves, such as simulating a camera moving through the lock in a door [Fin02]. Recent advances in computer vision and mapping have required new photo transitions not previously seen in movies [SSS06, SGSS08, GAF<sup>+</sup>10], and these can be adapted for video. Which transitions are most suitable for video collections of places?

This chapter attempts to determine exactly that: the most suitable video transitions for exploring sparse, unstructured video collections. First, we choose and justify a selection of movie and graphics-

---

<sup>1</sup>This is necessary to overcome technical or logistical problems such as seamlessly transitioning between a location and a sound stage [Hit63], or in the past to achieve long takes when the length of film in a reel was a problem [Hit48].

rendered video transitions to compare in an experiment (Section 6.2). Then, we explain the transition implementations, show how the Videoscape graph enables their computation, and assess their possible artefacts (Sections A and A.7). Videos are often captured with hand-held equipment, so we address how video stabilization fits into our system (Section A.8). Next, we psychophysically assesses transitions for participant preference. With these experiment results, we generate heuristics which suggest good transitions and we move towards an automatic scene-dependent transition selection system (Section 6.3).

## 6.2 Experiment Design

We wish to study viewer preference between video transitions. Broadly, there are three variables which affect this task: 1) perception and scene understanding; 2) all possible start and end video clips; and 3) all possible video transitions. In cinematic literature, different transitions have implied meaning, and the possible interpretations of scene and camera movement are numerous. However, our goal is only to maintain spatial awareness across transition between videos.

To restrict the problem to be within the scope of this work, we will sample from 2) and 3) and pick example start and end clips representing scenes which commonly appear in our databases. We will also pick a selection of transitions which span the language of film and the computer graphics literature. Each transition type will be compared against every other, and this will occur across different scenes. This approach should provide sufficient insight to generate rules which pick an appropriate transition to join two videos. We must carefully balance the number of scenes and transitions, as well as pick an appropriate methodology, so that the experiment can be completed in a reasonable amount of time. For instance, with five second videos displaying ten scenes, and with seven transitions each in a paired comparison, if we ignore repeatability and approximate five seconds for the participant to decide preference, then the experiment would last approximately one hour; with ten transitions, the experiment would last approximately two hours.

We consider three experiment methodologies: paired comparison, categorical judgement, and ordinal ranking. Paired comparison asks participants to repeatedly choose a preferred option from neighbouring pairs (or to state that there is no preference between the pair). Paired comparison is the most thorough and time-consuming of the three methodologies as  $\binom{n}{2}$  explicit comparisons are required. Categorical judgement asks participants to provide ordinal numbers for each choice on a rating scale. This approach is faster as each test case need only be observed once (rather than  $n - 1$  times), but the category choices may be unintuitive and do not force participants to directly compare test cases. Ordinal ranking asks participants to directly rank entries while comparing all at once. This provides a simple interface and is faster still than categorical judgement as participants need only decide whether a test case is preferred over its neighbours. However, in enforcing a hierarchic order on test cases, ranking order may introduce bias as participants are asked to distinguish between cases that may be equally preferred [Gui54]. The accuracy of these three methods and the time required per participant for psychophysical measurement is such that *paired comparison* > *categorical judgement* > *ordinal ranking*.

We wish to discover preference and to be able to both quantify by how much one transition is preferred over another and to state whether this amount is significant. As preference is not intuitively

quantifiable, we can transform the ordinal data produced by our three considered methods into a uniform interval scale for preference. Multi-dimensional scaling [Tor58] can be applied to all methodologies. This takes similarities from an  $n$ -by- $n$  matrix of test cases and places each case into an  $N$ -dimensional space to quantify the differences. In our case,  $N$  is one, as we wish to place ranked scores into a preference scale to assess whether the difference in preference between transitions is significant. If the resulting scale difference is significant for certain scenes, for example scenes with significant camera shake, then it is a strong indication that a transition is preferred under those conditions.

We choose to progress with an ordinal ranking experiment. Even though this raw ranking is not as accurate as other methods, it is the fastest, which allows us to compare more scenes and transitions in a set amount of time. This method also restricts a participant's choice to label test cases as equally preferred. However, the visual differences between our test cases should be noticeable as our transition techniques produce quite different results with different types of artefacts (see Section A).

In a ranking experiment, there is a trade-off between video size in pixels and layout of the experiment. The videos must be large enough to see scene detail and artefacts, but ideally the interface would allow participants the freedom to rank how they wished. Our balance in the trade-off was that, for a typical 1368x768 display [SS11], two 16:9 videos must fit vertically on the display. As each video has a border, label and spacing which adds extra vertical pixels, the final video size was 640x360. Our source videos and transition renderings filled a 1920x1080 frame, so a significant amount of detail was lost in both the feature of the transition and the artefacts. For three videos of this size to be visible the display would have to be tilted vertically. A display of all videos at this size would squeeze into a 2560x1600 display oriented vertically. However, we wish to use a Web-based interface to quickly and easily canvas many participants, and so this is a much less practical approach as few people have such displays. Our interface can be seen in Appendix C.

### 6.2.1 Transition Choice

In our experiment, we wish to include transitions commonly used in both movies and graphics-rendered applications: any example, interactive or otherwise, in which media is digitally transitioned from one image to another. From movies, we include cut and dissolve transitions. Cut transitions form a baseline as a cut is the simplest way to join two clips. We include dissolve transitions, which commonly represent the passage of time, as there are time differences between clips in our video collections. Other transitions, such as wipes and reveals, are less common; we do not include them as they add nothing over a cut or dissolve to help maintain the space of action.

Hidden cuts are carefully planned transitions which perform very similar motions in different clips over scene regions with matching image content. For example, a pan over a brick wall to end one video is matched with a pan over the same brick wall to start another video. This leads to the appearance of a single video. In contrast to wipes and reveals, hidden cuts do help maintain the space of action — this is exactly their goal. Specific visually-matching footage may exist in a spatially dense video collection, but it is likely to be rare in our spatially sparse video collections.

With computer graphics, the hidden cut can be replaced with a seamless computer generated tran-

sition which relies on scene geometry and a virtual camera. We call these *full 3D dynamic* transitions as they require full scene geometry (or in part a suitable geometry proxy) and they maintain the motion of dynamic objects through the transition. We include the *full 3D dynamic* transition in our study as it maintains the space of action and sense of orientation in the viewer by rendering a perspective-correct view from virtual cameras that join both clips. This transition also maintains as much as possible the motion of dynamic objects by projecting playing video clips onto the scene geometry.

We also wish to test seamless spatial transitions which do not maintain the motion of dynamic objects during the transition. In *full 3D static* transitions we render a virtual view using scene geometry as before, but do not keep playing the video as the virtual camera moves. During the transition the world appears as if time has stopped, similar to time-slice photography [Mac80, Deb81]. Unlike *full 3D dynamic* transitions, the motions in the clips do not blend into each other. All motions in the start clip freeze as the clip is paused. The virtual camera then moves by rendered views to the camera position in the first frame of the end clip, and finally the end clip starts playing. The inclusion of this transition will test under which conditions it is important to maintain dynamic object motions.

Both *full 3D* transitions use accurate scene geometry, but many existing applications employ simple proxy geometry, such as a single plane, to represent scenes [MCG05, SSS06, VCL<sup>+</sup>11]. Such *plane* transitions work well for camera rotations, but suffer artefacts if the start and end clips are shot from different positions. This transition type is currently popular among commercial touring and mapping applications [Mic08, Goo08], and is a baseline as the simplest registered graphics-rendered approach.

If partial scene geometry is available, *ambient point clouds* (APC) can help fill in gaps in geometry as an alternative to partial planar proxies. Goesele et al. introduced these transitions in 2010 [GAF<sup>+</sup>10] to provide visual hints at motion and depth. We include APC transitions as they represent the state of the art in automatic graphics-rendered transitions from community photo collections, where it is often the case that only incomplete geometry is recoverable or available.

Video morph or *warp* transitions are often used as a special effect in movies to transform one object into another, but recent advances in robust feature point correspondence ([Low04, LLN<sup>+</sup>10]) have allowed view change transitions as well. Warp transitions provide an alternative to both transitions that require geometry and to plane transitions: while plane transitions can be classified as a subset of warps with global 2D transformations (4-point correspondence), warps can also be computed with many hundreds or thousands of point correspondences. We include these many-correspondence local warps in our comparison as they maintain the space of action and are visually different from other transitions.

We exclude other transition types which currently require manual work because we need to produce many hundreds or thousands of transitions across our video collection. This includes any transition type which requires interactive segmentation [HAA97, MO09, CSDI11]. We also exclude any transitions which rely on geometry that cannot be reconstructed from the video collection itself. This includes transitions based on laser-scans [MO09, Goo07] and hand-modelled geometry [DYB98, OCDD01, Eve09], though technically these are all full 3D transitions with varying geometry fidelity.

## Transition Types in Detail

Each of the transition types is fully described in Appendix A rather than here because the descriptions are long and would otherwise interrupt the experiment description. For each transition, the description contains a historical review of application, our technical method to achieve the transition, and an explanation of artefacts that may appear. Following these transition descriptions, a further subsection explains techniques and issues in common between transitions, including how 3D scene geometry is recovered and how the Videoscape graph aids in this reconstruction. However, for referencing in this section, we collate and categorize all feature and artefact types in each transition in Table 6.1. We will use this table to cross-reference comments from participants in the user study in Section 6.3.4.

### 6.2.2 Clip Choice

Transition preference must be tested across different clips containing different scenes, as transition preference may vary between scenes and scene elements. Beyond this, we consider that transition preference may vary based on the view change between the start and end video clips in a transition. For instance, two clips with no camera motion shot from the same position would transition with less ghosting in a dissolve than clips shot from different positions. Likewise, a full 3D transition adds very little to two clips shot from the same position and may introduce artefacts from missing geometry, but provides a smooth virtual view transition that respects parallax to clips shot from different positions.

To include view change as a variable in our experiment, we select portals that contain both a slight view change and a considerable view change. A slight view change is a transition from one video clip to another where the visual elements in the scene, such as the buildings and people, approximately maintain their size and position. These slight view changes can still allow considerably different camera positions as we allow zoom to vary, but all camera positions in our chosen scenes lie within a  $10^\circ$  cone ( $9^\circ$  average) with its apex approximately at the depth of the scene in the middle of the video frame (Figure 6.1).

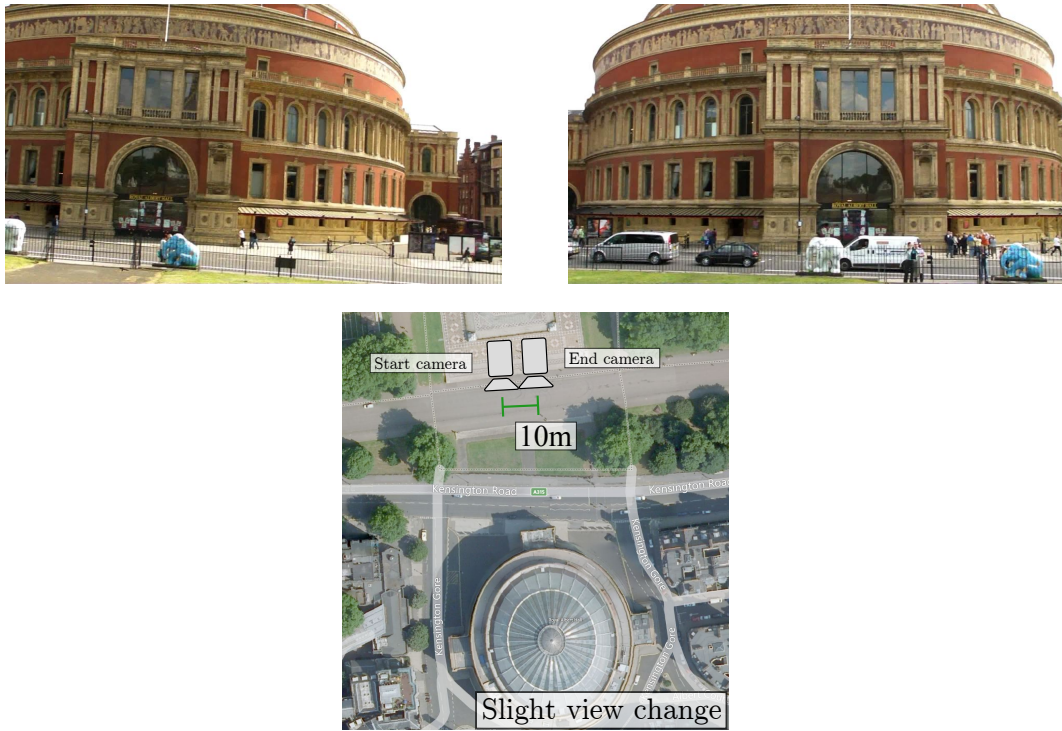
A considerable view change transition occurs between clips with camera positions outside a  $10^\circ$  cone, and in our chosen scenes this is maximally  $55^\circ$  ( $34^\circ$  average, Figure 6.2). While our full 3D transitions would be able to produce good virtual views for larger view changes, the portal selection stage (Section 5.5) explicitly rejects these cases as there is insufficient visual similarity in the start and end clips to maintain the two-dimensional space of action.

From our London database, we select by hand five portals each with two view changes. These portals were chosen as representative of the clips in the database, and contain buildings in the middle distance (approximately 50-300 metres) along with smaller dynamic objects such as birds, boats, cars, and pedestrians. For each portal we choose one clip as the reference end clip. Then, we choose two start clips: one each for slight and considerable view changes. The portal frame at which the start clip best matches the end clip is fixed, although for implementation reasons the transition may start before this portal frame (see Section A.7.5). Some clips contain camera shake - if this is the end clip then the shake is present in both slight and considerable view changes. In Scene 4, the shake is so significant to cause rolling shutter artefacts. Table 6.2 and Figure 6.3 describe and show the five scenes.

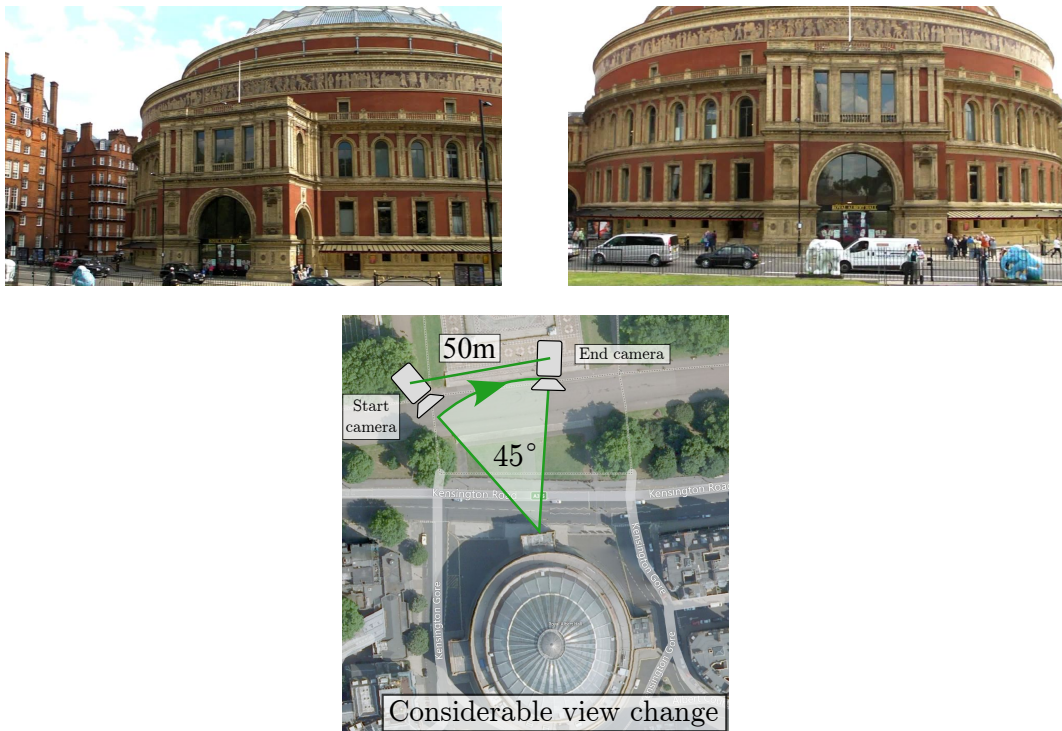
	Cut	Dissolve	Warp	Plane	APC	Full3DDyn	Full3DSta
<i>Feature</i>							
Registered scene			•	•	•	•	•
3D effect			○ <sup>1</sup>		•	•	•
Dynamic objects		•	•	•	•	•	
Smooth virtual camera (A.7.3)			○ <sup>2</sup>	•	•	•	•
Common familiarity	•	•					
Signifies change of time		•					
Explicit motion cues					•		
Frozen time							•
<i>Artefact</i>							
Ghosting (static objects)		•		• <sup>3</sup>			
Ghosting (dynamic objects)		•	•	•	•	•	
Orientation loss (A.1)	•	•					
Bad corresp. swirls (A.3)			•				
Frame edge flickering (A.3)			•				
Skewed scene (A.4)				•		○ <sup>4</sup>	○ <sup>4</sup>
Temporal pepper noise (A.5)					•		
Multiple scene elements (A.6)					○ <sup>5</sup>	•	○ <sup>6</sup>
Recovered geom. failures (A.7.1)					•	•	•
Empty black regions (A.7.4)				•	○ <sup>7</sup>	○ <sup>8</sup>	○ <sup>8</sup>

**Table 6.1:** A table collating all features and artefacts for each transition type. Section numbers for text explaining each feature or artefact are included in parentheses. 1: Partial, only with good regular correspondence and flow correction. 2: Velocities only from feature-point tracks. 3: Although the scene is sparsely registered, ghosting is still present in almost all transitions because the plane is an inaccurate proxy to the true geometry. 4: On proxy planes only. 5: An image is formed within the APC as it appears as a noisy plane during slight view changes only. 6: Not as prominent as full 3D case as global registration at portal frames is better aligned to geometry, but still possible. 7: APC reduces, but not maximally, empty regions; introduces pepper noise. 8: Minimized as much as possible given video-video-geometry registration.





**Figure 6.1:** An example of a slight view change from Set 7 in the experiment from the Albert Hall scene. Approximately, the camera translates 10 metres and undergoes a  $5^\circ$  roll rotation (roll not shown on diagram). Map image courtesy of Google.



**Figure 6.2:** An example of a considerable view change from Set 8 in the experiment from the Albert Hall scene. Approximately, the camera translates 50 metres and undergoes a  $45^\circ$  yaw rotation. Map image courtesy of Google.



Scene	Name	Set	S/C	Scene Description	Special Features
1	County Hall	1	S	View change along a bridge over a river into pan left, observing Edwardian Baroque County Hall on the bankside.	Camera shake in start video. Bird in flight in foreground, people in foreground to left.
		2	C	Pan left over a river from bank-side changes to pan left from bridge observing County Hall.	Travelling boats on river, people in foreground to left.
2	Palace of Westminster	3	S	Middle distance shot of Neo-Gothic Palace changes to farther distance shot in pan right.	Road traffic and pedestrians at bottom of frame.
		4	C	Palace in far shot in pan right changes to middle distance shot in pan right.	Many flying birds, people, travelling boats, distant road traffic and lamppost occluder.
3	Victoria Embankment	5	S	Pan left on bridge over river across Neo-Gothic buildings changes to pan right with bridge road in foreground.	Flying bird and pedestrians in foreground, distant road traffic and flags.
		6	C	Pan right across bankside changes to view of bridge surface in pan right.	Many flying birds, people, travelling boats, distant road traffic and lamppost occluder.
4	Royal Albert Hall	7	S	Translate right changes to translate left with full frame Neo-Romanesque building.	Significant camera shake, rolling shutter artefacts, road traffic in middle distance.
		8	C	Pan left changes to translation left with full frame building.	Significant shake in end video, rolling shutter artefacts, road traffic at frame bottom.
5	Millennium Bridge	9	S	Translate forward along modern glass/steel bridge changes to translate plus pan left.	Camera shake and many people in foreground.
		10	C	Pan left and zoom from bank-side changes to pan left upon suspension bridge.	Camera shake, travelling boat in middle distance and people in foreground.

**Table 6.2:** All scenes breakdown with set number, common names, slight or considerable view change (S or C), contents and special features identified.



(a) Scene 1: County Hall, sets 1 &amp; 2.



(b) Scene 2: Palace of Westminster, sets 3 &amp; 4.



(c) Scene 3: Victoria Embankment, sets 5 &amp; 6.



(d) Scene 4: Royal Albert Hall, sets 7 &amp; 8.



(e) Scene 5: Millennium Bridge, sets 9 &amp; 10.

**Figure 6.3:** All scene slight and considerable view changes. Left: Start video frame for slight view change transition. Middle: Start video frame for considerable view change transition. Right: End video frame for both slight and considerable view change transitions. That is, the slight view change transition moves from the left column to the right column, and the considerable view change transition moves from the middle column to the right column in each case.

## 6.3 Experiment: Towards Automatic Transition Type Choice

### 6.3.1 Hypothesis

Only certain video-to-video transition types are appropriate for certain scenes.

Of course, the scope of possible scenes is vast and so any findings must be taken in context. Our goal is to derive criteria to automatically choose the most appropriate transition type for a given portal. We expect warps and blends to be better when the view change is slight, and transitions relying on 3D geometry to be better when the view change is considerable. We anticipate that transitions relying on 3D scene reconstruction will only be visually pleasing if the reconstructed geometry is good.

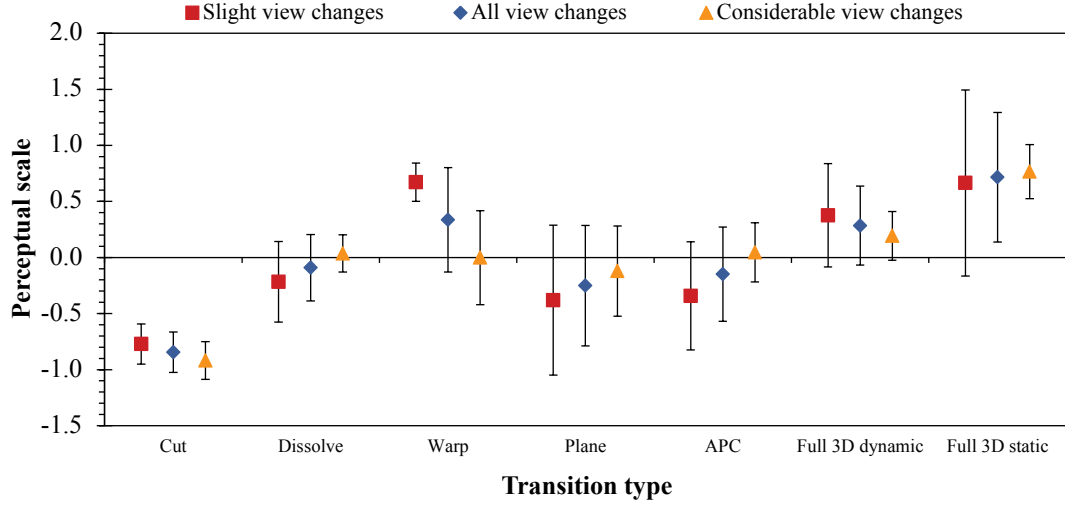
### 6.3.2 Method

We proceed with a ranking experiment followed by multi-dimensional scaling to calculate whether a significant preference difference exists between transition types. Participants will view seven transition types (Section A): *cut*, *dissolve*, *warp*, *plane*, *ambient point cloud*, *full 3D static* and *full 3D dynamic*. Participants will view and rank these transitions across five different scenes. The five scenes were chosen as they each display a potentially difficult situation: Scene 1: dynamic objects at boundary; Scene 2: many dynamic objects with view occlusions and panning camera; Scene 3: panning cameras and dynamic objects; Scene 4: fast moving dynamic objects and shaking camera/rolling shutter; Scene 5: complicated foreground objects and moving, shaking camera.

For each scene, one transition forms a slight view change ( $10^\circ$  maximally,  $9^\circ$  average, but with zoom changes) and one transition forms a considerable view change ( $55^\circ$  maximally,  $34^\circ$  average, also with zoom changes). Our portal selection method tries to avoid portals with very large view changes, and for these chosen scenes the considerable view change cases represented rotations up to  $55^\circ$ , though larger changes are possible if no better match exists between two videos. While our slight view changes always have only small rotations around the scene ( $10^\circ$  average), this does not mean that the camera position does not vary much, e.g., in Scene 5, the virtual camera moves two hundred meters because one clip is zoomed out and one clip is zoomed in. The target video is always the same for both view changes. Each transition will consist of two seconds of video, plus one second of transition where necessary, followed again by two seconds of video. The experiment should take approximately one hour per participant.

Our pilot study involved two participants and mostly generated feedback on difficulties and errors in the user interface. Comments from the pilot study, such as “rank label (position) next to videos”, “increase [video] label size”, and “very hard to verify that videos have been sorted”, were considered and improvements were made to the participant website interface. Some comments could not be integrated: “scroll [page] automatically when dragging video”. Unfortunately, this is behaviour defined by the browser and so could not easily be changed. The experiment website is shown in Figures C.1 and C.2.

We began the experiment proper. Participants ranked the seven video transition types for each of the ten portals. First, the scenario of navigating a video collection is explained to the participant and two example transitions are shown. The participant is then presented with each set of video transitions in a random order. Each transition type is randomly placed into a vertical video list. Participants drag



**Figure 6.4:** Mean and standard deviation plotted on a perceptual scale for the different transition types across all scenes. Perceptual scales are in z-score order, with the group mean at 0 and y-value representing multiples of the group standard deviation.

Significance	Cut	Dissolve	Warp	Plane	APC	Full 3D dyn	Full 3D sta
Cut		$2.65 \times 10^{-5}$	$2.89 \times 10^{-5}$	$1.34 \times 10^{-2}$	$3.38 \times 10^{-4}$	$7.17 \times 10^{-6}$	$5.84 \times 10^{-5}$
Dissolve	$2.65 \times 10^{-5}$		$5.57 \times 10^{-2}$	$4.31 \times 10^{-1}$	$5.61 \times 10^{-1}$	$6.49 \times 10^{-2}$	$9.89 \times 10^{-3}$
Warp	$2.89 \times 10^{-5}$	$5.57 \times 10^{-2}$		$2.26 \times 10^{-2}$	$8.20 \times 10^{-2}$	$8.11 \times 10^{-1}$	$1.82 \times 10^{-1}$
Plane	$1.34 \times 10^{-2}$	$4.31 \times 10^{-1}$	$2.26 \times 10^{-2}$		$6.79 \times 10^{-1}$	$7.42 \times 10^{-2}$	$7.46 \times 10^{-3}$
APC	$3.38 \times 10^{-4}$	$5.61 \times 10^{-1}$	$8.20 \times 10^{-2}$	$6.79 \times 10^{-1}$		$3.16 \times 10^{-2}$	$1.23 \times 10^{-2}$
Full 3D dyn	$7.17 \times 10^{-6}$	$6.49 \times 10^{-2}$	$8.11 \times 10^{-1}$	$7.42 \times 10^{-2}$	$3.16 \times 10^{-2}$		$2.51 \times 10^{-2}$
Full 3D sta	$5.84 \times 10^{-5}$	$9.89 \times 10^{-3}$	$1.82 \times 10^{-1}$	$7.46 \times 10^{-3}$	$1.23 \times 10^{-2}$	$2.51 \times 10^{-2}$	

**Table 6.3:** Student's t-test matrix for significance of preference, with  $p - value < 0.05$ . Green cells denote significantly preferred, and red cells denote significantly less preferred. The table should be read as follows: Column 'Cut' with row 'APC' is red, which equals that Cut is significantly less preferred than APC. Column 'APC' with row 'Cut' is green, which equals that APC is significantly preferred than Cut.

and drop videos to reorders the list from most preferred to least preferred. Each of the videos can be played any number of times. Of the 21 participants in our experiment, 12 were self-described experts with experience in graphics and media production, 4 were amateurs, and 5 were novices. On average, it took 52 minutes to complete the study. Participants could optionally provide comments in text.

### 6.3.3 Results

We perform multi-dimension scaling [Tor58] to place our transition type ordinal rankings onto an interval scale. Figure 6.4 shows the mean and standard deviation across all scenes and view changes, with Table 6.3 showing significance values and whether these cross a positive/negative threshold of  $p - value < 0.05$ . Perceptual scores for all scenes and conditions are summarized in Table 6.4.

The slight view change condition perceptual scale is also shown in Figure 6.4, with significances noted in Table 6.5. The considerable view change condition perceptual scale is again shown in Figure

Transition	Scene 1		Scene 2		Scene 3		Scene 4		Scene 5		Overall
	S	C	S	C	S	C	S	C	S	C	
Cut	-1.06	-0.75	-0.61	-1.10	-0.72	-0.84	-0.65	-0.81	-0.82	-1.10	-0.84
Dissolve	-0.81	0.00	-0.24	0.09	0.12	0.30	0.01	-0.11	-0.18	-0.09	-0.09
Warp	0.50	-0.39	0.67	0.09	0.50	0.08	0.87	-0.40	0.82	0.61	0.33
Plane	-0.72	-0.25	-0.42	0.12	-1.23	-0.74	-0.08	-0.05	0.54	0.31	-0.25
APC	-0.95	0.19	0.02	-0.29	0.22	0.29	-0.68	0.22	-0.33	-0.19	-0.15
Full 3D dynamic	0.93	0.32	0.41	-0.03	0.72	0.22	-0.09	0.47	-0.08	-0.02	0.28
Full 3D static	2.10	0.87	0.16	1.12	0.40	0.69	0.61	0.68	0.05	0.48	0.72

**Table 6.4:** Perceptual scaling values for transition types across video sets. ‘S’ and ‘C’ denote slight and considerable view changes.

Significance	Cut	Dissolve	Warp	Plane	APC	Full 3D dyn	Full 3D sta
Cut		$6.58 \times 10^{-3}$	$6.13 \times 10^{-5}$	$2.66 \times 10^{-1}$	$7.16 \times 10^{-2}$	$1.11 \times 10^{-2}$	$3.09 \times 10^{-2}$
Dissolve	$6.58 \times 10^{-3}$		$4.01 \times 10^{-3}$	$6.55 \times 10^{-1}$	$4.84 \times 10^{-1}$	$1.35 \times 10^{-1}$	$1.58 \times 10^{-1}$
Warp	$6.13 \times 10^{-5}$	$4.01 \times 10^{-3}$		$1.08 \times 10^{-2}$	$1.36 \times 10^{-2}$	$3.57 \times 10^{-1}$	$9.86 \times 10^{-1}$
Plane	$2.66 \times 10^{-1}$	$6.55 \times 10^{-1}$	$1.08 \times 10^{-2}$		$9.32 \times 10^{-1}$	$1.92 \times 10^{-1}$	$1.33 \times 10^{-1}$
APC	$7.16 \times 10^{-2}$	$4.84 \times 10^{-1}$	$1.36 \times 10^{-2}$	$9.32 \times 10^{-1}$		$7.06 \times 10^{-2}$	$1.42 \times 10^{-1}$
Full 3D dyn	$1.11 \times 10^{-2}$	$1.35 \times 10^{-1}$	$3.57 \times 10^{-1}$	$1.92 \times 10^{-1}$	$7.06 \times 10^{-2}$		$3.73 \times 10^{-1}$
Full 3D sta	$3.09 \times 10^{-2}$	$1.58 \times 10^{-1}$	$9.86 \times 10^{-1}$	$1.33 \times 10^{-1}$	$1.42 \times 10^{-1}$	$3.73 \times 10^{-1}$	

**Table 6.5:** Slight view change sets student’s *t*-test matrix for significance of preference, with  $p$  – value  $< 0.05$ . The table should be read as in Table 6.3.

6.4, with significance noted in Table 6.6.

Comments from the experiment provided by participants are summarized in Tables 6.7 and 6.8. The first table collates comments by transition, noting positive and negative feedback; the second table collates comments by feature/artefact as in Table 6.1. These points will be discussed in Section 6.3.4.

### 6.3.4 Discussion

The results show that there is an overall preference for full 3D static transitions (Figure 6.4). This is not surprising as the video frames are projected onto actual 3D geometry and this provides the strongest spatial cues of all transitions. From the comments of participants (Table 6.8), we also know that the 3D transitions not only have smooth camera motion but also provide the effect of being spatially immersed in the scene — these features were positively noted most often when compared to other features. Surprisingly, full 3D dynamic transitions where both videos continued playing were preferred less, and this is also reflected in the comments as frozen time was positively noted more often than the presence of dynamic objects. Looking at the per-scene results, we hypothesize that this is due to ghosting which stems from inaccurate camera tracks in the difficult shaky cases.

The warp is preferred for slight view changes, and is significantly better than non-full 3D transitions when considering slight view changes only ( $p$ -value  $< 0.05$ , *t*-test, Table 6.5). While it is not signif-

Significance	Cut	Dissolve	Warp	Plane	APC	Full 3D dyn	Full 3D sta
Cut		$6.60 \times 10^{-4}$	$2.19 \times 10^{-2}$	$2.84 \times 10^{-2}$	$5.94 \times 10^{-5}$	$1.22 \times 10^{-5}$	$2.34 \times 10^{-4}$
Dissolve	$6.60 \times 10^{-4}$		$8.48 \times 10^{-1}$	$5.46 \times 10^{-1}$	$9.63 \times 10^{-1}$	$3.07 \times 10^{-1}$	$2.94 \times 10^{-3}$
Warp	$2.19 \times 10^{-2}$	$8.48 \times 10^{-1}$		$5.88 \times 10^{-1}$	$8.76 \times 10^{-1}$	$5.18 \times 10^{-1}$	$3.65 \times 10^{-2}$
Plane	$2.84 \times 10^{-2}$	$5.46 \times 10^{-1}$	$5.88 \times 10^{-1}$		$5.90 \times 10^{-1}$	$2.61 \times 10^{-1}$	$1.38 \times 10^{-2}$
APC	$5.94 \times 10^{-5}$	$9.63 \times 10^{-1}$	$8.76 \times 10^{-1}$	$5.90 \times 10^{-1}$		$6.83 \times 10^{-2}$	$1.59 \times 10^{-2}$
Full 3D dyn	$1.22 \times 10^{-5}$	$3.07 \times 10^{-1}$	$5.18 \times 10^{-1}$	$2.61 \times 10^{-1}$	$6.83 \times 10^{-2}$		$2.06 \times 10^{-2}$
Full 3D sta	$2.34 \times 10^{-4}$	$2.94 \times 10^{-3}$	$3.65 \times 10^{-2}$	$1.38 \times 10^{-2}$	$1.59 \times 10^{-2}$	$2.06 \times 10^{-2}$	

**Table 6.6:** Considerable view change sets student’s *t*-test matrix for significance of preference, with *p* – value < 0.05. The table should be read as in Table 6.3.

Transition	# Positive	# Negative	Difference
Cut	6 <sup>1</sup>	2	+4
Dissolve	5 <sup>1</sup>	1	+4
Warp	3	0	+3
Plane	2	1	+1
Ambient Point Clouds	6	6	0
Full 3D Dynamic	5	3	+2
Full 3D Static	20	2	+18

**Table 6.7:** Numbers of positive and negative comments from participants for each transition type. 1: All of these positive comments were left when artefacts in other transitions became overwhelming, making the fall-back transitions preferred.

# Liked (# from unique participants)		
<i>Feature</i>		
Registered scene	0	(0)
3D effect	6	(5)
Dynamic objects	1	(1)
Smooth virtual camera (A.7.3)	5	(5)
Common familiarity	1	(1)
Signifies change of time	0	(0)
Explicit motion cues	2	(2)
Frozen time	3	(2)
# Disliked (# from unique participants)		
<i>Artefact</i>		
Ghosting (static objects)	6	(3)
Ghosting (dynamic objects)	3	(3)
Orientation loss (A.1)	3	(3)
Bad corresp. swirls (A.3)	0	(0)
Frame edge flickering (A.3)	0	(0)
Skewed scene (A.4)	1	(1)
Temporal pepper noise (A.5)	3	(2)
Multiple scene elements (A.6)	2 <sup>1</sup>	(1)
Recovered geom. failures (A.7.1)	4	(1)
Empty black regions (A.7.4)	4	(3)

**Table 6.8:** A table collating all comments related to specific features and artefacts identified in Table 6.1. Section numbers for text explaining each feature or artefact are included in parentheses. 1: Multiple scene elements were not explicitly mentioned, but participants did comment on Sets 9 and 10 (in which multiple scene elements are present) that the geometry was incorrect. Thus, 2 of the comments from ‘Recovered geom. failures’ have been additionally attributed to Multiple scene elements.



icantly preferred over full 3D transitions, opinion on the warp transition in slight cases was consistent, with a very small variance and the highest mean score of any transition (Figure 6.4). The static 3D transition is among the top 3 transitions for all sets, and overall is significantly better than all other transitions for considerable view changes ( $p\text{-value} < 0.05$ ,  $t\text{-test}$ , Table 6.6). This justifies the computational cost of reconstructing and rendering such a transition. In almost all cases, cut transitions were significantly unpreferred — only plane and APC slight view cases were not significantly unpreferred over a cut.

Beyond these results, it is hard to make strong statements with statistical significance about our scenes and transition types. We did not rigorously test for specific features and artefacts, or test for specific scene objects and effects — this would be a much larger experiment and is beyond the scope of this thesis. However, it is still worthwhile to discuss these issues based on per-set and per-transition results and on user comments to deduce as much as possible about transition preference. Thus, the following discussion should be taken as having only anecdotal evidence to support statements.

### Participant Comments

Of the 22 participants, 13 left comments. Approximately 78 comments were left in total; some interpretation is necessary to separate compound block comments. Table 6.7 collates these comments into per-transition positive and negative comments. The collation rules are as follows:

- To be counted, comments must refer to specific positive or negative features of the transition (“pepper noise is bad”).
- Any comment that specifically mentions a transition by name is included regardless of where a participant ranked that transition (“APC is good”).
- Comments about specific features of a transition that do not mention the transition explicitly by name are included:
  - As positive comments for transitions ranked 1st or 2nd.
  - As negative comments for transitions ranked 6th and 7th.

For instance, the hypothetical comment “pepper noise is bad” would only score as a negative comment for APC if the participant also ranked the APC transition in 6th or 7th. This general comment is unquantified; pairing it with the rank position gives us some confidence that, in this case, the pepper noise was a major factor in the low ranking rather than just a minor factor. Additionally, the fact that a transition is ranked 1st/2nd or 6th/7th and a comment has been made does not include it in this compilation. The participant must have made a specific positive or negative comment to provide justification as to why the transition was ranked 1st/2nd or 6th/7th.

It is very clear that people overwhelmingly preferred to comment on the full 3D static transition, with comments such as “my favourite by far due to [the] subtle change and pleasant feeling” (participant 5), “is really good” (participant 12), and “looks great” (participant 17). Negative comments were left for full 3D static transitions when the recovered geometry was inaccurate or incomplete or both. Cut and dissolve transitions appear to have comparatively more positive comments than the perceptual score



might suggest. On closer inspection, this is because people positively commented upon cut and dissolve transitions in cases where geometry reconstruction was poor: “Some of the middle-ranked vids had a vague object in the middle of the frame during the transition. So they ranked worse than flick/fade” (participant 22).

APC appears most divisive with equal positive and negative comments. Three of the positive comments were similar to the cut/dissolve positive comments in that they applied in scenes only where geometry reconstruction was poor. However, some comments appreciated the motion cues in some sets: “Although sometimes the point render helped identify the geometry that was being matched. It wasn’t pretty and I wouldn’t choose it in a presentation, but it was probably more useful” (participant 20). The negative comments mostly related to artefacts: “pixelated and empty spaces are bad” (participant 05), “pretty jarring” (participant 20), and “looked like the film had been spoiled” (participant 21).

Table 6.8 collates comments on specific features and artefacts as in Table 6.1. It is clear that 3D geometry affects transition preference greatly: participants commented positively most frequently on the 3D effect and smooth virtual camera that comes with these transitions. Comments on artefacts are more difficult to assess, as many participants repeated their comments on artefacts on multiple occasions. Still, many artefacts caused by poor geometry or registration of all kinds were noted, and these mostly concerned static ghosting and geometry failures.

### Methodology and Interface

Of all 22 participants, 7 left a total of 10 comments about the difficulty in ranking transitions. These comments referred to 6 different sets, with each set gaining no more than 2 comments. From this we suggest that across participants and sets there was no strong agreement about what was difficult to rank. However, we can say that generally some participants felt it difficult to rank at least some transitions, and this may not have occurred had we chosen a different methodology, such as paired comparison, that allowed transitions to be ranked as equal.

The first consequence of the trade-off between video size and screen resolution (Section 6.2) is that most people could see only two videos at a time, and so only insertion and bubble sort ranking strategies were possible. From the comments, a bubble sort strategy was more common. The second consequence is that it was very difficult to directly compare three or more transitions. The difficulty in differentiating between transitions was commented on by 7 participants a total of 10 times, and here it may have helped to allow more transition videos on screen at once. Likewise, it may have helped to allow participants to trigger transitions to play at exactly the same time, rather than having to manually press the play button on both videos.

### 6.3.5 Individual Transition Types

#### Cut

These transitions are strongly disliked under both slight and considerable view changes, and our experimental findings suggests that cuts are not appropriate for our system. This is expected as cuts provide no additional cues to aid the orientation of the viewer. From the 30° rule, we might expect cuts to be pre-

ferred more in considerable view changes than slight view changes. However, for these scenes our data suggests otherwise: the cut transition was unpreferred against fewer transitions in the slight case (with plane and APC having insignificant differences, Table 6.5), and the absolute difference in preference between the nearest transitions is greater in the considerable case (Figure 6.4, 0.16 for slight vs. 0.80 for considerable). Further observation reveals that in the plane and APC cases, we can see that this difference in significance comes from the variance due to artefacts which appear more objectionable in certain scenes, specifically sets 5 and 6 for the plane transition (skewed scenes and large empty areas) and set 7 for the APC transition (pepper noise).

Some of our participants commented that cuts were preferred in cases where geometry recovery and/or video registration was poor and resulted in many artefacts. These cases are covered in sets 9 and 10. However, over all participants the per-set results do not support these comments: in sets 9 and 10, cut transitions are still the least preferred transitions by some margin (0.49 and 0.91 scale difference respectively to the next preferred transition). One participant consistently ranked cut transitions as most preferred. Upon further questioning, the participant explained that they had no tolerance for any kind of double image whatsoever, and always preferred the sharpest image.

## Dissolve

In all sets and both view change conditions, dissolve transitions sit largely as a middle tier transition, neither significantly unpreferred nor preferred against most other transitions. Dissolve transitions are always significantly preferred over cut transitions, so this forms a better baseline than a cut in cases a) where correspondence is very hard to achieve or when geometry reconstruction fails, such as for highly reflective buildings which are constructed with lots of glass, and b) where no additional processing should be undertaken, for instance, on compute constrained platforms. In the slight view case, warps are significantly preferred over dissolves; in the considerable view case, full 3D static transitions are significantly preferred over dissolves. These two results fit well with our expectations.

Sets 4, 6 and 7 present interesting cases for the dissolve transition as it ranks comparatively highly (second or third, Figures B.5, B.7 and B.8). Set 4 transitions between slowly panning videos, but also undergoes a significant zoom: the building in the start clip is in the distance, but is much closer in the end clip. This scale change masks the double image that would normally appear if both clips were similar distances from the building. The scene context just happens to presents a higher quality dissolve than is typical in our system. The dissolve for set 6 is interesting because it somewhat represents a hidden cut. Here, both start and end clips are panning at similar velocities, and one particular building is in a similar position in the frame in both videos. As the videos dissolve, this building catches the eye and becomes an anchor because it is roughly registered relative to the rest of the frame. In this way, the part of the frame that this building occupies undergoes a crude and accidental hidden cut. Finally, set 7 is high ranking for the dissolve transition because it involves considerable camera shake and other transitions contain considerable artefacts.

## Warp

The warp transition is the only transition across all sets for which the full 3D static transition is not significantly preferred. Why this is becomes clear when looking at the individual case results: in the considerable view change case, the warp is perceptually similar to all but the cut and full 3D static transitions. However, in the slight view case, the warp is the only transition to be significantly preferred if we ignore the cut. While it is still not significantly preferred against the two 3D transitions, it has the highest perceptual score and a much smaller variance (Figure 6.4). This result is unsurprising as an image-based technique should perform better when the transition start and end frames visually share more in common.

More surprising was that nobody commented on any warp-specific artefacts such as swirling and frame edge flickering (Table 6.8). While we cannot say why, it might be that these image-based artefacts are less objectionable than geometry-based artefacts such as double images. Alternatively, they may simply be less noticeable at the edges of the frame.

## Plane

The plane transition sits in the middle tier of preference. Warp and full 3D static transitions are significantly preferred over the plane transition across all sets. As expected, the warp transition is significantly preferred over the plane transition in the slight view change case, and the full 3D static transition is preferred over the plane transition in the considerable view change case. Typical skew artefacts are mentioned only once explicitly in the comments.

Sets 5 and 6 are particularly bad for the plane transition, with set 5 seeing the plane transition with the lowest perceptual score of all transitions for this set. These two sets cause considerable scene skewing, more than any other set, due to the variation in depth in the scene making a plane a particularly bad proxy. Given this, these results are not surprising.

## Ambient Point Clouds

The ‘noisy’ transition, as one participant labelled it, sits similarly in the middle tier of preference. Across all sets, both full 3D transitions are significantly preferred over APC. In the slight view change case, warp is preferred over APC; in the considerable view change case, full 3D static is significantly preferred over APC. We may have expected APC to perform better in the considerable view change case as the motion cues are stronger in these cases of more difficult orientation, but this is not the case. Another reason why this was the expected result is that APC tends to add pepper noise artefacts in slight cases without being able to show the real benefit of streaking motion cues. Regardless of this result, we suggest that more points may be needed to fill in empty noise regions and create more coherent streaks, though this comes at an added memory and rendering cost (see Section A.5).

From the comments, participants complained about gaps in the rendering manifesting as pepper noise and as black regions created by not generating an APC for each video frame. Strangely, APC did generate positive comments in set 2, with participants commenting on “the smoothness and dynamism and fluidity of the movement” (participant 22; an example from this transition is shown on the right in Figure A.11). This case is actually a failure for APC: the generated result is incorrect, as discussed

in Section A.5. However, here the APC transition appears faster than other transitions with smooth camera motion due to the motion of the APC: the APC appears to be behind the geometry, and the strong motion cues created by the explicitly incorrect introduced black borders have the effect of speeding up the appearance of the camera motion.

### Full 3D Dynamic

Our most surprising result was with full 3D transitions. We expected these to score very highly, but across all sets they were only significantly preferred over APC and cut transitions. In the slight view change case, while the perceptual mean seems much higher, in fact there is no significant preference over any other transitions other than cuts. In the considerable view change case, full 3D dynamic transitions have similar perceptual scores to dissolve, warp, plane and APC transitions, and again are only significantly preferred over cuts. The mediocrity is reflected in the comments, with very few specific mentions — only one participant noted liking that objects still moved during transitions.

Further, we did not anticipate two additional artefacts that the full 3D static transition does not suffer: added per-frame ghosting on static scenes, and extra empty areas. First, the added ghosting comes from minor inaccuracies in the video registration, but also from geometric errors that are not revealed when viewed from the portal frames (at which the full 3D static transition starts and ends). The portal frames are used in geometry reconstruction, but neighbouring frames in the video which have undergone parallax are not, and so the geometry is not photometrically consistent with these neighbouring video frames. Second, pans in the start and end video clips reveal empty areas where no projection exists. This is not a problem for areas with geometry coverage as the underlying geometry still displays content, though with a cruder vertex coloured rendering. However, it is a problem for areas only covered by planes, such as the sky, as these have no texture without video projection. These extra empty areas create a large visual difference from the full 3D static transition, which has relatively few empty areas.

Looking at the per-set results, the full 3D dynamic transition only outperforms the other transitions in set 5. In this case, the pans work in the favour of the transition and the frame is almost completely filled with projected geometry. The opposite is true in the full 3D static case, where the pans introduce black areas. Had we used a more simple interpolation scheme for the full 3D static case of just interpolating the transition start and end camera poses (instead of all start and end clip frame poses), then the frame would be equally filled though the motion would be less smooth.

### Full 3D Static

Our most successful transition was significantly preferred over all transitions but warps across all sets. In the slight view change case, its perceptual score variance was much higher and so it was only significantly preferred over cuts. In the considerable view change case, it was significantly preferred over all other transition types. Participants generated many positive comments, only leaving negative comments when the geometry reconstruction was poor (sets 9 and 10). Participants also commented that it was noticeably more stable in sets with considerable shake (sets 7 and 8) due to the time freeze and reduced ghosting.

Sets 1 and 4 show particular improvement with scores 1.17 and 1.00 standard deviations higher respectively than the second ranked transition. Both of these transitions fill almost the entire frame with

content, have very few artefacts (one minor geometric anomaly in set 1 only) and have smooth, shake-free camera motions. Indeed, participant 9 states of set 4: “My top listed transition [Full 3D static] looks really nice, the only jarring artefact is the border of the frame sweeping across the introduced video.” This border is the frame edge introduced between the different coloured skies as the video frames are blended. This artefact could be reduced in real-time with feathering, or offline with Poisson blending [PGB03].

Apart from sets 9 and 10, sets 3 and 5 have comparatively low scores with the full 3D transition appearing 0.51 and 0.32 standard deviations lower than the top ranked transition. Set 5 was previously discussed in the full 3D dynamic paragraph, and we postulate that the full 3D static transition is less preferred because of the introduced empty regions. Set 3 is more difficult to explain: warp ranks first, with full 3D dynamic second and full 3D static behind in third (0.67, 0.41 and 0.16 standard deviations respectively). Warp generates a full frame with no empty regions and has minimal artefacts. Both full 3D transitions appear extremely similar, though the motion of the panning end video across the geometry during the dynamic transition creates a smoother overall camera motion and a less jarring ease back into video from the virtual view. While there are dynamic objects visible in this set, they are small — it is more likely to be the smoother camera motion which makes the preference difference.

### 6.3.6 Individual Sets

In Appendix B, perceptual scales are shown for each set, and specific features/artefacts are cross-referenced with comments to investigate the properties of the scene which have affected the result. Scene descriptions and special features are from Table 6.2.

Notably, set 10 presents a previously ignored effect. Participant 21 stated that this set introduces an interesting inherent clip meaning which does not appear in any other set. The first clip pans across a bridge and zooms towards the dome of a cathedral in the distance. Then, we transition to a shot upon the bridge also looking at the dome of the cathedral. The participant suggested that they anticipated moving onto the bridge because it was *presented* to them in the start clip through the pan. This kind of clip context and continuity understanding is the beginnings of storytelling and is well beyond the scope of this thesis, but it is interesting to consider the possible implications, such as misunderstandings and orientation losses, that could occur from a participant expecting to be taken somewhere by the content in the start clip. This effect may be less pronounced in a complete system where the user plots or is shown a route on a map before a video tour begins, rather than viewing a transition in isolation.

### 6.3.7 Outcomes

Our experimental results help develop rules for selecting appropriate transition types. There are many factors which may have contributed to the preference of participants, but slight vs. considerable view changes is a key factor for which we received statistically significant results. Warp transitions are the perceptually preferred transition type for slight view changes, and warps are significantly preferred over all other transitions except the full 3D transitions. In this case, the warp transition has a higher perceptual score and a much smaller variance (Figure 6.4). As such, our results indicate employing warps if the view rotation is slight, i.e., equal to or less than  $10^\circ$ . Further, in our experience with the system, slight

view change portal transitions that have good geometry reconstructions and do not suffer shake (similar to Scene 3, Figure B.4) will also provide high-quality results when using the static or dynamic full 3D transitions. In general, the success of the full 3D transitions are more scene dependent than the warp with the possibility of geometric errors and empty regions caused by matching or conflicting camera motions. However, if the full 3D transitions are used then the camera motion will be smooth — this is in contrast to the warp.

We use the static 3D transition for considerable view changes. In this case there is no discussion — the full 3D static transition is significantly preferred over all other transition types, and no other transition type is also preferred over any other (ignoring cuts). If video-video-geometry registration were accurate then dynamic 3D transitions should be at least competitive; these improvements are left for future work.

Our results also show that a dissolve is preferable to a cut. Should any portals fail to reconstruct, either from insufficient context or a failure of camera tracking, then it is always preferable to fall back to a dissolve instead of a cut. As one of our participants displayed an unwavering preference for cut transitions (verified in post-experiment questioning), we also allow manual choice of transition type as an override.

We will now describe outcomes that were not tested explicitly and are not significant, but from the per-transition and per-set analysis merit discussion:

- Participants did not seem to notice or care about dynamic objects in scene transitions. If all other artefact-causing issues are solved then this may become more important, but as it stands it does not appear to be significant. This is also expected to change if dynamic objects are the content focus of the videos.
- APC works better with considerable view changes and not zooms, that is, large angular view changes or large translations. Slight view changes tend to cause double images as the geometry contrasts with an image formed from a slightly different view within the cloud.
- Good video registration is imperative, though this is difficult to achieve under considerable shake with rolling shutter distortion. Set 4 shows us that inaccurate video registration can turn a convincing transition (as full 3D static) into one that is perceptually equivalent to a dissolve (full 3D dynamic).
- We cannot overlook the importance of clip content as a cue to where a participant expects to be after a transition. Set 10 exemplifies this. These story-telling cues are beyond the scope of this thesis, but would make for interesting further work.
- Camera motion in the start and end video clips is important for non-obvious reasons. Even with accurate video registration and smooth camera interpolation, panning and zooming in the clips can cause empty regions to appear, and these were disliked by our participants more than we anticipated. The effects of different camera motions are complicated, and we describe this further in the following section.

Start clip	End clip	Pan case	Full 3D dynamic empty areas	Largest empty areas
No pan	No pan		No empty areas	
Left pan	No pan		Slight empty areas to right	At $\frac{1}{4}$ and $\frac{3}{4}$ , none at $\frac{1}{2}$
Right pan	No pan		Slight empty areas to left	At $\frac{1}{4}$ and $\frac{3}{4}$ , none at $\frac{1}{2}$
No pan	Left pan		Slight empty areas to left	At $\frac{1}{4}$ and $\frac{3}{4}$ , none at $\frac{1}{2}$
Left pan	Left pan	Matching	No empty areas	
Right pan	Left pan	Converging	Empty areas to right	At $\frac{1}{4}$ and $\frac{3}{4}$ , none at $\frac{1}{2}$
No pan	Right pan		Slight empty area to right	At $\frac{1}{4}$ and $\frac{3}{4}$ , none at $\frac{1}{2}$
Left pan	Right pan	Diverging	Empty area to left	At $\frac{1}{4}$ and $\frac{3}{4}$ , none at $\frac{1}{2}$
Right pan	Right pan	Matching	No empty areas	

**Table 6.9:** Horizontal pan effects on empty areas in full 3D dynamic transitions. This table assumes that the start clip camera pose is to the right of the end clip camera pose, and directions should be reversed if the start clip pose is to the left of the end clip pose. Relative pan speeds affect the size of the empty area, where larger differences in pan speeds equals larger empty areas.

### Camera Motion Effects on Empty Regions

Camera motions in the start and end clips can cause empty areas to appear in the rendered transition due to the difference between the projection camera pose and the virtual interpolated camera pose. As the full 3D static transition requires a different interpolation method to the plane, APC, and full 3D dynamic transitions, this causes further complications. This section presents tables and diagrams that cover common pan and zoom cases. However, these tables and diagrams present ideal results: pans are assumed to move only in the horizontal direction and at equal velocities with no wobble or shake; zooms are assumed to have constant velocity. Real-world cases are more complicated, but these ideal results can be used to predict areas of empty regions. The tables and diagrams also explain why full 3D static and dynamic camera motions have different empty areas, and why this can effect perceptual preference so much as seen in sets 3 and 5 (Figures B.4 and B.6 respectively).

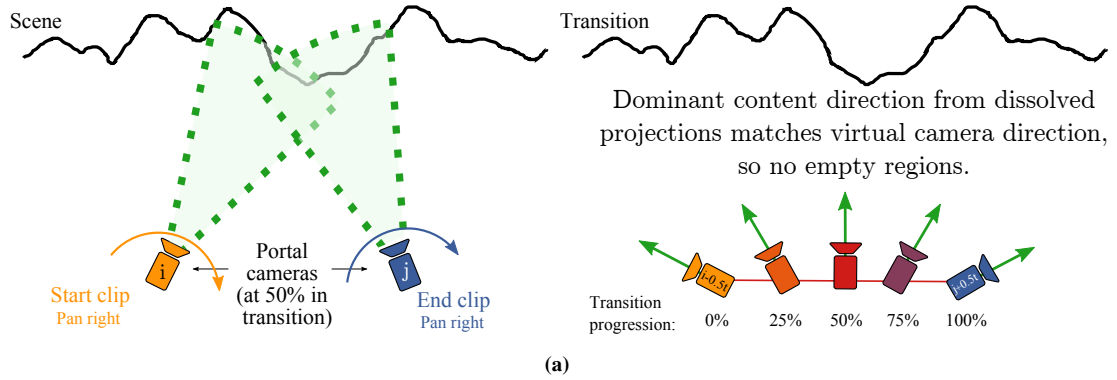
Table 6.9 presents the pan breakdown for full 3D dynamic transitions, with Figure 6.5 demonstrating the matching, converging and diverging cases. The ‘no pan/pan’ cases can be explained by simplifying these cases and so are not shown. The camera interpolation schemes for APC and plane transitions are identical to the full 3D dynamic transition. Table 6.10 and Figure 6.6 provide the same details for full 3D static transitions. Finally, Table 6.11 and Figure 6.7 explain empty areas for zooms. If a fixed focal length is used, then zooms change to translations (Section A.7.1). This does not affect empty areas.

### Artefact Hierarchy

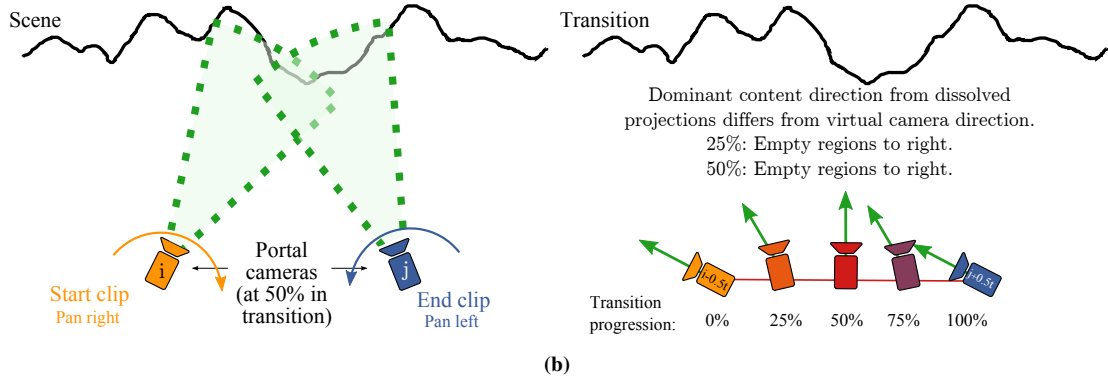
We would like to be able to produce a hierarchy of artefacts which orders or numerates their perceptual importance. Our experiment does not support the quantitative creation of such a hierarchy: an experiment which did, across different scene types, would be very involved and is beyond the scope of this thesis. Nevertheless, we can present a qualitative hierarchy for our scenes from our per-transition and per-set



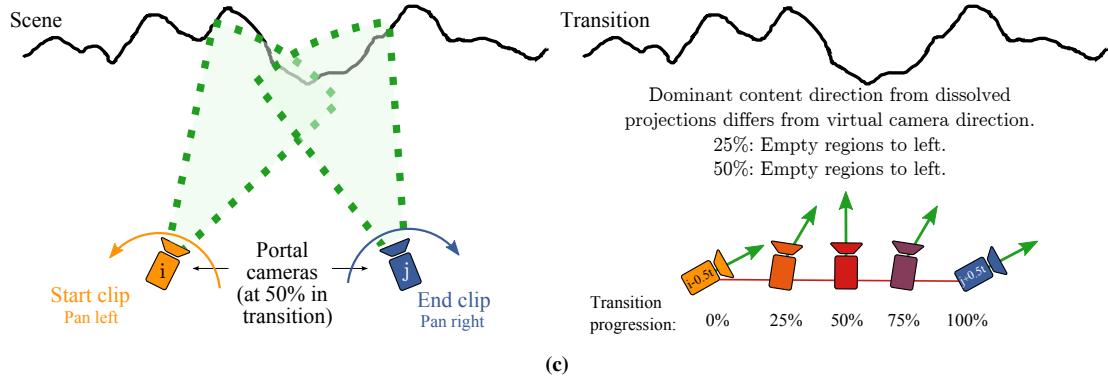
## Matching pans, full 3D dynamic:



## Converging pans, full 3D dynamic:



## Diverging pans, full 3D dynamic:



**Figure 6.5:** How horizontal pans in start and end clips affect empty areas during full 3D dynamic transitions. On the right, start and end clip camera positions and rotations are interpolated to create the virtual camera. Video content is projected onto the scene, with the dominant content direction shown in green (as projections are dissolved across the transition, save the middle of the transition, there is always one video clip which dominates). The difference in angle between the dominant content direction and the virtual camera direction notes the location of the empty area in the view. Relative pan speeds affect the size of the empty area, where larger differences in pan speeds equals larger empty areas.

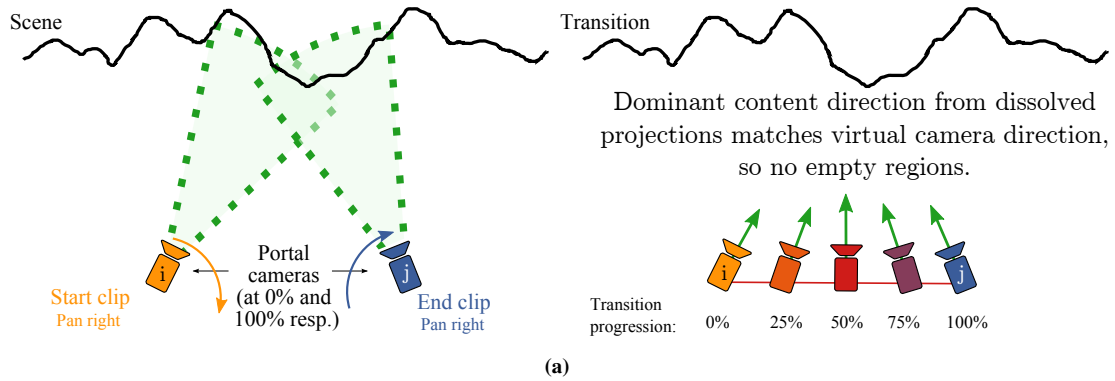
Start clip	End clip	Pan case	Full 3D static empty areas	Largest empty areas
No pan	No pan		No empty areas	
Left pan	No pan		Slight empty areas to right	At $\frac{1}{2}$
Right pan	No pan		Slight empty areas to left	At $\frac{1}{2}$
No pan	Left pan		Slight empty areas to left	At $\frac{1}{2}$
Left pan	Left pan	Matching	No empty areas	
Right pan	Left pan	Converging	Large empty areas to left	At $\frac{1}{2}$
No pan	Right pan		Slight empty area to right	At $\frac{1}{2}$
Left pan	Right pan	Diverging	Large empty area to right	At $\frac{1}{2}$
Right pan	Right pan	Matching	No empty areas	

**Table 6.10:** Horizontal pan effects on empty areas in full 3D static transitions. This table assumes that the start clip camera pose is to the right of the end clip camera pose, and directions should be reversed if the start clip pose is to the left of the end clip pose. Relative pan speeds affect the size of the empty area, where larger differences in pan speeds equals larger empty areas.

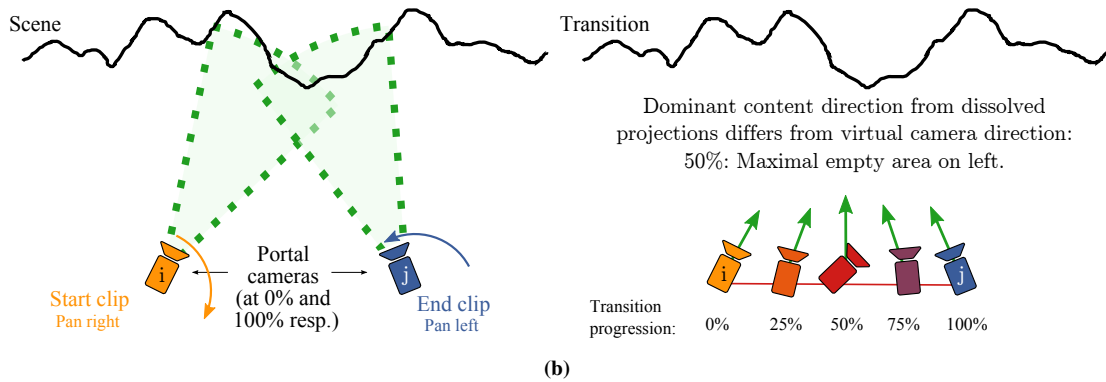
Start clip	End clip	Zoom case	Empty areas in frame	Largest empty areas
No zoom	No zoom		No empty areas	
Zoom in	No zoom		Surrounding start clip frame	At $\frac{1}{4}$ , none at $\frac{1}{2}$
Zoom out	No zoom		Surrounding start clip frame	At $\frac{1}{4}$ , none at $\frac{1}{2}$
No zoom	Zoom in		Surrounding end clip frame	At $\frac{3}{4}$ , none at $\frac{1}{2}$
Zoom in	Zoom in	Matching	No empty areas	
Zoom out	Zoom in	Contrasting 1	Surr. start clip, then end clip	At $\frac{1}{4}$ and $\frac{3}{4}$ , none at $\frac{1}{2}$
No zoom	Zoom out		Surrounding end clip frame	At $\frac{3}{4}$ , none at $\frac{1}{2}$
Zoom in	Zoom out	Contrasting 2	No empty areas	
Zoom out	Zoom out	Matching	No empty areas	

**Table 6.11:** Zoom effects on empty areas in transitions. This table assumes that zooms are of the same speed. Zoom velocities and distances affect the size of the empty area — the larger the difference between zooms, the larger the empty area. Figure 6.7 demonstrates the two contrasting cases and explains why they have different outcomes.

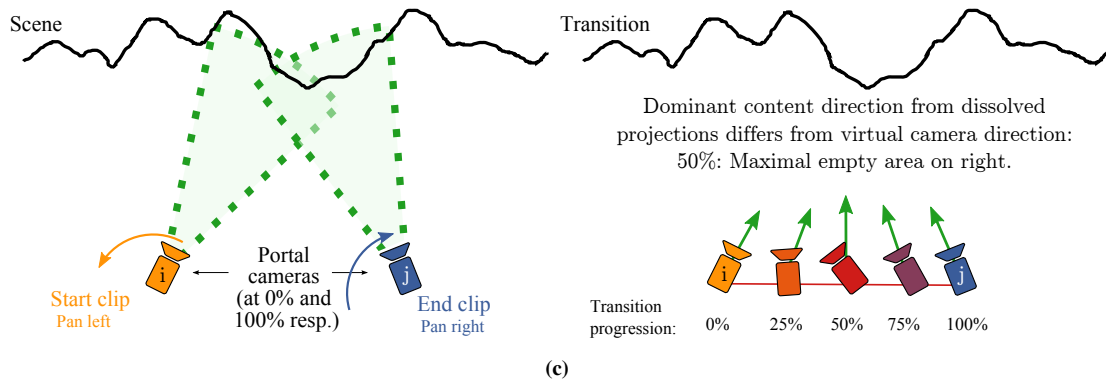
## Matching pans, full 3D static:



## Converging pans, full 3D static:

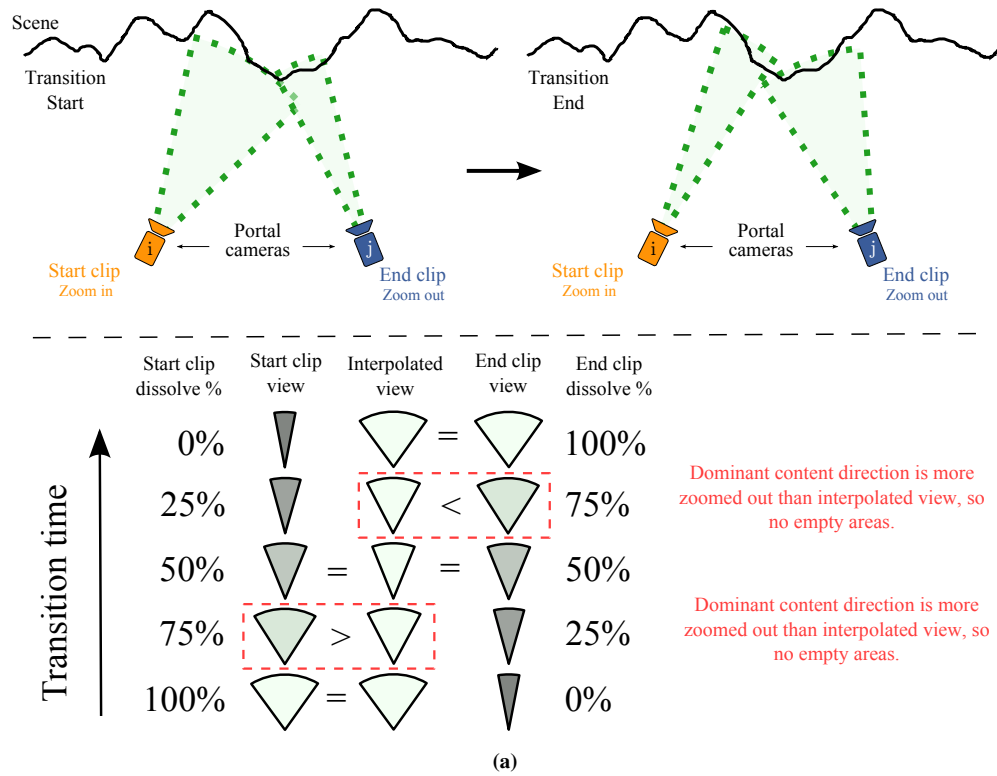


## Diverging pans, full 3D static:

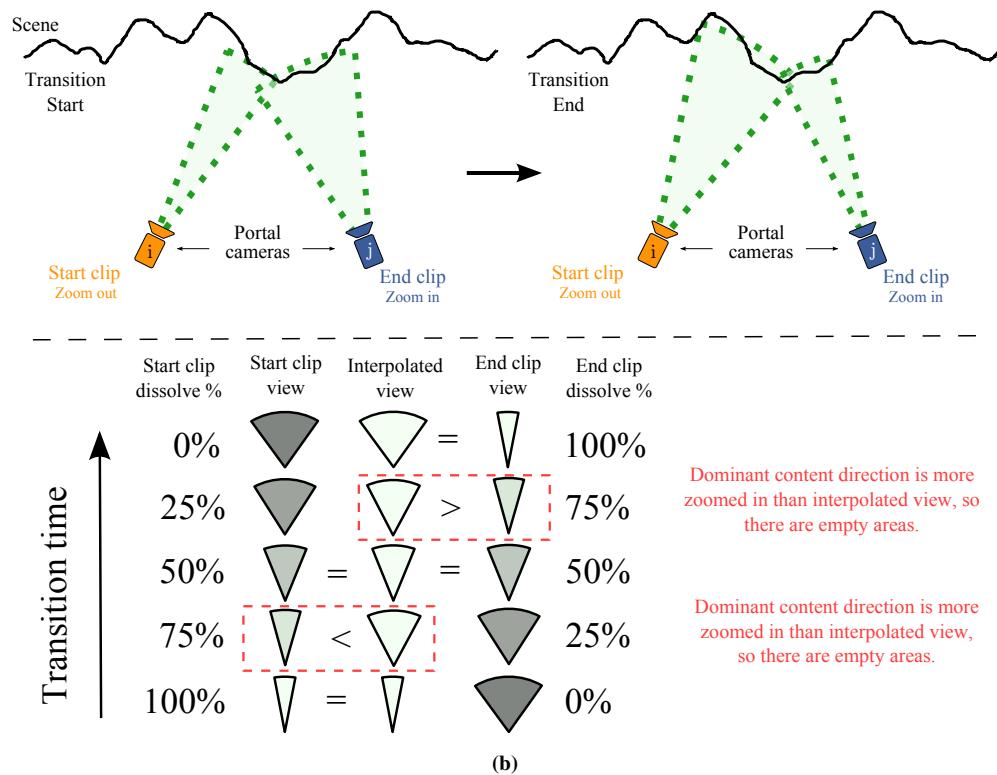


**Figure 6.6:** How horizontal pans in start and end clips affect empty areas during full 3D static transitions. On the right, the frames used for interpolation are different from those in the full 3D dynamic case (Figure 6.5, see Section A.7.3). Still image content is projected onto the scene from the portal frames, so the dominant content direction shown in green is consistently between the portal frames. The difference in angle between the dominant content direction and the virtual camera direction notes the location of the empty area in the view. Relative pan speeds affect the size of the empty area, where larger differences in pan speeds equals larger empty areas.

## Contrasting Zoom In/Out:



## Contrasting Zoom Out/In:



**Figure 6.7:** How zooms in start and end clips affect empty areas during camera interpolated transitions. Top: Transition zoom demonstration. Camera frustums are simplified to circle segments. Bottom: The effect of camera interpolation on camera views. When the dominant content projection has a larger field of view than the interpolated virtual camera there will be no empty areas.

analysis and corroborate this with the comments of participants.

Ghosting on objects which should be static has a significant effect on preference. This is demonstrated in sets 4, 7, 8, 9 and 10, where the warp (in slight view change cases) and full 3D static transitions score highly on the perceptual scale. These transitions have very little static ghosting. This is clear in set 4 where a large perceptual difference exists between the static and dynamic full 3D transitions which we argue is caused by static ghosting from errors in video registration. We include skewing and geometry errors in this case as all have similar causes.

Surprising to us was the effect of empty regions in the transitions. In sets 1, 2, 5, and 6 the major difference between the full 3D static and dynamic transitions was empty regions, and in all cases the transition with less empty regions was perceptually preferred.

APC pepper noise received many negative comments. These artefacts can be reduced by using more points or by implementing true video APC, though the technique is already quite expensive computationally. The true video APC case is considerably more expensive, and is not feasible to implement on current hardware.

Image-plane artefacts in the warp, such as bad correspondence swirls and undulating or flickering parts, seem not to be important in slight view change cases. In considerable view change cases, where the warp is perceptually less preferred, it is more difficult to judge whether these artefacts have an effect because much of the resulting image is different and so correspondence cannot be found.

Given these findings, we order the artefacts such that:

- Ghosting on static objects  $\approx$  empty regions
- > pepper noise
- > swirls  $\approx$  temporal flickering
- > ghosting on dynamic objects.

While the evidence supporting this ordering is anecdotal and comes from interpretation, it should help direct effort in correcting artefacts in these and other transitions.

## 6.4 Summary

In this chapter we have explored video transitions as a way to join two video clips. The aim of this chapter was to perform an experiment comparing many different transition types over many different scenes, and to study in an experiment which transitions were preferred by participants. The space of all potential video transitions is very large, and so applying these experimental results categorically is not possible, but we devise heuristics to move towards automatic scene-dependent transition type selection, and suggestions for which artefacts to minimize to improve the quality of transitions.

We tested 7 transition types: cut and dissolve transitions from movies, then warp, plane, and full 3D static and dynamic transitions from more modern computer graphics applications. We tested 10 scene types, each with a slight and considerable view change condition. We discovered that there was a strong preference for full 3D static transitions in the considerable view change case, and a preference for warp

transitions in the slight view change case.

In the process of building the transitions subsystem of the Videoscapes system and preparing for the experiment, we have described the many decisions and processes that go into rendering graphical transitions. Having analysed the per-transition and per-set results, it is clear that in some cases certain transitions produce results which demonstrate well the change in orientation from one video to the next. However, no transition is universally applicable: under certain conditions, such as shake or geometry reconstruction failure, it is less clear what benefit graphical transitions deliver. Chapter 8 discusses how these issues could be solved to produce more robust transitions.

## Chapter 7

# Videoscape Exploration

## 7.1 Introduction

Historically, exploring a video collection has been an arduous task. A film or broadcast shoot or archive required manual labelling, indexing, and retrieval, usually with the help of an assistant or librarian [Dmy84, Mur01]. Finding content within reels was by linear access. Digital video brought about automatic indexing and retrieval with random access, but labelling was still manual. Online video archives provide vast repositories, freedom of access, and fast retrieval, but the way in which this content is accessed is still very much by manual labelling and indexing.

Work in computing to provide alternative video collection browsing methods arguably began just over 30 years ago, with the geographical video system of Lippman et al. [Lip80]. With increases in computation and the introduction of digital photography, more advanced methods improved upon this Movie-map system [Nai91, Nai94, Nai96, KF01, UCK<sup>+</sup>03]. However, in all cases, the geographical links between movies are defined manually by hand with no computer-derived content-based knowledge.

More recently, attention has been brought to how we browse photo collections [SSS06, Goo07, SGSS08], with automatic or semi-automatic geographical and content-based systems now in commercial use [Mic08]. Extensions to video exist [PWC08, KN09, BBPP10] but, as the dimension of the problem is now larger, they deal with specific cases of the larger problem of providing video collection browsing interfaces (see Section 3). Our work tries to bring these works together and provide a more general, multi-faceted solution to the problem of exploring video collections.

From our literature review, we identified 5 recommendations for any video collection exploration interface (Section 3.4). These are:

1. Presenting many video streams at once may be confusing.
2. Map-based browsing is liked, is useful, and helps scene comprehension.
3. Pins are not a good abstraction for videos.
4. Any representation of video frames on a map must be density aware.
5. Video collection summarization should be provided as a means of exploration.



Over the course of this chapter, we will explain how these recommendations are met by our interface. Our interface also has the more general goals of quantitatively outperforming existing interfaces for finding specific content, being preferred qualitatively for attributes such as sense of place and spatial awareness, and for participants to want to use our system. We will experimentally verify that our interface meets these goals when compared to existing systems, and that users would want to use our system.

This chapter proceeds to describe the interface we have developed, explaining important features and how they help improve the video exploration experience (Section 7.2). Then, we discuss how our collection structure and interface aids in labelling and retrieving content for text and image searches (Section 7.3). Finally, we verify our interface with three different user-study experiments (Section 7.4). Each experiment is designed to address a different part of the system: one assesses whether we improve spatial awareness when transitioning between videos, one assesses preference for video tours, and one assesses the tools we provide for browsing video collections. While these experiments are limited in scope, we receive both quantitatively significant results and positive qualitative feedback, which suggests that we improve interfaces for video collections.

## 7.2 Videoscapes Interfaces

We have developed a prototype explorer application (Figures 7.1 & 7.5) which exploits the Videoscape data structure and allows seamless navigation through video collections. We identify three workflows for interacting with the Videoscape, and develop three different interfaces to accommodate these workflows. The explorer application itself seamlessly transitions between these three workflows via animations. This important aspect maintains the visual link between the graph (via its embedding) and the videos through workflow changes, and helps the viewer from becoming lost. The three workflows are:

**Interactive Exploration Mode** (Section 7.2.1) We assume that only video data is available, and create an interface for navigating between videos which relies only on the user picking what they wish to see next in a video collection exploration.

**Overview Mode** (Section 7.2.2) The Videoscape graph is embedded into a two-dimensional cartographic map or a three-dimensional globe. Portals are represented as icons placed above their geographic objects of interest, and tours can be built by selecting portals in turn.

**Fast Geographical Browsing Mode** (Section 7.2.3) Individual videos are represented geographically by the path that was physically travelled during capture, allowing fast non-linear access to large video collections.

This interplay between the three workflows allows for fast exploration of large Videoscapes with many videos, and provides an accessible non-linear interface to content within a collection of videos that may otherwise be difficult to penetrate.

### 7.2.1 Interactive Exploration Mode

Watching videos is often an immersive full-screen experience, and a Videoscape need not be any different (Figure 7.1). In this first workflow, an initial video is chosen and plays full-screen. As the video

progresses and a portal becomes near in time (within 5s), we notify the viewer with an unobtrusive camera icon in the top-left corner of the screen. If the viewer chooses to switch videos at this portal opportunity, they can move the mouse to cause a thumbnail strip of destination choices (at neighbouring graph nodes to the upcoming portal) to smoothly appear from the top of the screen. If the viewer does not wish to switch, the original video continues to play uninterrupted.

Once triggered, the thumbnail strip figuratively asks “what would you like to see next?”. The video slows to give time to make a decision, and the viewer can also pause the video if desired. Each thumbnail shows a static view of the frame at the end of one edge connected to this portal node, providing the view at which the next path decision might be made. Each thumbnail also provides access to all video content along the entire edge via scrubbing (right-click dragging with the mouse), allowing the viewer to quickly scan the contents of future paths and assess their suitability. It might be thought that these thumbnails should show video or video summaries instead of a single frame; however, recommendation 3a) (Section 7.1) suggests that we should not show too many videos at once and potentially overwhelm the viewer.

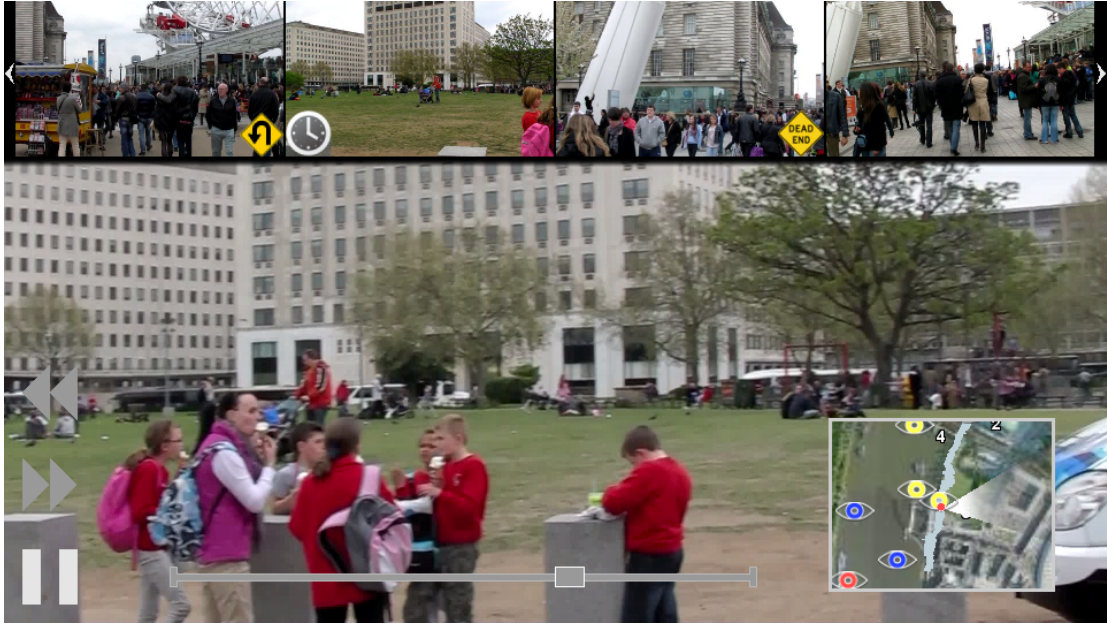
With a thumbnail selected, the strip disappears and in the background our system generates an appropriate transition (as defined in Chapter 6) from the current video view to the chosen new edge video. This new video, after the transition, begins with the contents of the current view as seen from a different spatio-temporal location, and ends with the chosen destination view. Audio is cross-faded as the transition is shown. This process then repeats as a portal approaches in time in this new video, and as such the viewer walks around the graph while taking in their desired views.

This paradigm of moving between views of scenes is applicable when no other data beyond video data is available. This forms our baseline experience.

## Temporal Exploration

We add a clock icon to the destination choice thumbnails when views are time-synchronous (see Figure 7.1). This represents moving only spatially but not temporally to a different video. While this might seem like a counter-intuitive icon — why show a clock when the only thing that doesn’t change is time — it is difficult for any icon to represent this complicated idea (especially attempts at representing the concept of ‘spatial movement only’ in an icon). As such, we believe it is suitable as an immediate associator for time (“Something special will happen with time if I select this thumbnail”) as the effect of this destination thumbnail choice will be clear once the transition and new video edge are shown.

A temporally consistent exploration of the Videoscape may be expected for videos that are taken during an event, such as at a BMX tournament. We enable this by restricting portal transitions to other videos that have the same time stamp as the current frame (for this, synchronized time stamps must be available, see Section 5.6.2). Additionally, we enable purely temporal exploration of the video data, i.e., the user can ignore portals and transition (using a blend) to any other video at the same time instance. This is a useful override in time-critical databases when the viewer must see another video stream immediately. Even though a portal does not exist at these points, with a general direction of motion and distance known between the two video streams it would be possible to add a motion cue to aid in orientation (such as a blurred matched wipe with the speed and length adjusted approximately for distance).



**Figure 7.1:** An example of a portal choice in the interactive exploration mode. The mini-map follows the current video view cone in the tour. Time synchronous events are highlighted by the clock icon, and road sign icons inform of choices that return to the previous view and of choices that lead to dead ends in the Videoscape.

### Intelligent Fast Forward

Figure 7.1 also shows additional features. The fast forward/fast rewind buttons (above the pause button, to the left, currently not selected) act as expected and simply increase playback speed either forward or back (including through transitions). However, with our pre-processing, it is simple to provide more advanced functionality with these buttons. As we compute features throughout videos in our collection, we have added the ‘intelligent’ fast forward of Pongnumkul et al. [PWC08]. This speeds through videos and returns to normal speed during ‘high-quality’ or ‘coherent’ sections. However, we question the validity of these metrics in Section 3.3.2, and find that they provide an experience which is less preferred than our summarization method in Section 7.4.2; hence, this feature is turned off by default. Similarly, we can exploit the Videoscape graph and cause the fast forward button to return to normal speed during a time window around portals. Again a type of intelligent fast forward, this relies on the popularity of content within the collection to represent ‘high quality’ or ‘interesting’. For certain kinds of collections, and for certain applications, e.g., finding touristic landmarks among a holiday collection, this would be well suited. Intelligent fast forward functions act as summarizations of a place, and so help to meet recommendation 3e).

### Mini-map Mode

If a graph embedding is available, then interactive exploration mode supports a togglable mini-map. Should GPS data be available, the mini-map displays and follows the video position in time from overhead. Should orientation data also be available, then we add a view frustum to the mini-map which rotates in time. When hovering over a destination choice thumbnail, the mini-map shows the frustum

and real-world point on the mini-map, and updates the timeline. The mini-map acts as a strong geographical cue to prevent the viewer from becoming lost during the tour. When transitions take place, the mini-map interpolates between start and end video positions and frusta. This helps meet recommendation 3b) (Section 7.1).

### Miscellaneous Interface Elements

If a destination thumbnail choice leads to a dead end in the graph, or if a choice leads to the previously seen edge but in reverse, we add commonly understood road sign icons as well ('dead end' and 'U-turn', see Figure 7.1).

The timeline lies at the bottom of the screen and acts subtly differently to a normal video playback timeline. As the user is interactively walking around the graph, we do not know the total time length of the exploration, and so the timeline visibly extends to encompass their journey. When selecting from destination choices in the thumbnail strip, the timeline again extends to show how much time would be added to the tour if that edge were taken. In this way, the timeline is a visualization of the graph path. As the size of the timeline approaches the limits of the screen space available, it begins to increase the size of time unit that each pixel represents so that any length of tour (or any upcoming edge) can be represented. The user can also mouse over the timeline and drag to a previous point in the tour or further into the current edge.

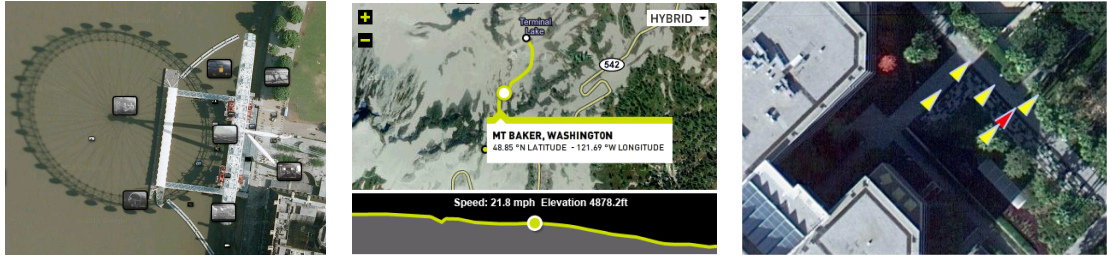
#### 7.2.2 Overview Modes

At any time, the mini-map can be expanded to fill the screen, and the viewer is presented with a large overview of the Videoscape graph embedded into a globe [BKM<sup>+</sup>07] (Figures 7.3 & 7.5, top). We add an eye icon to the map for each portal node in the graph<sup>1</sup>). Portals intuitively represent some landmark — a building, a park, a statue, etc. — and so have a natural geographical representation on a map of the physical area of the landmark. Graph edges between portals are connections between videos, and travelling along an edge represents switching from looking at one landmark to looking at another. This is a complicated idea which requires a sophisticated representation. We do not directly represent graph edges in our embedding as 1) they are not intuitively geographical, and 2) even with a good representation, there are usually too many edges in the graph for the embedding to be clear and comprehensible. Instead, we choose to keep the same metaphor as the interactive exploration mode — “what would you like to see next?” — and so we symbolize view choices with eye icons which encompass all edge information.

Portal eye icons are the key exploration tool in this mode. Existing interfaces typically use pins, trails, or camera frusta to represent videos (Figure 7.2). Pins and trails inform only from where a video was taken, and not in which direction it was looking nor where exactly the content is located. Camera frusta inform from where the video was taken and in which direction, but it is often difficult to know exactly where the content is located. The geographical location of a portal eye is estimated from converging sensor data so that the eye icon is placed approximately above the viewed scene in the video.

---

<sup>1</sup>The reverse assignment of portal as edge and video as node is as intuitive for displaying the topology of the Videoscape; however, it becomes less intuitive for specific geography as a video can cover a wide geographical areas.



(a) A video represented as a pin, here with a small embedded thumbnail. Pins tell the viewer from where the video was taken at one time instance of its duration. Image courtesy of [Gool2a].

(b) A video represented as a trail, here also with elevation. A trail tells the viewer from where the video was taken at all time instances. Image courtesy of [Con11].

(c) A video represented as a view frusta. View frusta tell the viewer from where the video was taken at one time instance, and in which direction. Image courtesy of [WTA11].

**Figure 7.2:** Different options for representing videos that are currently used in map-based image and video exploration softwares.

The position and the yaw component of the orientation data forms a vector in geographical space for each portal frame. Averaging the intersection points of these vectors provides the geographical location of a portal. In this way, portal eyes switch from a camera-focused abstraction to a content-focused abstraction: instead of browsing a collection of cameras in hope of finding the desired content, the viewer geographically browses the content directly. Each portal eye abstraction represents many short spans of content within video clips, but has problems in displaying the view of a camera over time. We combine portal eyes with trails to overcome this limitation (Section 7.2.3).

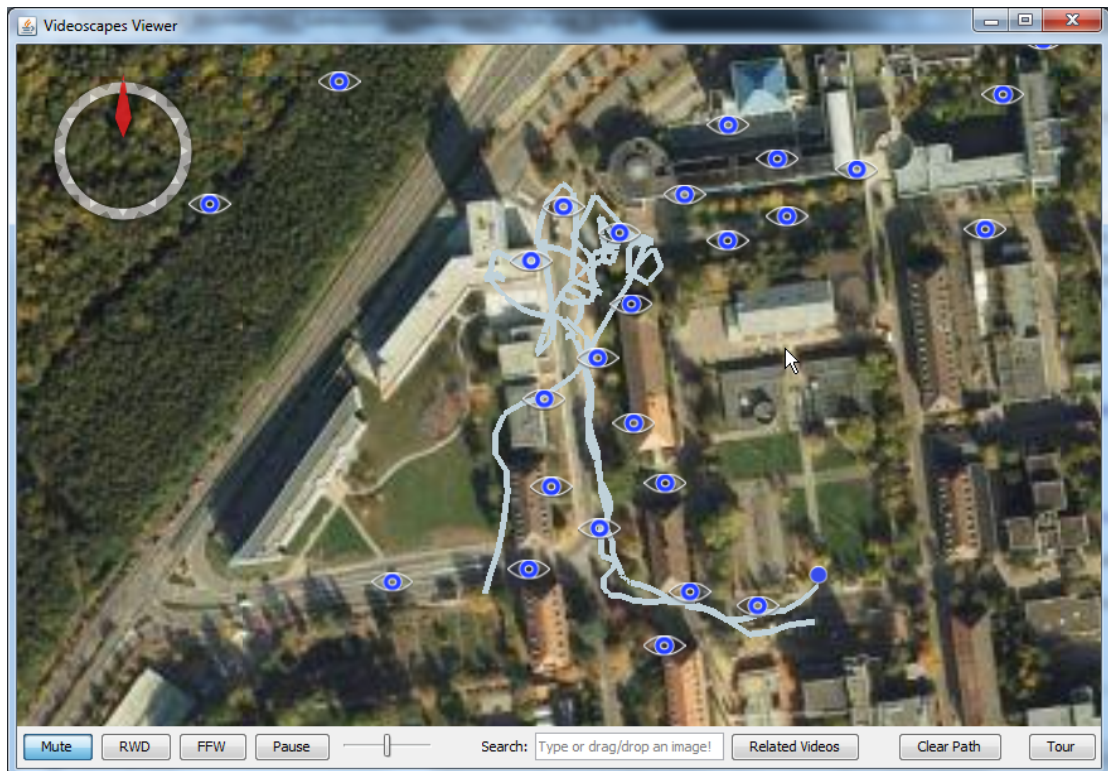
As a Videoscape can contain hundreds of portals, we adaptively change the density of the displayed eyes so that the user is not overwhelmed. Eyes are added to the map in representative graph connectivity order, such that the most-connected portals are always displayed first. As the viewer zooms into the map and the portal density decreases, less-connected portals are added to the map.

When hovering over an eye, we inlay images of views that constitute the portal, along with cones showing where these views originated (Figure 7.3b). The viewer can construct a video tour path through the graph either by clicking eyes in sequence or by clicking start/end eyes<sup>2</sup>. The defined path is summarised in a strip of video thumbnails that appears to the right. As each thumbnail can be scrubbed, the suitability of the entire planned tour can be quickly assessed. The generated video tour can then be played with joining transitions, after which the interface returns to overview mode.

The ideas in this interface mode meet recommendation 3b), 3c) and 3d) (Section 7.1) in providing a map-based exploration with a new density-aware icon abstraction for video content. Video tour paths act as geographical summarizations of a place, and so help to meet recommendation 3e).

<sup>2</sup>In this case, Dijkstra's shortest-path algorithm [Dij59] finds a tour. This is primitive but suitable for short tours; unfortunately, anything more complex is beyond the scope of this thesis. Section 8.3 contains discussion on appropriate future work to correct this limitation.





(a) Portal eye icons in overview mode along with travelled path lines for fast geographical browsing.



(b) Highlighting an eye icon inlays a frame of video and highlights frusta from all videos viewing that scene. The specific view shown has its frusta highlighted in yellow. This view changes over time to cycle through all views.

**Figure 7.3:** Overview mode with eye icons.

## Unsuccessful Alternatives

Earlier alternatives of the overview mode employed more direct embeddings of the graph and had clarity problems (Figure 7.4), and systematic changes were introduced to overcome these problems. Initially, only GPS data was recorded along with each video. In this case, the location data available for a portal only includes from where each video was taken and not where the camera is looking. This data does not allow us to place an eye icon above the portal landmark, and so finding portals is much more difficult as the location of each portal in the embedding is not particularly informative geographically. To correct this problem, we captured orientation data along with GPS data, allowing us to perform sensor fusion and to intersect view frusta to find the location of the portal landmark.

We tried to represent graph edges directly on the map, but they do not translate well geographically if they are shown directly as edges between portal locations. Our initial prototype showed solid lines to represent geographical paths travelled when videos were captured, then dashed lines from these paths to portals (where the video matched others in the collection). Figure 7.4 demonstrates that this method is largely indecipherable as portals group videos which may have very different geographical locations. We expected dotted lines to show patterns in the graph structure, but there are so many edges which cross that these lines are a hindrance to understanding.

Video frame thumbnails which appear when highlighting a portal also used to hover over the map at the location of the portal, but this often obscures important information such as camera frusta. Instead, in the current system, the thumbnail is anchored to the bottom-left corner of the display window so as not to obscure any information.

### 7.2.3 Fast Geographical Browsing

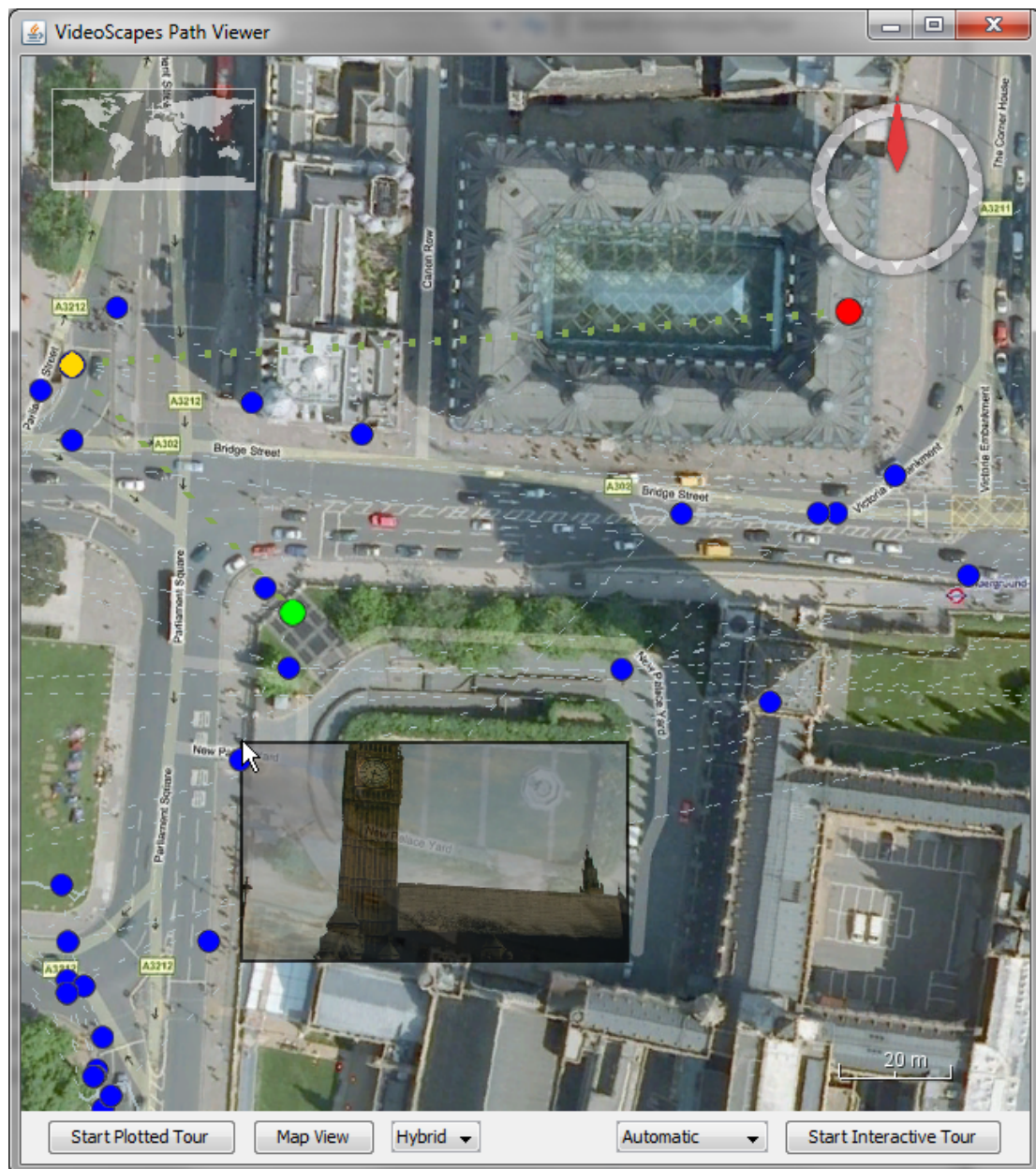
In this workflow, we draw real-world travelled paths, or trails, onto the map as polylines (see Figure 7.3). When hovering over a line with the mouse, the appropriate section of video is displayed along with the respective camera frusta. This is more than scrubbing as the video continues to play and the frusta updates synchronously. The video is typically shown side-by-side with the map to expose both video and polyline detail. The viewer has full control over the size of the video should they prefer to see more or less of the map (Figure 7.5, bottom). With fast globe zooming and per-second geographical segmentations of videos along the trails, this is a very fast way to scan long videos or many videos in the same geographic location. However, this workflow is not well suited to static cameras as it does not provide easy access to all of the content shot across time from one location. Here, a small time counter or slider could be provided to jump through time. Alternatively, to extend the hovering interaction style, a small winding circle could be displayed next to the location to cycle quickly through time.

Using this mode, as time progresses and the video plays, portals are identified by highlighting the appropriate eye and drawing secondary view cones in yellow to show the position of alternative views. Clicking when the portal is shown appends that view to the current tour path.

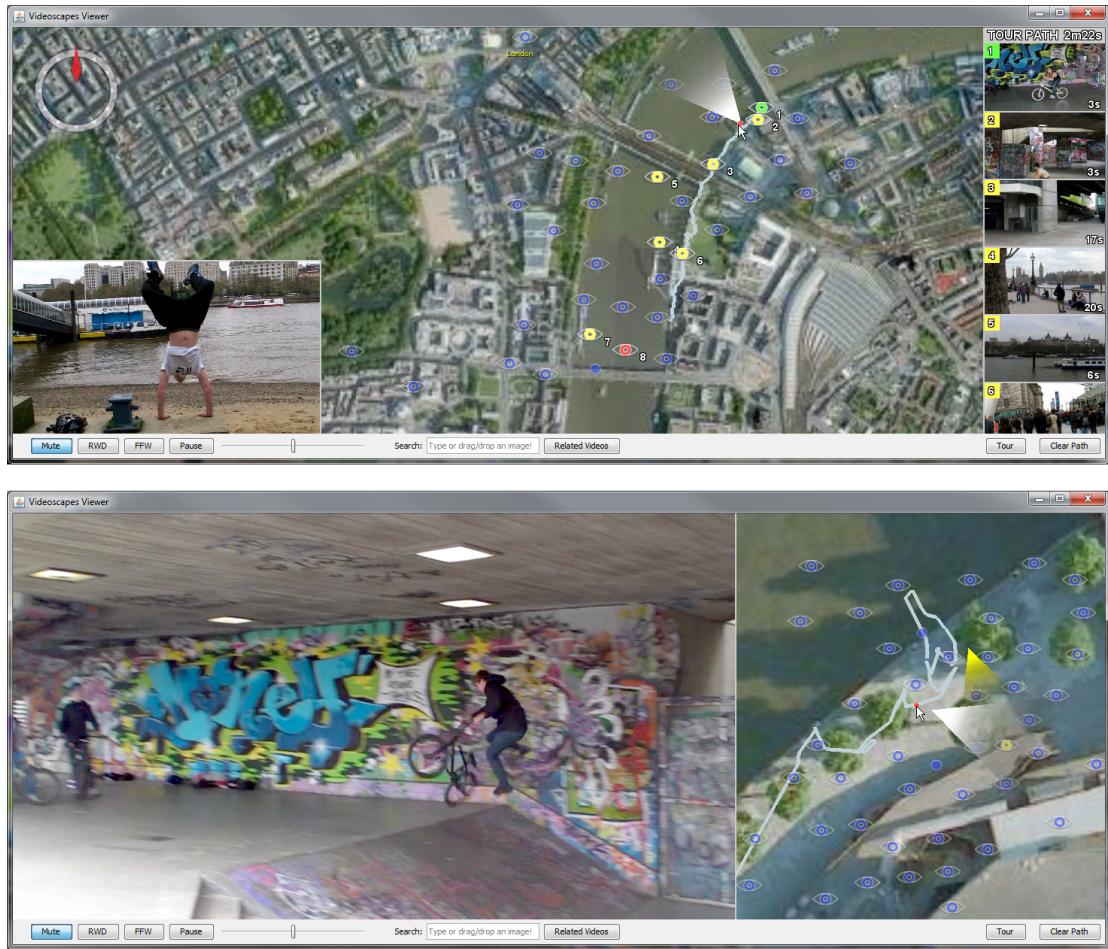
### 7.2.4 Workflow Switching Animations

The explorer application switches between the three workflows via animations to maintain the visual link between the graph, via its embedding, and the videos. The number of animations necessary is small:





**Figure 7.4:** Earlier prototype of the overview mode. A short plotted tour around Big Ben, starting from the green node, traveling to the yellow node, and ending at the red node. The user is hovering over an edge (representing a video), and a video thumbnail of that edge is playing. The portal node locations are not directly over their content on the map, which makes geographically finding content difficult. The dotted-line edge representation makes finding connections between portals difficult as these often cross and cover large geographical areas.



**Figure 7.5:** Top: Our interface for the path planning workflow. A tour has been defined, and is summarised in the interactive video strip to the right. Bottom: Our interface for the video browsing workflow. The video inset is resized to expose as much detail as possible, and alternative views of the current scene are shown as yellow view cones.

**Overview  $\Leftrightarrow$  Interactive** The overview graph embedding (usually as a 2D map or globe), which currently fills the window of the application, shrinks down to the size and position of the mini-map while keeping centred the current geographical location of the video. This shrinking reveals the interactive exploration interface behind, and so provides a smooth transition from large overview map to mini-map. We feel this aids orientation in the viewer as sight of the location in the embedding is never lost. When moving from interactive exploration to overview modes, the animation repeats but in reverse. If the mini-map is not turned on, then it fades in before the switch occurs to allow the transformation into overview mode.

**Overview  $\Leftrightarrow$  Fast Geographical Browsing** The lines representing the travelled paths during capture are drawn along with the embedding onto a 2D map or globe, and a large video inlay appears when hovering over lines. Strictly, there is no animation here as the workflow switch is transparent: both workflows exist on the same map.

**Fast Geographical Browsing  $\Leftrightarrow$  Interactive** As in the overview case, as the fast geographical browsing is also map-based.

### 7.3 Videoscape Labelling, Propagation, and Search

We augment the browsing experience by providing semantic labels to objects or locations in videos. This feature has been demonstrated in photo exploration applications [SSS06, KNC<sup>+</sup>08], and we adapt it here to the Videoscape. For instance, if given the names of landmarks, then we can allow keyword-based indexing and searching. Users may also share subjective annotations with other people exploring a Videoscape (e.g., "Great cappuccino in this café"). Further, the Videoscape graph gives us a structure with which to propagate labels. For example, if a user labels a landmark in one portal, we can push this label to all connected examples of that landmark in the graph.

A Videoscape provides an intuitive, media-based interface to share labels: During the playback of a video, the user draws a bounding box to encompass the object of interest and attaches a label to it. Then, corresponding frames  $\{I_i\}$  are retrieved by matching feature points contained within the box. As this matching is already performed and stored during Videoscape computation for portal matching, this retrieval reduces to a fast search. For each frame  $I_i$ , the minimal bounding box containing all the matching key-points is identified as the location of the label. These inferred labels are further propagated to all the other frames (matching  $F_i$ ) using optical flow or sensor data computed in the initial filtering stage of graph construction (Section 5.2). If more than two bounding boxes are identified for a single label in a frame then we simply construct a superset box. As the quality of individual key-point matches varies, the inferred bounding box may contain only a part of the object of interest. Thus, we show the centre of the box as the location when superimposing tags to video frames, as can be seen in Figure 7.6. As labels are propagated using the Videoscape graph, the accuracy of propagation to other videos is identical to that of the portal frame matches in the graph construction.

#### Image/Label-based Search Mode

We allow searching with images and labels to define tour paths. For image search, the user provides any number of images to our system. Providing a single image acts as a direct image search, and shows content in the video collection with similar image features. Providing multiple images defines a tour through the graph which takes in all views in images matched to the collection. To match, image features are matched against portal frame features, and candidate portal frames are found. A scrubbable video list appears showing the best matching candidates and from these a path can be selected. A new video tour is generated, bookended with warps from and to the submitted images.

For label search, the viewer provides key words and matching results are returned in a video list (as in image searches). Tours can similarly be defined by providing a set of text labels.

#### Label Graph Filtering

When the number of videos and corresponding portals is large (which is the case for a realistic application of the Videoscape), depicting the entire graph at once may be overwhelming. Currently, we vary the amount of portal eye icons shown to manage this complexity. However, we could exploit the meta data attached to the videos in the graph. Labels can be used as filters so that only the corresponding subgraph is visualized: if the user chose to filter with the "Big Ben" label, then we could retrieve the sub-graph which consists of nodes corresponding to these queries, their neighbours, and their connecting edges.





**Figure 7.6:** An example of label propagation in video. Left: Source video to be labelled. Right: Target video for label propagation. In the top two rows, the viewer creates a label for one frame in the source video (a), by drawing a rectangle and typing a label name and description. The target video has no labels so far. In the third row, this label is propagated to consecutive frames in the same video through key frames, and the label is transferred to the key frames of our target video and all other connected videos. This label is then subsequently propagated through the remaining frames of these connected videos.

## 7.4 Experiment: Interface Evaluation

We wish to evaluate the Videoscapes system experimentally to discover whether it improves upon current interfaces to video collections. We perform user studies to present tasks to participants which assess different parts of Videoscapes against existing systems, to gain insight into the performance of each interface component.

Evaluating our prototype interface in a meaningful way is challenging: existing systems do not provide comparative functionality, yet we do not wish to provide a trivial comparison. Equally, evaluating our large system as a whole is likely uninformative to the community as it would be too specific to our system. As such, we perform three different user studies designed to provide quantitative and qualitative feedback for major components of our system. Each study compares user performance/experience with Videoscapes to that achieved with existing alternatives:

**Spatial Awareness** We quantitatively and qualitatively assess spatial awareness through a transition from video to video, measured with a map task and a questionnaire. We compare Videoscapes, with 3D transitions and map-overlaid view frusta, against current map-based interface systems, with cut transitions and geolocated pins.

**Video Tours** Our video tours around the graph are, in spirit, geographical video summarizations. With a questionnaire, we qualitatively compare our video tours to two existing video collection summarization tools: automatically-edited sequences and intelligent fast forward.

**Video Browsing** We perform a quantitative task-based evaluation of our interface against a common commercial tool and a state-of-the-art research method applied to video collections. We ask participants to browse a video collection for specific footage, and canvass their qualitative responses with a questionnaire.

### Participants

Each of our 20 participants performed three experiments. All participants were self-assessed as familiar with the geographical area depicted in the tasks (different areas of London). We asked each person how long they had lived in London, and discovered a mean residence length of 3 years (max. 10 years, min. 3 weeks). An alternative is to use participants who did not know the geographical area in the dataset, either by using tourists or by using a ‘foreign’ dataset. Our choice of London dataset and familiar participants was solely driven by logistical considerations.

#### 7.4.1 Spatial Awareness Experiment

We designed a study which attempts to measure how much spatial awareness is retained through video transitions with and without geometry-based reconstructions. This experiment ties together video transitions, maps, and view frusta to see whether the coupling of these parts aids orientation when watching separate videos in a collection. The experiment proceeds as follows (summarized in Figure 7.7; Chapter D shows images of all website interfaces and sample output):



**Figure 7.7:** *Spatial awareness experiment steps.*

- ① A participant sees an overhead aerial imagery map marked with a ground-truth pin and a view direction. The pin marks the real-world camera position of video 1. After 8 seconds, a visible countdown begins (3...2...1).
- ② The map is removed and the participant watches a short clip of video 1 transitioning into video 2.
- ③ The video is removed, the map reappears, and the participant marks on the map the location and direction travelled to after the transition into video 2.

Objectively, we measure the deviation from ground truth of the position and direction marked with the red pin by the participant. Participants are free to replay the video and reposition the pin/direction as many times as they wish, and are also free to translate and zoom the map. When placing the pin for the second video, the pin for the first video is present on the map. Ground truth is generated from GPS coordinates hand-refined with local knowledge and known positions of the camera shots.

We test two conditions in our experiment: 1) cut transitions without providing view directions on the map (i.e., just a location pin; the condition most akin to other existing systems, in particular to a multi-video variant of Pongnumkul et al. [PWC08]), and 2) static 3D transitions with provided view direction on the map (the Videoscapes condition). The first condition, as our comparison case, is what viewers would see in current geographical video collection exploration softwares. Pins are used to represent geolocated videos, and switching between videos causes a cut between streams. Each participant completed 1 practice trial as often as they wished, followed by 8 randomly ordered trials each of a different scene, of which 4 are from each condition.

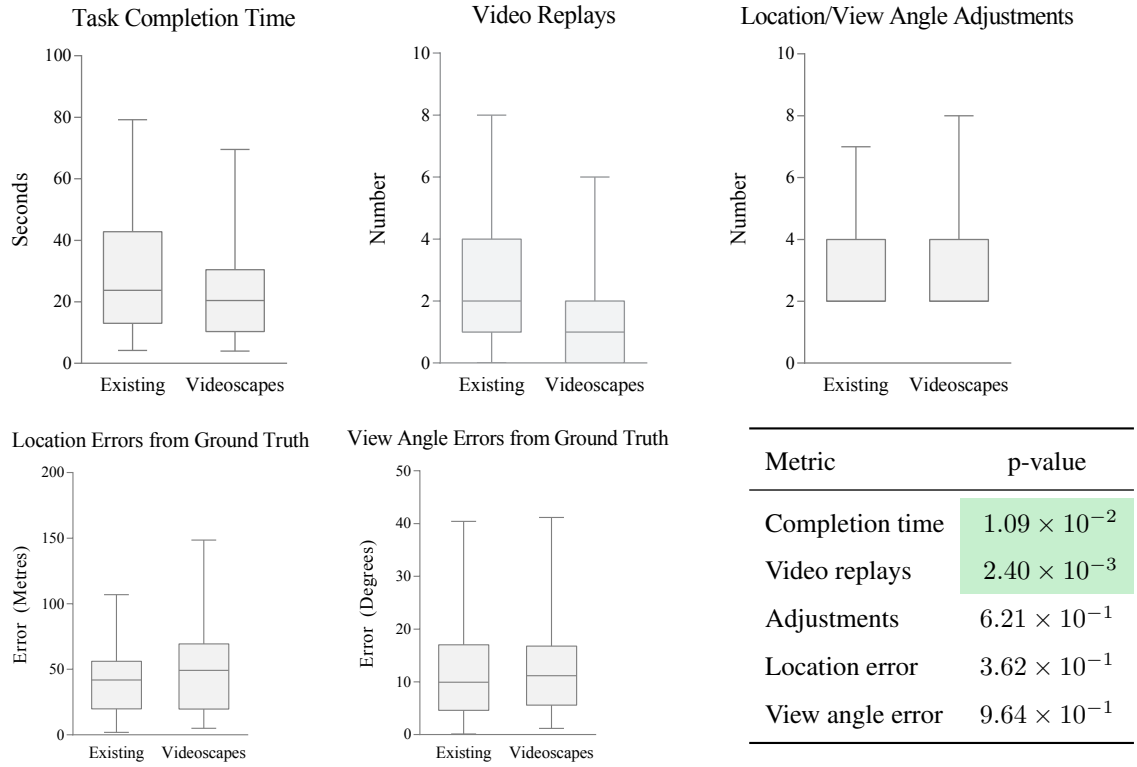
## Hypothesis

Condition 2, the Videoscapes system condition, will have a shorter task completion time because participants will maintain more spatial awareness through the transition. For the same reason, condition 2 will also cause participants to replay the video fewer times and make fewer location/angle adjustments. Again for the same reason, the error in location and view angle will be greater in condition 1.

## Results

Performance was measured with five criteria, which are presented in Figure 7.8 along with statistical significance. Some measurements in this experiment appeared to be outliers — assuming the response of participants would follow a normal distribution, we observed the frequency of values more than  $2\sigma$





**Figure 7.8:** Results for the spatial awareness experiment ('existing system' and 'Videoscapes' conditions). The alpha values for significance tests were 0.05. All values are averages over all participants.

away from the mean being higher than expected. To accommodate this, we performed outlier rejection based on robust non-linear regression [MB06]. These results are different from those published in the journal publication of this thesis [TKKT12]. The previously published results did not include the robust outlier rejection step.

Overall, both the location and view angle error from ground truth were similar between the Videoscapes and the cut/no direction conditions. Further, there were no statistically significant difference between the number of adjustments made to the view cone in both conditions. However, the Videoscapes condition produced significantly fewer video replays and so took significantly less time, which we suggest are indicators of increased spatial awareness.

While this quantitative data does not conclusively suggest an improvement in spatial awareness, our qualitative data for this experiment indicates that Videoscapes does aid in retaining spatial awareness during transitions. Following the task, each participant completed a questionnaire (Table 7.1 summarizes the results):

**Q1a:** "With which interface did you find it easiest to complete the task?"

*Condition 1 / Condition 2 / Both same*

**Q2a:** "Which interface did you find provided the greater spatial awareness and sense of orientation?"

*Condition 1 / Condition 2 / Both same*

Two further questions asked participants to rate their preference for their preferred interface (Ta-



Question	Existing	Videoscapes	Equivalent
Q1a: Which preferred for task?	1	<b>17</b>	2
Q2a: Which provides greater spatial awareness?	1	<b>19</b>	0

**Table 7.1:** Results for the spatial awareness experiment questionnaire over the ‘existing system’ and ‘Videoscapes’ conditions.

Question	Slightly	Easier/More	Much	Both same
Q1b: If one was easier, by how much?	6%	59%	35%	0%
Q2b: If one provided more, how much more?	16%	26%	58%	0%

**Table 7.2:** Results for the spatial awareness experiment questionnaire. This table shows percentages only for those participants who preferred Videoscapes. This is independent per question, with 17 of 20 participants preferring Videoscapes for Q1, and 19 of 20 preferring Videoscapes for Q2.

ble 7.2 summarizes the results):

**Q1b:** “If one interface was easier for completing the task, by how much?”

*Slightly easier / Easier / Much easier / Both same*

**Q2b:** “If one interface did provided more spatial awareness, how much more?”

*Slightly more / More / Much more / Both same*

We can see clearly that participants think the Videoscapes condition was easier for completing the task. Particularly encouraging is that the majority of all participants found that the Videoscapes condition provided much more spatial awareness.

### Comments

Of 20 participants, 7 left meaningful comments. Table 7.3 summarizes these comments. Unfortunately, there is little here to corroborate our quantitative or qualitative results.

### Outcomes

The Videoscapes condition required significantly fewer video replays and took significantly less time than the existing condition for the same accuracy. The questionnaire shows that most participants preferred our system, and almost all save one thought our system provided more spatial awareness. This correlates with our quantitative data, and demonstrates that our system helps maintain greater spatial awareness through transitions.

## 7.4.2 Video Tour Experiment

We wish to compare our generated tour results with existing methods, but these methods do not produce comparable output from comparable input. However, our video tours could be thought of as a geographical summarization of the video database, particularly for the case where the user selects start and end locations and leaves the path to be generated by our system instead of interactively navigating. As

Comment	# Participants
Know area well so task was not difficult.	4
Used semantic knowledge for Condition 1.	1
Confused by which video frame green view represented.	1
3D transition gave camera movement.	1

**Table 7.3:** Comments left by participants during the spatial awareness experiment.

such, in this experiment, each participant watches three videos which have been automatically edited by software, and these form the three conditions in our experiment:

1. An *InstantMovie* generated by Adobe Premiere Elements 7.0,
2. A video with the *intelligent fast forward* of Pongnumkul et al. [PWC08] (Section 7.2.1), and
3. A non-interactive video tour generated by Videoscapes (with blend transitions).

The first two conditions serve to sample one commercial example of video tours and one state-of-the-art research example from the map-based video browsing literature (see Sections 2.5.2 and 3.3.2 for detailed explanations of these two approaches). We do not include a hand-edited ‘ground truth’ video as we wish to see the difference between automatic techniques; however, this would be an interesting experiment for future work.

In these three conditions, we do not show effects or any interface elements<sup>3</sup> so that the videos are viewed independently of any other system functionality. Each ‘summarization’ video was generated with the same input database. For participants, videos could be replayed at will, and videos were presented in a random order. Participants were asked to concentrate on the way the content was presented (*style*), and not on any specific content. Chapter E shows images of all website interfaces and sample output.

After watching the videos, participants completed the questionnaire listed below and ranked the three styles explicitly. These questions are highly subjective — from this questionnaire, we wish to broadly qualitatively assess our tours. The questions were: “Which style...”

**Q1:** “...did you most prefer?”

**Q2:** “...did you find most interesting?”

**Q3:** “...did you find provided the best sense of place?”

**Q4:** “...did you find most spatially confusing?”

**Q5:** “...would you use most often in your own video collections?”

**Q6:** “...would you view most often for online video collections?”

<sup>3</sup>The exception being the *InstantMovie*, which contains two instances of hard-to-remove overlaid theme graphics and infrequent minor ‘effects’. Participants were explicitly asked to ignore these and concentrate only on content when considering their answers.

Method / Question	Q1	Q2	Q3	Q4	Q5	Q6
1. InstantMovie	34	37	23	55	31	31
2. Pongnumkul et al.	38	38	43	40	35	38
3. Videoscapes	<b>48</b>	<b>45</b>	<b>54</b>	<b>25</b>	<b>54</b>	<b>51</b>
2. significant vs 1.?	$4.08 \times 10^{-1}$	No	$1.03 \times 10^{-5}$	$8.26 \times 10^{-4}$	$2.58 \times 10^{-1}$	$8.23 \times 10^{-2}$
3. significant vs 1.?	$9.32 \times 10^{-3}$	No	$3.27 \times 10^{-8}$	$6.89 \times 10^{-8}$	$4.06 \times 10^{-5}$	$4.76 \times 10^{-4}$
3. significant vs 2.?	$4.80 \times 10^{-2}$	No	$7.38 \times 10^{-3}$	$8.26 \times 10^{-4}$	$5.86 \times 10^{-5}$	$3.94 \times 10^{-3}$

**Table 7.4:** Results of video summarization experiment questionnaire, showing the total score assigned to each condition across all participants. Bold figures highlight the best score for each question — for Q4, this is the lowest score, representing the style which is least spatially confusing. Alpha is set to 0.05 for all significance test.

## Hypothesis

Condition 3, the Videoscapes system condition, will be preferable to participants because the summarization will have visual continuity. Condition 3 will also be the most interesting because the graph portals, around which the summarization transitions occur, represent connected content. We posit that this connectedness can be thought of as a measure of how interesting the content is. As condition 3 will maintain visual continuity, it will also provide the best sense of place, and will be the least spatially confusing. Finally, condition 3 will be used and viewed most often as it will be preferred, more interesting, provide the best sense of place, and be the least spatially confusing.

## Results

Table 7.4 summarizes the results. Ranking scores (from 3 to 1) are accumulated over all participants. Significances were computed by the Kruskal-Wallis test and then by pairwise Mann-Whitney U-tests, with alpha at 0.05. Videoscapes is significantly the most preferred summarization style, provides the best sense of place, and is least spatially confusing of the three videos for the database used. We must take care not to extrapolate this to all databases as one example is not conclusive, but Q5 and Q6 significantly suggest that our system may be preferred most often for personal and online video collections.

## Comments

Of 20 participants, 9 left meaningful comments. Table 7.5 summarizes these comments, in which some participants discussed multiple points. Here, we receive some interesting feedback. The most frequent comment was that the speed of the intelligent fast forward was too fast. Setting this speed is difficult to judge as it is content dependent. A range of good speeds likely exists but these depend on the camera motion and action present in the video. Adjusting the fast-forward speed to camera motion is something that could be accomplished from the data we pre-process for each video. However, adjusting fast-forward speed to action within the videos is beyond the scope of this work.

Other participants commented on their expectation that the preferred style would vary depending on the content. One participant comments that “a fast-forward style may give a better sense of place in a busy London street whereas a slow, similar content style [Videoscapes condition] might be preferable

Comment	# Participants
Fast forward too fast (slightly slower is better).	4
Fast forward too slow when at normal speed.	1
Fast-forward only comprehensible due to area knowledge.	1
Fast-forward is most predictable.	1
Fast-forward useful for long videos.	1
InstantMovie gives overall sense of area.	1
InstantMovie more artistic.	1
Videoscapes transitions unpredictable.	1
Videoscapes transitions good for short videos.	1
Preferred style varies with content (extrapolation).	2
Videoscapes + fast-forward is best of both worlds.	1

**Table 7.5:** Comments left by participants during the video tour summarization experiment.

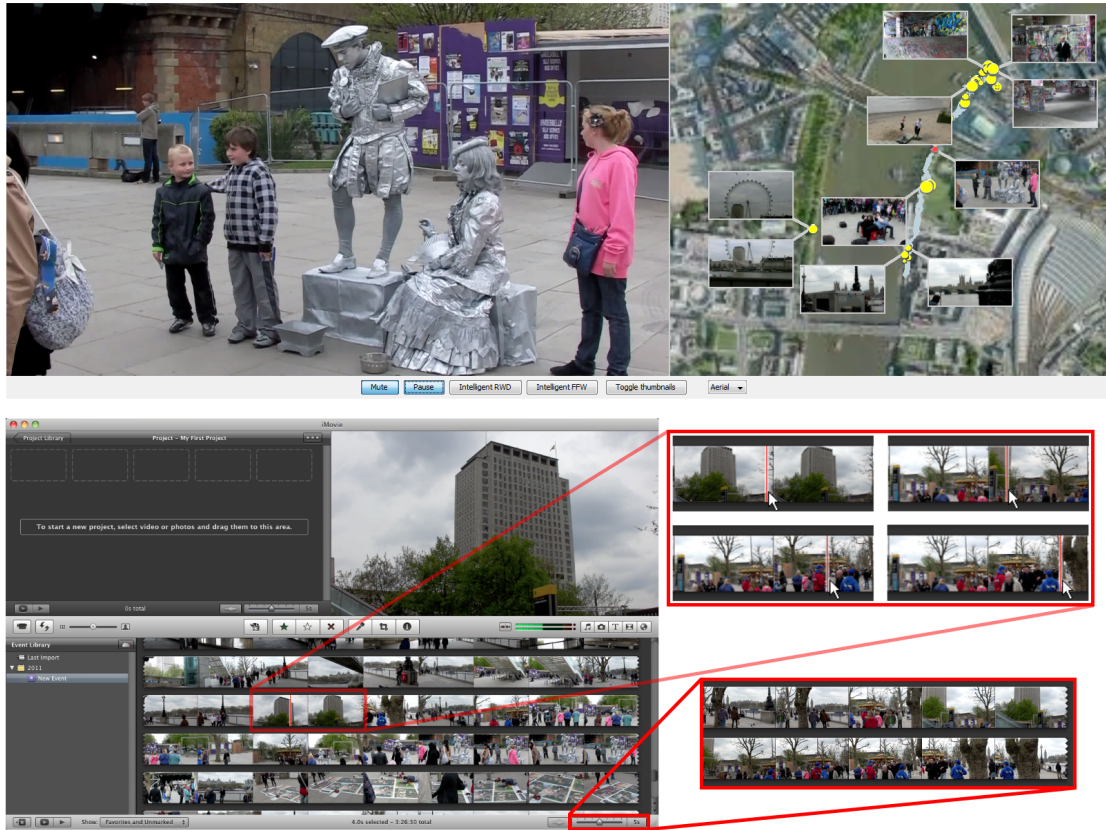
for expansive vistas or *somesuch*”, while another participant adds: “I would think that using just this example to decide which was the ‘best’ (in terms of preference, spatial confusion, etc.) might be misleading as a conclusion.” We previously aired caution that such extrapolation was unwise for this reason; however, as our interface includes more exploration tools than were visible in this experiment (where participants watch static video only), we feel our results are a promising affirmation of the utility of our system.

### Outcomes

We discover that, for the specific videos we showed to participants, the Videoscapes-generated tour was significantly preferred in many categories including ‘least spatially confusing’. While this is a promising result, we cannot reliably extrapolate to different contents as it seems likely that the preferred style of summarization will vary with content. Some content-adaptation is possible with our system concerning camera motion, but not scene action. That said, our Videoscapes interface does also include intelligent fast forward and, as these two conditions make up the top two choices for each question, we feel confident that our video tours are useful tools for collection summarization. As one participant wrote: “Summing up, the best of all worlds in my vision would be to have a joined similar content video [Videoscapes condition] to give a story line to the whole collection but with the fast forward function added on top to skip to the highlights quickly.” Our system would meet the expectations of this participant.

### 7.4.3 Video Browsing Experiment

Our final experiment attempts to evaluate the overview and fast geographical browsing map interfaces as tools for browsing and retrieving video contents. Participants were asked to find five videos among a collection which contained contents similar to an image query of a major landmark. An error was marked every time a participant accidentally found a previously found video. This video collection contained



**Figure 7.9:** Video browsing comparison interfaces. Top: Our implementation of [PWC08] adapted for video databases. Bottom: iMovie '11. Cut-outs show scrubbing and thumbnail expansion (which frequently made participants feel lost and from comments was difficult to use).

45 videos, ranging for 5 seconds to 30 minutes (mean 4m49s). 20 videos contained the target landmark, with 35 contiguous temporal regions of video frames (clips) within those videos containing the target landmark. Participants attempted to find content using three different interfaces:

1. Apple iMovie '11.
2. Our implementation of a multi-video version of Pongnumkul et al. [PWC08]. As written, their paper only supports a single video and manual geotagging. We automatically place thumbnails and vary their density on zoom so that the visualization of all interesting shots in a database is possible.
3. Videoscapes (Figures 7.3 and 7.5).

The first two conditions again serve to sample one commercial example of video collection interfaces and one state-of-the-art research example from the map-based video browsing literature (see Sections 2.5.2 and 3.3.2 for detailed explanations of these two approaches).

For this task, four browsing tools within Videoscapes were available: *image search*, *label search*, *browsing eye icons* (the second workflow, Section 7.2.2), and *geographical video browsing* (the third workflow, Section 7.2.3). Questionnaire results about these four tools are presented later in this section,

	1. iMovie	2. Pongnumkul	3. Videoscapes	2. sig. vs. 1.?	3. sig. vs. 1.?	3. sig. vs. 2.?
Q1:	27	40	<b>53</b>	$7.46 \times 10^{-3}$	$1.32 \times 10^{-6}$	$7.41 \times 10^{-3}$
Q2:	30	41	<b>49</b>	$3.30 \times 10^{-2}$	$3.01 \times 10^{-4}$	$1.21 \times 10^{-1}$

**Table 7.6:** *Videoscapes preference in video browsing results. Bold results are the greatest scoring for each question.*

while Appendix F shows images of the questionnaire website interface and sample output. In this experiment, the label database held only objective labels of specific landmarks. Before use, each interface was thoroughly explained to the satisfaction of the participant, and participants were given the option to use whichever methods they wished within each interface for completing the task. Each participant is asked to perform the same task in each interface in a random order. When assessing how similar a particular video is to the target image query, participants were asked to use their own criteria to assess the suitability.

The two main evaluation criteria were:

**T1:** The average time taken to complete the task in seconds, and

**T2:** The average number of occurrences of finding the same video clip more than once — the duplicate-find error rate.

## Hypothesis

Condition 3, the Videoscapes condition, will have shorter task completion times as the visual relationships between videos have been computationally discovered and structured into a graph, which is explored through various different interface tools. Condition 3 will also induce fewer errors in the participants because the structuring and the interface tools help prevent becoming ‘lost’ in the collection.

## Results

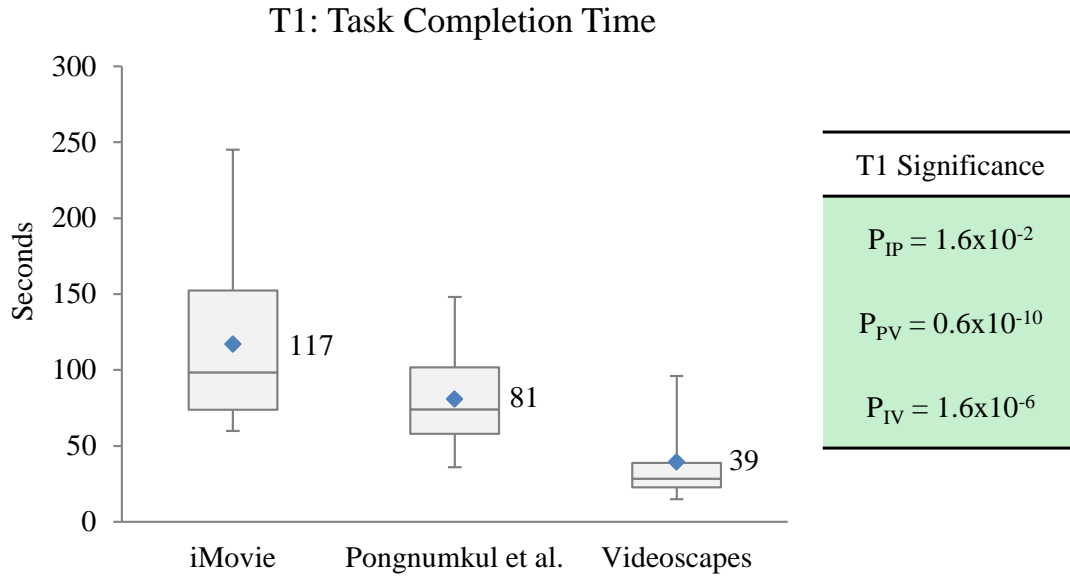
Figure 7.10 shows the completion time results and Figure 7.11 shows the error results. The significance of the results is indicated by the p-values:  $P_{IP}$ ,  $P_{PV}$ , and  $P_{IV}$  represent the pairwise p-value of iMovie and Pongnumkul et al., Pongnumkul et al. and Videoscapes, and Videoscapes and iMovie, respectively. These results indicate that the speed and accuracy of browsing is significantly improved by using Videoscapes over existing systems. We suggest that this is because our video database structuring sorts and groups similar material, meaning that video browsing is more accurate and more efficient. Our interface exposes this structure in two fast ways: image or label search, and geographically-placed visual groupings with our eyes icons.

Participants also completed a questionnaire following the task (Table 7.6 summarizes the results):

**Q1:** “Which interface did you most prefer for completing the task of finding content?”

**Q2:** “Which interface do you think you would most prefer for browsing content generally?”

These two questions mark an important distinction which was explained to participants: that an interface may be preferred just for finding specific content, but would otherwise not be preferred in a more



**Figure 7.10:** Completion time results for our video browsing experiment. Significance of results is shown by pairwise  $p$ -values, e.g.  $P_{IV}$  denotes Videoscapes significance against iMovie, where green denotes significance.

Would you use our system?	Yes, often	Yes, sometimes	Yes, rarely	No
For personal collections	6	<b>10</b>	4	0
For online collections	<b>12</b>	6	1	1

**Table 7.7:** Would participants want to use our system? Bold signifies the most frequent answer for each question.

casual map-based geographical browsing experience to, for instance, gain a sense of place. The results were computed as before, where ranking scores (from 3 to 1) are accumulated over all participants and significances are computed by the Kruskal-Wallis test and then by pairwise Mann-Whitney U-tests, with alpha at 0.05. Videoscapes is significantly preferred over both other systems in the task and significantly preferred over iMovie in the general case. Again we must not generalize beyond the experiment and database, but the responses to our interface are promising.

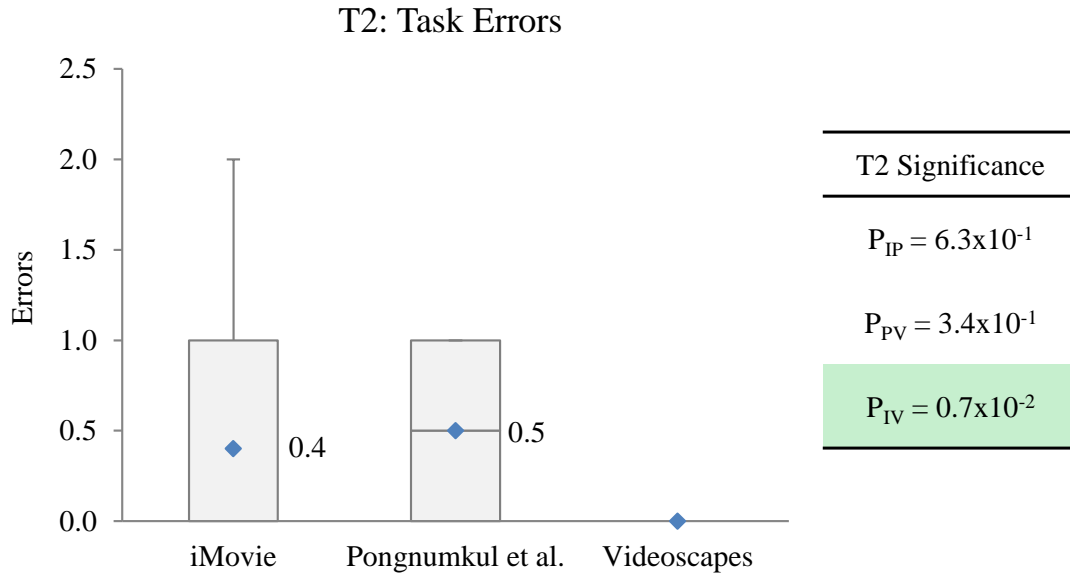
Finally, we asked participants whether they would want to use our interface for browsing personal and online video collections. The results are promising, with 95% responding that they would use it, and at least 80% responding sometimes or often (Table 7.7).

### Individual Interface Components

Further questionnaire questions focused on which interface components were preferred among the browsing methods provided by Videoscapes. Summarily, the *browsing eye icons* and *image search* methods were the two most preferred for completing the task and for the potential of browsing video collections in general.

We asked participants 10 questions. The questions were: “For the Videoscapes interface, how useful...”





**Figure 7.11:** Error results for video browsing experiment. Significance of results is shown by pairwise p-values, e.g.  $P_{IV}$  denotes Videoscapes significance against iMovie, where green denotes significance.

**Q1a:** “...were the portal eyes?”

**Q2a:** “...were gray trails showing from where the video was taken?”

**Q3a:** “...were the white camera field of views showing from where the video was taken?”

**Q4a:** “...was the search box for text searches?”

**Q5a:** “...was the search box for image searches?”

and “In general, how useful do you think...”

**Q1b:** “...the portal eyes would be for browsing video collections?”

**Q2b:** “...the gray trails would be for browsing video collections?”

**Q3b:** “...the white camera field of views would be for browsing video collections?”

**Q4b:** “...the search box for text searches would be for browsing video collections?”

**Q5b:** “...the search box for image searches would be for browsing video collections?”

Table 7.8 summarizes the results. Participants found that interfaces provided by Videoscapes are broadly ‘useful’, even if they didn’t use them for the task. The portal eyes and image-based searches were especially preferred for the task, while the text search and camera frusta features were not as preferred for the task. However, even though they were not used in the task, most participants regarded them as useful features for general video browsing systems. This suggests that our prototype interface has some merit beyond the specific task of the experiment.

## Comments

All 20 participants left meaningful comments for each interface. Tables 7.9, 7.10 and 7.11 summarize these comments for the three interfaces.

Question / Utility	Very useful	Somewhat useful	Not useful	Did not use
Q1a: Portal eyes for task	<b>14</b>	0	1	5
Q2a: Grey trails for task	7	4	1	<b>8</b>
Q3a: View frusta for task	6	6	1	<b>7</b>
Q4a: Text search for task	6	4	0	<b>10</b>
Q5a: Image search for task	<b>11</b>	1	0	8
Q1b: Portal eyes in general	<b>13</b>	7	0	0
Q2b: Grey trails in general	6	<b>13</b>	1	0
Q3b: View frusta in general	<b>10</b>	<b>10</b>	0	0
Q4b: Text search in general	<b>12</b>	7	1	0
Q5b: Image search in general	<b>16</b>	2	2	0

**Table 7.8:** Further questionnaire results of video browsing experiments showing the number of participants who responded for each choice. Bold signifies the most frequent answer for each question.

**iMovie** (Table 7.9) From the comments left, participants felt that it was good that the iMovie interface provided a summary of the whole collection at once when zoomed out very far. Otherwise, most of the comments were negative. Many participants felt that searching for content was slow, or that the zoom tool was difficult to use or to set in such a way that it made searching as easy as possible (though this is content dependent). From the comments, it seems that there was only one search strategy used by participants to browse the collection: to expand the clips with the zoom tool to a suitable granularity, and to scroll through all clips in turn. Two participants also commented that they introduced local knowledge as a fast rejection method.

**Our Pongnumkul Implementation** (Table 7.10) With our implementation of a multi-video version of Pongnumkul et al. [PWC08], participants found that the geographic layout was helpful in finding content, and that the interface was generally easy to use and that the thumbnails helped. Participants did not like that the interface provided no view direction information for videos as this made orientation difficult. Participants also found it difficult to distinguish between different videos when many were in the same location, as the number of thumbnails visible is limited by the screen space. From the comments, there was one predominant search strategy shared by participants: first, to use the thumbnails to find one example of the target image, then to use the corresponding yellow dot thumbnail pin as a geographical marker to find similar videos shot from a similar location. Some participants also integrated their local knowledge as a fast rejection scheme.

**Videoscapes** (Table 7.11) Participants were overwhelmingly complimentary about our Videoscapes interface, with over half of participants commenting positively on the search functions. Participants also found our portal eyes convenient, and the interface fast and clear in general. Both negative comments related to the grey trails, with one participant commenting that they didn't work as expected because the polyline trail sampling was not sufficiently high for per-frame seeking, and another suggesting that

iMovie Positive Comments	# Participants
Collection summary (i.e., view at minimum zoom) is good.	3
Zoom tool is good.	2
Familiar interface.	2
Mouse-over scrubbing is good.	1
Looks nice.	1
Is OK.	1
Total	10
iMovie Negative Comments	# Participants
No search tool / Slow to search manually / Linear search only.	8
Zoom tool is difficult to use or to set correctly / Thumbnails not representative.	7
Interface is unresponsive.	3
No inter-video relation.	3
Thumbnails too small.	1
Difficult to see how viewpoint will change.	1
No clear delineation between separate videos.	1
Total	24

**Table 7.9:** Comments left by participants about the iMovie interface during the video browsing experiment.

they were not useful without knowledge of the video content. Participants commented that they used three methods to complete the task (image search, text search, and portal eyes) and this correlates with our per-feature questionnaire results. One participant commented that they would use these advanced features first and then, if they failed, resort to a more dependable interface (e.g., iMovie).

**General Comments** Finally, participants were allowed to leave general comments about anything they wished. While many participants repeated praise for our Videoscapes interface (10 comments), some participants asked for further features or suggested problems that might arise. One participant asked for a sketch-based image search system. This is an active research field in its own right (see [CCT<sup>+</sup>09] for a recent creative example), but this is beyond the scope of this work. Another participant suggested that our interface was only useful for browsing, and that editing and montaging clips from a video collection would require additional tools. We agree wholeheartedly, and suggest that our system could be used as a first step to sort through video collections to find suitable footage.

Addressing limitations, one participant mused that it was hard to say how useful the text search would be without knowing the quality of the labels. This is true: our propagation allows labels to be attached to the correct object across the collection, but not necessarily for the label to be semantically

Pongnumkul Positive Comments	# Participants
Geography/map helpful.	6
Easy to use.	4
Thumbnails useful.	3
Fast.	2
Easy to pick out view point/location.	1
Thumbnails easier than view point/location	1
Very nice.	1
Total	18

Pongnumkul Negative Comments	# Participants
No knowledge of view direction difficult/confusing.	5
Hard to distinguish dense videos/thumbnails.	3
‘Interesting’ thumbnails not useful/a distraction.	2
No rewind.	2
No scrubbing.	2
No image search.	1
Need to know video capture location.	1
Thumbnail density variation difficult on zoom.	1
Requires imagination of map terrain.	1
Thumbnail click / map panning difficulty.	1
Total	19

**Table 7.10:** Comments left by participants about our implementation of the Pongnumkul [PWC08] interface during the video browsing experiment.

meaningful. Finally, the question of the size of the video collection that our interface can represent was raised. Each portal eye can realistically supporting only tens of views before it becomes tedious or overwhelming. Likewise, we can only draw so many grey trails onto the map before it is saturated. Our search tools can support more as the results are presented in a list. While there are simple ways to fix many of these problems, our system was designed to cope with video collections of hundreds of videos only. Moving to Internet-scale collections would require further thought and the use of scalable techniques across the whole pipeline. We discuss this more in Section 8.4, as this significant challenge is beyond the scope of this work.

## Outcomes

We discover that, for the London video collection that we showed to participants, the Videoscapes interface was significantly faster at finding target content than either iMovie or our implementation of

Videoscapes Positive Comments	# Participants
Search is fast / easy / useful.	11
Eyes are convenient / fast.	6
Interface well suited / good / fast / clear, etc.	5
Geography / mapping better than Pongnumkul.	2
View frusta more useful than Pongnumkul yellow dots.	2
View frusta easy.	2
View frusta accurate.	1
Grey trail scrubbing useful.	1
Many options to find content.	1
Easy to identify different videos.	1
Total	32

Videoscapes Negative Comments	# Participants
Scrubbing grey trails didn't work.	1
Grey lines not useful without knowledge of content.	1
Total	2

**Table 7.11:** Comments left by participants about the Videoscapes interface during the video browsing experiment.

Pongnumkul et al. [PWC08]. We find that our Pongnumkul et al. implementation was also significantly faster than iMovie. When it comes to errors in finding content, specifically accidentally finding the same content twice, we find our interface caused significantly fewer errors than iMovie. Qualitatively, our participants significantly preferred our Videoscapes interface over the two other interfaces for the task of finding content. When asked to extrapolate and consider which interface they would prefer for browsing content in general, participants significantly preferred our Videoscapes interface over iMovie. When questioned, participants expressed strong interest in using our system for browsing both personal and online video collections. On the whole, participants found all of our interface features useful for the task and for browsing in general. Finally, all these points are corroborated by comments left by participants, which were largely negative for iMovie, were mixed for our Pongnumkul et al. implementation, and were almost all positive for our Videoscapes interface.

## 7.5 Summary

In this chapter we have looked at different interfaces for exploring video collections. Having reviewed interfaces in Chapter 3, and briefly recapped at the beginning of this chapter, we presented a new interface specifically for exploring video collections. Our interface contains three workflows: one for interactively navigating video collections that only requires video data (no position/orientation data),

one for geographically navigating content similarities within the collection, and one for fast geographical browsing of the collection. We also provide text and image search functions and label propagation within and between videos.

We tested our interface against existing solutions to see whether it provides benefits to spatial awareness, to see whether the generated video tours compare well to existing solutions, and to evaluate our video browsing interface and see whether it is faster and more accurate at finding content. For spatial awareness, we found that providing geometry-based video transitions and providing view frusta instead of pins on a map significantly decreased the orientation time and so increased transition comprehension. Qualitatively, participants preferred our interface and found it provided greater spatial awareness. When assessing our generated video tours against two existing alternatives, participants preferred our tours, found they provided the best sense of space, found them least spatially confusing, and would use them most often for personal and online collections. All of these results were significant. Finally, we found that our interface was significantly faster than two existing interfaces for finding content within a video collection, and in doing so produced significantly less errors than one of these existing interfaces. Participants provided feedback on individual elements of our interface and found them all useful. 95% of participants stated that they would use our interface for personal and online video collections, and qualitatively participants provided many more positive comments about our interface than about the comparison interfaces. All these points together strongly suggest that our interface provides a significant improvement over existing interfaces for exploring video collections.

## Chapter 8

# Discussion

Sparse, unstructured video collections of places present many challenges to providing useful interfaces to the content, and we believe our system rises to many of these with convincing results. Such a broad problem space requires significant investigation, and many challenges remain.

## 8.1 Portal Finding

### Sensor Data and Map Embedding

For the overview mode, our system optionally uses GPS and orientation information to embed the Videoscape into a map. Automatic embedding of the videos into a map if GPS is not available may be feasible by using metadata [TLRA03, KN08] or geolocation from the video itself by matching against image databases [LWZ<sup>+</sup>08, BKC<sup>+</sup>10, ZS10, KWO10], but this is left as future work. For the core of our method, we intentionally only use the video frames for maximum generality. We incorporate GPS and orientation information into the filtering phase of portal finding, but we have not yet extended this into the matching phase. However, as GPS data is often unreliable in cities, and as we allow large view changes (e.g., zooms), integrating this data is not trivial. Orientation data may be more useful to rule out incorrect portals in the matching phase. One simple sanity check is to see whether feature-based matches also have intersecting camera frusta.

### Portal Selection

Our criteria for portal selection, as described in Section 5.5 in Equation 5.7, chooses portals with good feature matches and with a non-disorientating transition between videos by preferring small displacements between matched features. However, other choices are possible; with the data that we compute during portal finding, we know approximately the motion of the camera at every frame. Hence, a range of options are possible and would be simple to add depending on the desired application. For instance:

- A score which prefers large displacements between matched features to show as much variety of viewpoints onto the content as possible within the video collection.
- A score which assesses the motion of the camera before and after the portal to provide various effects:
  - Only selecting portals when videos are still.



- Only selecting portals when both videos show complementary pans, zooms, and world motions.
- Only selecting portals which do not suffer camera shake in either video.

Our geometry reconstruction approach can support all of these cases as it exploits the Videoscape graph to robustly estimate the geometry without relying on either of the constituent videos.

Portal selection which depends on dynamic objects, such as a score which penalizes transitioning when dynamic objects are present, is more difficult. It might be possible to compare the recovered geometry against the video frame to isolate dynamic objects (which typically do not appear in the reconstructions); however, we leave this for future work (see also Section 8.2).

### Graph Complexity

When finding portals, we intentionally only select the best portals for each pair of video clips (see Section 5.5) to keep the number of portals down. Here, recall that each video file is masked into 30 second clips to ensure that there are frequent portal opportunities along each video. While the visualization of the Videoscape graph to the user is independent of the underlying complexity, a densely connected graph provides the viewer with too many transition possibilities and hinders effective exploration. For instance, using *all* identified portals is conceptually possible, but would produce a graph with potentially thousands of nodes and hundreds of thousands of edges. Selecting only the highest-scoring portals reduces this number to a level of connectivity which can be directly displayed without overwhelming the user.

Even though we attempt to select portals whose frames are largely aligned, by minimizing the screen-space distance of feature points within candidate frames, the best portals may still exhibit large displacements which lead to transitions with wide camera motions. From our transition experiment, we see that this is not a problem: by exploiting the Videoscape graph we can reconstruct convincing scene geometry and provide convincing transitions in considerable view change cases.

### Portal Clustering

Often many similar portals are discovered — this is especially true if videos linger on particular objects or events for minutes at a time. Video frames which look very similar but only differ in time often include minor differences due to dynamic objects which stops them being identified as contributing to the same portal. For example, in our South Bank database, this is particularly prevalent for a set of videos which linger on a group of acrobats for 5 minutes. To solve this problem without affecting existing parts of the pipeline, we could post-process cluster the found best portals based on a less discriminative portal selection score, such as one based on image context or ‘gist’ features [OT06].

## 8.2 Transitions

### Quality/Collection Size Trade-offs

The quality of our geometric reconstructions is limited by the available views of the scene within the video collection. Increasing the size of the video collection would increase the number of views of a scene, and so help to improve the quality of 3D reconstruction. Of course, increasing the size of the data

set has its own drawbacks. There is scope to speed up our processing, and recent work [FGG<sup>+</sup>10] has demonstrated speed improvements with images. Coupled with more aggressive filtering, this approach should enable a much larger video set to be processed in a similar amount of time.

### Camera Shake

For some hand-held footage, obtaining camera tracking is challenging, especially if rolling shutter artefacts occur as well. In these cases, we can still use *full 3D* — *static* transitions, as our interpolated virtual camera provides convincing camera motion style blending despite inaccurate camera tracks. This is justified, as our participants preferred the static 3D transitions over other transition types in this scenario. Importantly, 3D geometry is still recovered in these difficult cases because the support set of the portal provides sufficient context.

### Empty Areas in Transitions

In our transition user study in Section 6.3, we discovered that participants dislike empty regions appearing in transitions, where the virtual camera path is interpolated into regions that have no content due to converging and diverging pans and contrasting zooms. Without affecting the current virtual camera path, we could extend the geometry into these empty regions by increasing the number of neighbourhood matches used to generate the portal support sets (Section 5.5.1). However, this approach will fail if the empty regions are simply not present in the video collection or if the portal geometry fails to reconstruct. Even then, these areas of the image will appear as textured geometry rather than video and will not contain dynamic objects.

With the camera pose knowledge discovered in the video-to-geometry registration (Section A.7.2), it is possible to predict the appearance of empty areas given our analysis in Section 6.3.7. Future work should investigate correcting the currently linearly interpolated path to minimize empty areas, though this will result in more complicated camera motions with faster accelerations which might be disconcerting to the viewer.

### Foreground Objects

By design, our proposed method does not model foreground objects. In both portal identification and 3D reconstruction, foreground objects are regarded as outliers in matching and accordingly are ignored. This can sometimes introduce distracting artefacts: some objects warp or vanish during the transition. For example, pedestrians may warp into each other. This is mitigated when using spatio-temporally coherent exploration (Section 7.2.1) when temporally aligned video data is available (Section 5.6.2). Another option is to specifically look for portal frames that are devoid of foreground objects. This could work well for certain data sets such as nature landscapes, but again is unrealistic for anything but sparsely populated environments.

If foreground objects could be reliably segmented, then there is an opportunity to remove them before a transition occurs (by dissolving against an inpainting), and then to dissolve in the new dynamic objects in the second video after the transition. However, video inpainting is unreliable and computationally expensive, and is another source of potential artefacts. Alternatively, with good video-to-geometry

registration and high-quality reconstructions, we could replace the dynamic object regions with textured geometry; however, the end result will not look like video and will not exhibit world effects such as variable illumination. As our dissolve strategy is supported by evidence from perceptual studies, we believe it is preferable currently.

## 8.3 Interfaces

### Geographical Geometry Reconstructions

For portals, it may be possible to automatically directly embed geometry onto a map or globe. This would provide additional context information to the viewer and would help strengthen visual relationships in the interface between map and video views. Recent work by Crandall et al. [COSH11] has showed promising results with databases of geotagged photographs. However, due to the sparse and unstructured nature of capture in our video collection, this would not always be possible. In preliminary experiments, we could not reliably embed geometry within our globe by the GPS coordinates of video frames as the recovered and real-world camera poses were too inaccurate: either the GPS position was incorrect, or the recovered pose was incorrect.

### Interactive Mode Video Inlays

Expressing spatial information for a portal choice in our interactive navigation mode is challenging. The mini-map shows frusta and paths when hovering over portal thumbnails to show to where the video choice will move, but one might think to express this within the view of the camera. We believe that this is a difficult problem, and is not an appropriate interface for a video collection as the content can quickly change with camera movement: Portal images by definition look similar, so placing them into the current view (equivalent to [SSS06]) tells the user very little about what will happen in each candidate video path beyond the portal. Each of these video choices has its own independent camera motion and may move to look at entirely different things in a few seconds. Choosing between different views of the current scene is limited if you do not know what the camera will look at next. Instead, our interface shows directly in the thumbnails what will be seen next if that path is chosen, and all content along each edge can be accessed very quickly by scrubbing the thumbnail.

Embedding different views of the current scene into the view at portal choices has other problems related to graph connectivity. What if there are 10+ videos connected, each moving away in different directions? We believe our approach is fit for the purpose of exploring video collections; in general, this is a challenging visualization problem which we would like to address further in future work.

### Overview Mode Path Planning

Our current implementation for automatically generating video tours around the graph is limited. The viewer chooses a start and end point, and we find the shortest path between the two graph nodes. The viewer is always able to manually construct a video tour by clicking in turn the portal eyes that they wish to be included; however, more advanced automatic methods are desirable. Our current ideas for future work include:

- Generating tours of a specific length.
- Generating tours with specific inclusion/exclusion portals.
- Generating tours with specific camera motions, such as only those captured on foot or from vehicles, by better analysing the camera motions we compute in the filtering step.
- Generating tours which take paths through time, not just temporally-consistent paths. For instance, to see how some particular landmark or plaza varies over time within the video collection.

## Audio

This work does not make any contributions for dealing with audio in video collections. While it is trivial to cross-fade soundtracks across transitions (indeed, we do this), this may not be the best method for all videos and transition types. If the content of a video contains speech, then it may be appropriate to delay transitions until the end of the speech. Improvements in spatial awareness may also come from spatially positioned audio. For instance, during a large camera sweep transition, the sound could attenuate realistically as the virtual camera moves away from sound sources. Sound in tours or summarizations also brings another element we do not address: narration. Video collections of places could be augmented with appropriate narration from online sources such as Wikipedia. A system that resolved both constraints on video sequences and on audio narration would allow a greater number of applications to benefit from automatic tour generation.

## 8.4 Databases

### Temporal Coherence

Our results use databases that require only loose temporal coherence; other databases may require this functionality more strongly. Videos taken during an event, such as our simulated sports event ‘campus bike’ database example (Section 4.6), may require a temporally consistent exploration of the Videoscape, else, e.g., the position of the leader may suddenly change. Many other databases do not require temporal consistency, and novel experiences may come from intentionally disabling temporal consistency. Our system is sufficiently general to accommodate these scenarios.

### Failure Databases

We tested our system on a database created from high-definition television broadcast footage of a Formula 1 race. We isolated all in-car vehicle footage captured from on-board cameras (facing both in front and behind the car). Then, we segmented from the frame only the region of the image which identified the track and its environment, to crop the static view of the car and any broadcast graphics from the shot. Unfortunately, for these camera placements, the remaining image regions were very thin  $500 \times 100$  sections of road and race-track barriers. Our system completely failed to find correspondence in this situation, partly due to the motion blur and partly due to the fact that, at car level, one corner of a race track looks like any other to a local feature detector. Strong features are present in the images, but these are usually in advertising hoardings which, as expected, are identical in many places around the track.

We postulate that our system would fare better if given the raw camera feeds without broadcast graphics, and if we picked a city race such as Monaco instead of a track race.

We also tested our system on three park databases, and our system had trouble finding stable features across videos. Parks are difficult test cases, with many similar features in the flora and fauna of each shot and, with the wind, many moving shadow lines and so moving image features. However, these problems are not specific to our system, and represent difficult test cases for current computer vision algorithms.

### Web Video Collections

Finally, the question arises as to whether our system could be used on community video databases. We cannot directly apply our system to Web video collections as there are numerous problems which require solutions before this is possible, including how to find content, how to scale massively to cope with millions of videos, and how to present these graphs to the user once they have been structured.

### Finding Content

Web video collections are vast and contain many different classes of videos, most of which our system makes no attempt to handle. The first problem when dealing with Web video collections is to find relevant content, and this is a major research field in its own right (Section 2.4). We explain in Section 4.5 when defining the scope of our system that, in preliminary experiments, we could not easily or automatically find sufficient appropriate videos of a location for our system in current community databases as the signal-to-noise ratio was too low — this is in stark contrast with community photo databases. Perhaps: 1) currently, online databases do not contain sufficient suitable videos for our system, which we believe unlikely, but which would be corrected over time as more videos are added; or 2) online databases do contain such videos, but they are very difficult to find as current commercial search technologies are based on key-word or whole-video label associations and not on visual content or geographical features. Further research into video-based content retrieval will help this situation, but robustly and efficiently finding specific video content in massive collections is a difficult problem.

Other smaller matters cause complications even once appropriate videos have been found: Often the resolution of videos is very low, or the video is heavily compressed. These make feature matching more difficult and less reliable. Furthermore, many online videos contain editing and overlays, making automatic cut detection and logo/overlay detection necessary. There are existing solutions for many of these problems, and they could be integrated into the preprocessing stages of the Videoscapes system.

### Internet Scale

Currently, our system can deal with hundreds of videos; but web video collections contain hundreds of millions of videos. Section 4.5 touches on this problem, and we are approximately comparable to the state-of-the-art in handling this many videos. If we suppose that an appropriate subset of videos has already been found, it is still likely to contain upwards of tens of thousands of videos of a place, and this is more than an order of magnitude larger than what we can currently handle. Comparing to existing methods from the literature, our system uses similar approaches and algorithms to existing state-of-the-art works [ASS<sup>+</sup>09, FGG<sup>+</sup>10] and so performs approximately comparably (< 5x) given leeway for the

focus on speed and engineering efforts of these works.

However, Internet scale not only means massive data but also massive processing. Most of our pipeline is parallelized — if we imagine applying our approach to cloud computing environments with thousands of processors, with further work it does not seem so far-fetched to handle this many videos in the near future. Still, the larger problem is in discovering an appropriate initial subset of videos.

### Interface Scale

Currently, our interfaces are designed to show the connections present within a few hundred videos. Given tens of thousands of videos of a place, many of our interface elements might have to change. Our work is still immediately applicable to some extent because we provide image- and label-based search modes. However, more radical changes are required, and here there are broadly two options:

1. To keep similar interfaces to those we have described, and to focus on filtering and prioritizing content for display within these interfaces. We already exploit the graph to perform some choice filtering operations, but there is scope for more advanced analysis. For example, to prioritize clips with lots of dynamic objects or clips which are uncommon among the collection. Additionally, social aspects of Web video collections might be integrated to prioritize the results of automatic recommender systems or the recommendations of friends.
2. To redesign the interfaces to allow more connections to be quickly accessible. Massive graphs are a well-studied problem in visualization fields [Cyt12], but we believe these solutions must be married with an immersive video viewing experience which helps maintain spatial relationships for collections of places. This is a large and complicated problem which is a thesis in its own right.

From the comments of our experiment participants, new and efficient interfaces for massive media-collection navigation are definitely desired and, like the solution we present in this thesis, will likely lie at the convergence of many computer science disciplines.

### Alternative Collection Sources

Our system is not required to work on Web video collections to be useful, as video can be captured specifically for the purpose of exploration with the Videoscapes system. It is easy to imagine a Videoscapes-like system which uses a smartphone application to handle data collection and cloud processing to structure the graph, and our motivating examples (Section 1) and experimental results on our hand-collected databases (Section 7.4.3) provide compelling evidence that this use of our system is desired.

## Chapter 9

# Conclusion

This thesis has described the development of an end-to-end system for exploring sparse, unstructured video collections of places or events. In reviewing the literature, we find that existing systems cannot provide interfaces for such video collections which expose the implicit visual and geographical relationships. We devise a set of recommendations for any new system which attempts to solve this problem, then design the *Videoscapes* system to meet these recommendations.

Videoscapes automatically finds content relationships between videos by comparing video frames with a series of increasingly robust matching methods. We form a graph from these content matches, where edges are videos and nodes are possible transition points, or *portals*, between videos with similar content. To provide spatial awareness when travelling through portals, we exploit the context information present in the graph to build geometry reconstructions of the environment at portals. This enables dynamic 3D transitions when switching between videos. Videoscapes provide novel interfaces to navigate the recovered graph which overcome the problems associated with extending existing map-based video systems to video collections. These include the baseline interactive interface, our map-based overview interface, geographical summarization tours, and image- and label-based searches and tours.

To validate our system, we performed 4 experiments:

1. We assess transition types for preference, and provide perceptual-scale rankings across different scenes and view changes. We perform an artefact analysis which, when coupled with user feedback, leads to heuristics for automatic transition type choice and for ordering artefact preference.
2. We quantitatively and qualitatively assess spatial awareness through transitions with a map-based task. While our quantitative results suggest only mild improvement with transitions, our qualitative results strongly indicate improvement.
3. We assess our summarization tours against existing summarization alternatives for qualitative measures such as how interesting the tour is and how much of a sense of place it provides. We find for most questions asked that our approach is significantly preferred.
4. We present a video browsing task to users to gauge both the effectiveness of and preference for our interface elements. We find that image search and portal eye functions are preferred, but that all our interface elements were perceived as broadly useful. When compared to existing solutions



for exploring video collections for content, we find Videoscapes both significantly quantitatively and significantly qualitatively preferred. We confirm that users are interested in using our system for personal and online video collections.

Finally, we discuss the limitations of our system and its potential application to Web video collections. In completing this work, we have demonstrated that it is possible to provide compelling new interfaces for video collections of places by automatically exploiting implicit visual and geographical relationships.

# **Appendices**

## Appendix A

# Transition Types in Detail

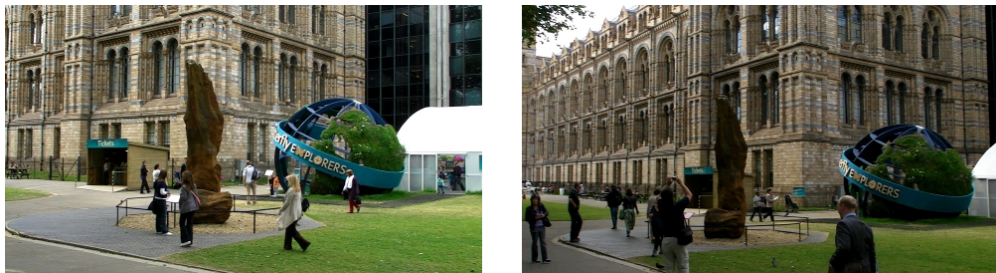
This appendix fully explains the transition types used in our Videoscapes system and in our perceptual experiment in Chapter 6. For each transition, the corresponding section contains a historical review of its application, our technical method to achieve the transition, and a description of artefacts that may appear in the transition. Following these transition explanations, a further subsection explains techniques and issues in common between transitions, including how 3D scene geometry is recovered and how the Videoscape aids in this reconstruction. Finally, we collate and categorize all feature and artefact types in each transition in Table A.1.

## A.1 Cut

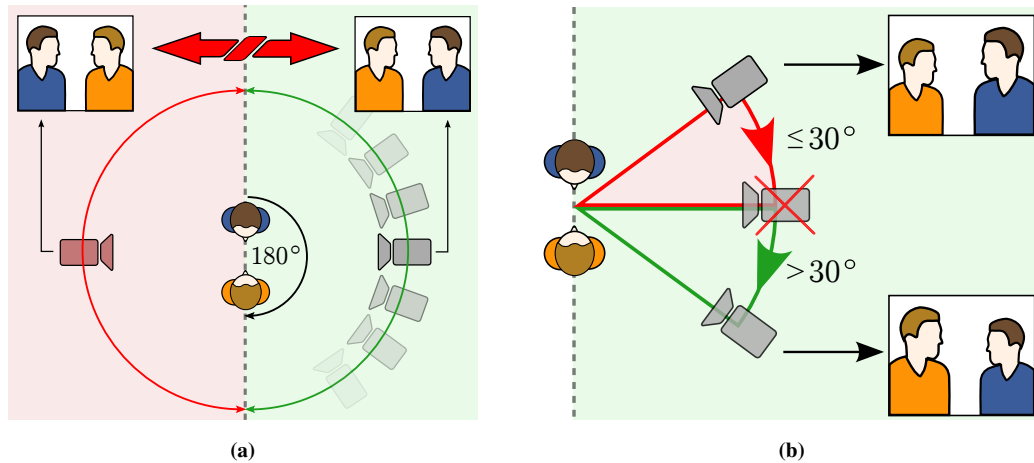
### Background

A cut in film or video occurs when a frame from one shot is immediately followed by a frame from another shot. The phrase describes the manual process of cutting a strip of film with a blade and joining it to another strip with adhesive. Since the birth of cinema a language of cuts has developed describing how to invoke a response (or not) in the audience. In our case, the audience might expect this language to be respected when cutting between shots in a video collection. Figure A.1 demonstrates a slight view change cut transition.

The personal cutting rules of Dmytryk and Murch, formed from many years of experience in editing, are described in their respective books [Dmy84, Mur01]. Rules referring to emotion and story are beyond the scope of this thesis as they are difficult to formalize. However, there are some rules which are easier



**Figure A.1:** An example of two portal frames which constitute a jump cut. We can see a slight camera position and rotation change and a temporal change.



**Figure A.2:** a) The 180° rule. Cameras should not cross the dotted line when cutting between objects. b) The 30° rule. Cameras should be at least 30° apart relative to the scene when cutting. a) courtesy of [Wik11]; b) created from a).

to consider (McCurdy provides a summary [McC07, p.100]). The first is the 180° rule: once established, the frame location of characters or scene elements should not be mirrored during a cut. This rule most often relates to characters in conversation. When a cut is made, the position of the camera should not move between half-spaces formed by a ground-perpendicular plane that intersects both characters. Both our slight and considerable view change conditions enforce the 180° rule.

The second is the 30° rule: the view angle change of the camera to a scene during a cut should be at least 30°. This rule aims to enforce that a cut provides a significantly different view of the scene. If the rule is broken, we achieve a *jump cut*. Jump cuts often propel the viewer forwards in time, as the observed scene does not change significantly. If there is no angle change, and the camera position moves (or zooms) along a line towards or away from the subject, we achieve an *axial cut*. Jump and axial cuts create abrupt changes, and are often used to unnerve an audience. Often, the more conservative a filming and editing style, the larger the view angle change threshold in this rule [Fri03, p.37]. The slight view change condition of our chosen scenes most closely reflects both jump and axial cuts, whereas our considerable view change condition represents cuts which do not break the 30° rule. Figure A.2 visualizes both the 30° and 180° rules.

## Technical Method

A cut is relatively simple to implement. Modern PCs can easily decompress two HD video streams in real time; assuming both streams are synchronized as per the portal frame and are decoding in tandem, the cut is simply a swap of the source of the output buffer from one clip to the next.

Videoscapes renders all imagery in OpenGL for speed and flexibility, and so there are further complications. The decoded video frame must be uploaded to the GPU to be used as a texture. Pixel buffer objects allow DMA transfers from main memory and prevent synchronous locking against rendering operations, but require decoding to happen one frame before uploading. This introduces a one-frame



**Figure A.3:** *These two portal frames break the  $180^\circ$  cutting rule, as they are taken from opposite sides of Big Ben. SIFT features have been matched and verified on the tower even though the surrounding buildings are different. Yellow circles in the left frame are matched to green crosses in the right frame. Green crosses in the left frame show the relative offset from the original position of the yellow circle feature point in the left frame.*

delay over the CPU case, and so decoding must start one frame in advance of the cut. Hence, the cut is a two-frame operation.

Modern CPUs and GPUs allow decoding of certain video codecs in specialised hardware, which provides the fastest way to decode video on a PC. Videoscapes allows graphs of videos to be directed or undirected, so videos must play forwards and backwards. Codecs with fast bi-directional playback (such as MJPEG) are not supported by decoding hardware, which typically only accelerates MPEG4-class standards. To compensate, we decode the video into a monochrome YUV frame on the CPU, and then upload it to the GPU for colour space transformation. As we need the video frame on GPU memory for rendering (especially for the plane, ambient point cloud, and 3D transitions which use projective texturing), this is faster than allowing all decoding to occur on the CPU.

### Artefacts

Given that the aim of Videoscapes transitions is to link sequences while providing a sense of orientation, the major artefact of a cut transition may be that this sense is lost. In movies, this is not a problem as the space of action rarely plays a part in the story. However, for video collections of places where the goal is to maintain spatial awareness, this may not be the case.

Cuts work well for transitioning between videos where the two frames were taken from very similar world positions, i.e., when corresponding feature points have very small image displacements. However, symmetric buildings may break the  $180^\circ$  rule. If there is not sufficient context between portal frames to disambiguate the sides of symmetric buildings, the position of the camera may move between half-spaces (Figure A.3). Not only does this cause immediate visual discontinuity as peripheral buildings change positions within the frame, but it also disorientates the viewer of a virtual tour as the view has changed significantly geographically. However, this is uncommon as precision is the priority in portal matching (Table 5.2).

The  $30^\circ$  rule is commonly broken in Videoscapes, as portal finding expressly looks for images with little visual difference. As such, many typical cut transitions between portal frames will be jump or axial cuts (see Figure A.1). It would be possible to reject portal candidates with view change angles of less than  $30^\circ$  should the Videoscape creator wish to use cut transitions. However, this is not explored as we



**Figure A.4:** A frame in the middle of a dissolve transition (bottom) between two portal frames (top left and right).

focus on situations where a seamless transition provides orientation continuity.

## A.2 Dissolve

### Background

A dissolve merges between two shots gradually over time (Figure A.4). The effect was first created by the controlled double exposure of film, and later by using optical printers to project two negatives together and form a new exposure. Dissolves are now more commonly created by digitally interpolating intensity values in frames across time. The dissolve was often used to suggest the passage of time to an audience, or a change of place [Dmy84, ch.13]. Similarly, a fade to and from black was used when longer periods of time were suggested. Dissolves retain some visual link across sequences and were thought to be important to maintaining continuity in pre-1960s cinema. French ‘New Wave’ cinema in the 1960s showed that audiences did not need dissolves to demonstrate that time had elapsed, and that cuts served the same purpose (even jump cuts) [Wik10]. Since then, dissolves are used less often in this way, and are now more commonly employed as a special effect to soften a transition for aesthetic purposes (either for very similar content [Lin06, last transition in movie] or for when juxtaposition is undesirable [Ste51]); however, longer time period changes (“two years later”) and montages still often employ dissolve transitions. For our purpose, a dissolve can be both aesthetic and suggestive of a change of time. It softens the visual transition (which as a cut may be unnerving), and signifies a change in time as the source videos may have been shot at arbitrary times.

## Technical Method

Videoscapes performs dissolves by simple linear interpolation of RGB values across time. This is often known as a ‘video dissolve’ in some video editing software suites. Formally, for start and end portal frame images  $I_i^S, I_j^E$  respectively, at frame numbers  $i, j$  in their respective clips, with odd transition length  $t$  and transition frame index  $k = (1, 2, \dots, t)$ , new transition frame  $I_k^T$  equals:

$$\begin{aligned} ik &= i - \lceil \frac{t}{2} \rceil + k, \\ jk &= j - \lceil \frac{t}{2} \rceil + k, \\ I_k^T &= (1 - \alpha)I_{ik}^S + \alpha I_{jk}^E, \end{aligned} \tag{A.1}$$

where  $ik$  and  $jk$  are the frame indices for transition frame  $k$  in the start and end clips respectively, and  $\alpha = \frac{k}{t+1}$ . Should either video be playing backwards, we must add  $\lceil \frac{t}{2} \rceil - k$  to the portal frame index instead.

Video editing software may include other dissolve methods, such as ‘film dissolve’. These aim to better reproduce the effects of double exposure, or in some cases, a double negative exposure as would be used in an optical printer. Typically, these dissolves linearize the input video by radiometrically calibrating the camera, removing the characteristic response curve of the camera, interpolating, and then reapplying the response curve. This kind of dissolve is especially appropriate with high-dynamic range data, where the video is already linear. As such, these approaches are sometimes called ‘light linear dissolves’.

As the appearance differences between video and film dissolves is relatively small when compared with the differences between dissolves and the other presented transition modes, Videoscapes does not perform the extra calibration and computation required for film dissolves. However, film dissolves do prevent unnecessary darkening during the blend should this be a problem.

## Artefacts

Dissolve transitions, like cuts, are comfortable to watch. However, when interpreted literally they contain many artefacts: objects ghost and merge into one another. This is especially noticeable in dynamic objects which, when coupled with camera motion, regularly disappear into buildings and roads in transitions. In staged productions these artefacts can be planned away, but in arbitrary video collections these dynamic objects are difficult and costly to handle [MO09].

## A.3 Warp

### Background

Warping describes any image transformation that distorts shapes, and usually refers to non-linear local geometric operations which cannot be described by affine or projective transformations. Image warping pre-dates digital image editing: since the Renaissance artists have used anamorphosis to include warped elements in paintings, and mathematical biologist D’Arcy Thompson used image warping to describe natural variation in creatures [Tho17]. Digital image warping came to public prominence through the late 1980 and early 1990s as *morphing* (an operation where one object appears to transform into another)



in feature films such as *Willow* [How88] and music videos such as Michael Jackson’s *Black and White* [BN92].

There are many ways to warp an image, but they generally rely on finding correspondences between images, interpolating image shapes on an underlying 2D geometric representation (such as a grid or a triangulation), and dissolving image appearance across time. If we only had to deal with camera rotations, we could estimate a homography and warp with a projective transform; however, this is not the case and we must deal with parallax from differing camera positions. Still, the problem is not as difficult as arbitrary camera motions as portal frames often have similar visual content at similar 2D positions in the image.

Modern warping methods are numerous as there are many ways to find image correspondences [Low04, SRB10, LLN<sup>+</sup>10] and perform interpolation [IMH05, SMW06, LLB<sup>+</sup>10]. For instance, the SIFT flow work of Liu et al. [LYT<sup>+</sup>08] can robustly find dense matches between views of considerable difference; however, here this is not our only goal. Dynamic objects in the scene must be effectively ignored in the correspondence and must not create holes, and so existing dense correspondence approaches are not necessarily applicable. Furthermore, these approaches are typically very computationally expensive. For Videoscapes, as we have many hundreds or thousands of portals, any warping method must run in a short amount of time so that either preprocessing or real-time display is feasible.

### Technical Method

We start the warp transition implementation explanation by first describing how to warp between the two portal frames only. We automatically discover image correspondences by extracting SIFT feature points and matching their descriptors [Low04]. From these correspondences, we robustly estimate a fundamental matrix with the normalized eight-point algorithm [HZ04] and RANSAC [FB81]. This leaves only those image correspondences which conform to estimated good camera poses and removes correspondences on dynamic objects.

With these correspondences, we can warp between portal frames with an interpolation scheme — we choose moving least squares (MLS) image warping [SMW06]. This technique reconstructs a continuous warping function between the feature points using an underlying grid. This grid warping can be constrained by one of three schemes: as affine as possible, as similar as possible (no shear or non-uniform scaling), and as rigid as possible (no uniform scaling either). It might seem that an as-affine-as-possible approach is most suitable, given that, of the three, it best approximates a perspective transformation (the perspective transformation is the true image transformation for locally planar scene surfaces). However, in practice, an as-similar-as-possible warp is more robust: areas of the image that do not contain many feature points, such as sky, sometimes scale or shear inappropriately under an as-affine-as-possible warp. Also, as-rigid-as-possible warps only allow rotation and translation changes, making them unsuitable for the scale changes we have between portal frames.

After executing an as-similar-as-possible moving least squares warp, we obtain a per-pixel vector field describing the necessary pixel movement from one portal frame to the other. We compute this field bi-directionally. To generate a transition sequence which moves from source to target, we synthesize



**Figure A.5:** A frame in the middle of a warp transition (middle) between two portal frames (top left and right). This is the first stage of the warp transition effect, where the two videos are approximately aligned.

frames by blending fractional interpolations of the source-to-target and target-to-source warps. Formally, for start and end portal frame images  $I_i^S, I_j^E$  respectively, at frame numbers  $i, j$  in their respective clips, with transition length  $t$  and transition frame index  $k = (1, 2, \dots, t)$ , and 2D vector fields  $V_{source \rightarrow target}$ , new transition frame  $I_k^T$  equals:

$$I_k^T = (1 - \alpha) \text{invmap}(I_i^S, \alpha V_{I_i^S \rightarrow I_j^T}) + \alpha \text{invmap}(I_j^T, (1 - \alpha) V_{I_j^T \rightarrow I_i^S}), \quad (\text{A.2})$$

where  $\alpha = \frac{k}{t+1}$  and  $\text{invmap}(I, V)$  applies an inverse (or backward/reverse) mapping. This allows us to dissolve between registered portal frames, and to synthesize new frames which move the view from one portal frame to the other (Figure A.5).

Now that we can generate a still frame warp transition, next we need to generate a video warp transition. We use the same vector fields from source to target portal frames, but instead wish to warp with pixels from frames surrounding the portal frames. To do this, we must find vector fields for each frame of each video clip to the corresponding portal frame. We first find KLT feature points [TK91, ST94] over small windows in time around each portal frame, and robustly reject outliers on dynamic objects as before [FB81, HZ04]. Rejecting feature points on dynamic objects is not just to help find a robust warp; it is an essential part of the technique. We do not wish to inadvertently remove motions from dynamic objects and accidentally freeze our video. We find feature-point tracks which run through all relevant frames, and use these to compute a per-frame vector field by an as-similar-as-possible MLS warp. This registers the video frame to the transition portal frame warp interpolation, but still keeps the individual motions of dynamic objects. It is unnecessary to generate warped images from individual

video frames to the portal frame; we only need chain the vector fields together to pick the relevant pixels for the transition from the individual video frames. With this registration, we can now keep playing the video after the source portal frames and leading up to the target portal frame.

Formally, for notation as in Equation A.2, new transition frame  $I_k^T$  equals:

$$I_k^T = (1 - \alpha) \text{invmap}(I_{ik}^S, \alpha \text{lutbi}(V_{I_i^S \rightarrow I_{ik}^S}, V_{I_i^S \rightarrow I_j^T})) + \alpha \text{invmap}(I_{jk}^T, (1 - \alpha) \text{lutbi}(V_{I_j^T \rightarrow I_{jk}^T}, V_{I_j^T \rightarrow I_i^S})) \quad (\text{A.3})$$

where  $ik$  and  $jk$  are as in Equation A.1 and  $\text{lutbi}(V_1, V_2)$  bilinearly looks up  $V_1$  with an index derived from the  $(x, y)$  pixel location plus the vector offset from  $V_2$ .

Choosing the number of correspondences to use in the warp is important. As the two videos exhibit parallax, the more features the better to have ghost-free alignment on static objects when we finally blend. However, the cost of the MLS warp increases quadratically with the number of correspondences, so a balance must be struck. The true required number depends on scene complexity, but we found that 250 correspondences was sufficient for our scenes and left only small amounts of ghosting between the frames from the warped start and end clips.

As the image differences are now slight at this stage as the images are approximately registered, we can introduce an additional step and use dense optical flow to remove the ghosting on static objects in a similar way to Eisemann et al. [EDM<sup>+</sup>08]. Flow vectors are computed between the two approximately registered images for each frame. The flow vectors are then interpolated across the transition such that ghosting on static objects is removed throughout the transition (see Figure A.6). As we interpolate the warp and flow vector contributions from source to target across the whole transition, this provides a pseudo-3D effect where the scene appears to exhibit parallax.

Formally, for notation as in Equation A.3, and  $W_k^S$  denoting a warped frame from the start clip for transition frame  $k$ , new transition frame  $I_k^T$  equals:

$$\begin{aligned} W_k^S &= \text{invmap}(I_{ik}^S, \alpha \text{lutbi}(V_{I_i^S \rightarrow I_{ik}^S}, V_{I_i^S \rightarrow I_j^T})), \\ W_k^E &= \text{invmap}(I_{jk}^T, (1 - \alpha) \text{lutbi}(V_{I_j^T \rightarrow I_{jk}^T}, V_{I_j^T \rightarrow I_i^S})), \\ I_k^T &= (1 - \alpha) \text{invmap}(W_k^S, \alpha V_{W_k^S \rightarrow W_k^E}) \\ &\quad + \alpha \text{invmap}(W_k^E, (1 - \alpha) V_{W_k^E \rightarrow W_k^S}), \end{aligned} \quad (\text{A.4})$$

where the vector fields between  $W_k^S$  and  $W_k^E$  are formed by optical flow [BBP04].

Finally, the two now-registered video clips must be composited and dissolved together. Image regions in the new synthesized image where both clips contribute are given full brightness, and other regions where only one video contributes are slowly faded out (or in) over the transition down to a minimum contribution (set to 75% brightness). Edges between the contribution regions are feathered to mask sharp boundaries.

### Artefacts

Warp transitions suffer the same ghosting of dynamic objects that are present in the dissolve transition, but crucially not the ghosting of static objects as our two videos are registered. This dynamic object ghosting causes birds, pedestrians, and road traffic to merge into one another or background buildings.



**Figure A.6:** *The middle frame of a warp transition (top), this time with warp correction. Bottom left shows a zoomed (2.5x) version of a window region without flow correction, and bottom right shows the same region with the flow correction. This region is clearly better, but other regions (where the distance is too great for the flow to correct, or for pixels near boundary regions) are arguably worse under careful frame-by-frame observation. However, the effect in motion with the flow correct is much improved as ghosting on static objects is greatly reduced, and this provides a 3D effect. Section A.3 contains artefact discussion and example images.*

If feature point correspondences are incorrect then frames from each clip will not align. This is uncommon as we use robust outlier rejection, but it is still possible. The artefacts produced by MLS warping range from small swimming to large swirls in the worst case (Figure A.7). Generally, any correspondence mismatch causes a significant visibly distracting artefact that moves irregularly to the transition camera movement; an accurate dense correspondence field would minimize this error by maximally constraining neighbouring pixels.

Our anti-ghosting optical flow post-process for static objects has problems at edge boundaries, as the flow is undefined here. This causes flickering at frame boundaries (Figure A.7). Optical flow also has problems with dynamic scene elements that are present in one video but not in the other (such as people); or where scene elements have moved significantly within the image plane. Here, the flow flickers over time as it is undefined. This kind of effect, where some image features appear in one frame but not the other, will always be difficult for flow- and warp-based methods to deal with, though recent work furthers identification of such occlusions [HMB11]. We reduce the flicker by temporally smoothing the flow field with a 5-wide Gaussian kernel.

Unfortunately, these flicker artefacts often appear within the contrast sensitivity temporal frequency peak range of 5 to 10Hz [Wan95, Figure 7.23-A, p. 223], and so are easy to notice. However, the





**Figure A.7:** *Artefacts in the warp transitions, taken as zoomed regions of Figure A.6. Left: Swirl artefacts from incorrect feature point correspondence. Here, a feature to the left of the door has been matched with a feature to the right, causing the warp to generate a swirl. Right: Flickering at the boundary between images, where the flow between the two overlaid images is undefined.*

flow-based ghosting correction significantly improves the overall quality of the result and is much less computationally expensive than computing wide-baseline dense correspondences from the beginning. For instance, the state-of-the-art result by Lipski et al. [LLN<sup>+</sup>10] takes many hours to find and optimize correspondence, and would require dynamic object detection to prevent freezing dynamic motions. One way to reduce flicker is to explicitly inpaint missing regions at boundaries using context-aware fill methods [BSFG09, BSGF10], though this has large caveats which will cause other artefacts such as no temporal consistency and incorrectly inpainted structure causing other flow artefacts.

While warp transitions do interpolate camera velocities implicitly by interpolating feature point tracks, they do not maintain accelerations in to and out of the transition. When either video clip is under motion at the point of transition there is a noticeable change of speed. This could be solved by setting  $\alpha$  to a curve which matched the accelerations.

## A.4 Plane

### Background

A plane transition approximates all scene geometry as a plane. The plane is fitted to the scene using recovered 3D feature points that are common to both the start and end views [MGL06, SSS06]. While primitive, this method works well for many scenes with limited depth, with scenes that are very distant, and with views undergoing slight changes. It provides a good indication of the 2D or 3D camera motion between views, and is commonly used in online photo tourism sites [Mic08, Goo08].

Early research into image-based graphics used cylinders or spheres to reproject omnidirectional imagery [Che95, MB95], but these require pixel-accurate correspondence to perform transitions. Scenes with a single vanishing point, such as views looking down a street, are better approximated by cubes with tour-into-the-picture techniques [HAA97, KS02]. In these techniques, objects at contrasting depths, such as cars or pedestrians, are modelled by hand as individual plane billboards (or, when representing 3D geometry, as *impostors*), and have long been used in computer graphics [Sch95]. Recent work has automated the billboard modelling of a single dynamic object at the centre of converging viewpoints for

video-based viewpoint interpolation [BBPP10], or has asked the user to draw silhouettes to define depth boundaries within a recovered point cloud [CSD11].

### Technical Method

We choose to use the approach of Snavely et al. [SSS06, Sections 4.1 and 4.2] and forego the challenge of modelling individual dynamic objects as billboards at their own depths (a task requiring good segmentation that has yet to be robustly achieved in the literature for our kind of data). For pairs of images, we do not add to this technique and so only outline it here; we refer the reader to their paper and to the included references for further details. However, as we shall see the technique is not directly applicable to video transitions.

Snavely et al. [SSS06] first find SIFT feature points and candidate correspondences between images [Low04], then robustly find a fundamental matrix using RANSAC [FB81] and the eight point algorithm [HZ04]. They then find connected points, or *tracks*, of matching feature points across images, and optimize the 3D location of each track by the error in its reprojection using the Levenberg-Marquardt solver [NW99]. Each camera (and the tracks to which its feature points belong) is added iteratively to the reconstruction, with each addition followed by a sparse bundle adjustment optimization of all camera and 3D point parameters [LA04]. This process produces a set of camera poses, a set of 3D points, and a mapping between each camera and the points which it observed.

Next, they estimate a common plane between the two views [SSS06, Sections 5.1 and 5.2]. This plane is the best fitting plane in the least-squares sense to the union set of the points observed in both views, and is estimated robustly using RANSAC. The transition is created by projecting each photo onto the plane from its respective view, dissolving one photo into the next as in a still two-frame version of Equation A.1, and interpolating a novel camera between the two views from which to render.

Adapting this approach to video is not as simple as it may seem. We describe our approach separately in Section A.7.2, as this registration of both videos to geometry is shared by other transitions. The result of this process is that both videos are registered to the proxy geometry in a temporally consistent way, without jitter.

When it comes to rendering, there are only minor differences extending Snavely et al. [SSS06] to video having obtained our good video-to-video-to-geometry registration. One choice we might make is whether to estimate a new plane per frame. This would require having different 3D feature points from the tracked KLT points per frame. As we only use one set of recovered 3D feature points to optimize the video frame positions (i.e., the Snavely et al. [SSS06] recovered 3D feature points), we cannot produce *different* planes per frame. If we re-fitted a plane each frame, based on currently visible 3D feature points, then there would be temporal differences due to the RANSAC process that would cause edge flickering where the projections end.

Finally, our transition is created by projecting each frame of video from the two clips onto the plane, dissolving one frame into the next as in Equation A.1, and interpolating a novel camera between the two views from which to render (Section A.7.3). Figure A.8 shows an example frame from slight and considerable view plane transitions.



**Figure A.8:** Top left: *Slight view change start frame*. Top middle: *Considerable view change start frame*. Top right: *End frame for both transitions*. Bottom left: *Middle frame of a plane transition for the above slight view change. Ghosting is visible on static objects due to camera translation*. Bottom right: *Middle frame of a plane transition for the above considerable view change. Ghosting on static objects is visible as before, but the large translation and rotation causes buildings to appear skewed*.



**Figure A.9:** Left: *Ghosting on static objects such as Big Ben*. Right: *Significant skewing in a considerable view change transition*.

## Artefacts

Plane transitions suffers the same ghosting situation as warp transitions: static objects are registered, but dynamic objects are not. This causes birds, pedestrians and road traffic to merge into one another or background buildings. However, in considerable view change cases where the baseline is wide and the camera undergoes a large rotation, static objects are often no longer correctly registered in plane transitions. This is because a plane is often a crude approximation to the real scene geometry, which creates artefacts such as static object ghosting as well as dynamic object ghosting. These manifest as parallax or shear artefacts causing buildings to noticeably lean across wider view change transitions (Figure A.9). This type of artefact is typical of plane transitions where the geometry proxy does not well represent the scene.

However, the plane transition is commonly used because its artefacts are less objectionable than those in other transitions. For instance, Snavely et al. [SSS06] compare it to triangulated morphs and find it preferable due to robustness. Vangorp et al. [VCL<sup>+</sup>11] study parallax effects caused by inaccurate



geometry (among other IBR-related artefacts) and find them harder to spot (page 9, Section 7.2, paragraph 4): “...when there is no ground truth to compare against, subjects may be *unaware* that they are misperceiving the scene, and thus do not find the errors disturbing.” While they test with narrow angle changes, and continue to say that “it is thus best to avoid novel camera positions which result in oblique viewing angles with respect to the captured images”, this result suggests that these artefacts will not be as objectionable in our slight view change case.

## A.5 Ambient Point Clouds

### Background

Ambient Point Clouds (APC) is a recent technique developed by Goesele et al. [GAF<sup>+</sup>10] to provide motion cues during transitions. It builds upon earlier work in producing geometric models through multi-view stereo from community image databases [GSC<sup>+</sup>07]. In the APC paper, the authors harness the fact that some scene parts will not be recovered at all in multi-view stereo reconstruction (such as transparent, reflective or dynamic objects), and so the resulting processed geometry will contain holes. Coupled with that, fast transitions which have a large view change or wide baseline make it difficult for the viewer to gauge the motion of the virtual camera. APC tries to solve both of these problems by plausibly filling holes in a way that provides motion cues over time.

### Technical Method

The method is simple enough to summarize here, but for technical details, please refer to the original paper [GAF<sup>+</sup>10]. APC starts by computing the minimum and maximum depths of any recovered geometry (Section A.7.1) in the two views between which to transition. For each pixel in each view, APC generates points at random positions between the minimum and maximum depths along the ray through the centre of projection of the camera and the pixel. Typically, five points are generated along each ray, with the colour of each point taken from the respective pixel in the image. When the virtual camera interpolates between the two views, the APC is drawn in the empty spaces between the recovered geometry. The points in the cloud splay out in the direction opposite to the camera motion, and so provide strong motion cues to the virtual camera direction of motion.

Points along the ray are perturbed very slightly by random offsets in  $x$  and  $y$  to reduce aliasing and moiré-like patterns at the beginning and end of the transition (when the point cloud *almost* represents the original image). A plane (Section A.4) is rendered at the very beginning and end of the transition to smooth the introduction of the point cloud. Also, the point cloud is not used for very short transitions; here, a plane is used exclusively.

Converting this approach to video is not as simple as it may appear. One might expect to generate an APC for each video frame of each view in the transition, and to switch per frame between the APCs as the video for each view progresses during the transition. This would be the natural extension to video; however, the positions of the points in the cloud would shift at each frame and the naive implementation would contain jittering artefacts. Any video-based APC needs to generate temporally consistent results that do not introduce further artefacts.



**Figure A.10:** Left: *Transition start frame*. Middle: *APC rendering*. The two APCs from each video can be seen creating streaks at different angles. Right: *Transition end frame*.

Computation is also a problem. For a 1080p HD image, the creation of the approximately 10 million ambient points for a single image takes approximately 1 second per view on a single Intel Core i5 24.GHz core in my implementation. Simple rendering is also expensive as we maximally draw over 20 million points per view (two 1080p video frames with each pixel accounting for 5 points in the cloud). For a video transition 30 frames in length, we would need 30 seconds of computation, and this is far too long for a real-time system. Pre-computing the point cloud for all video frames and loading the data would also incur a high cost as 9.3GB of uncompressed data would need to pass from disk to GPU.

Instead, we choose to generate two APCs for only the start and end frames of the transition, and retain the same APCs for each video frame. While this isn't 'APC for video', we do not have to address the potential problem of temporal aliasing such as jitter and flickering. It also means that we have much more manageable computation requirements: in our implementation, a background thread generates the two APCs ahead of time in anticipation of the transition, and no pre-computation is necessary. For rendering, we have two options. On a powerful machine, we can brute-force render from vertex buffer objects all points at full resolution and maintain real-time frame rates. On more modest hardware such as a laptop, we perform two simplifications. First, we downsample the resolution of the APC, generating points for every 4 or 16 pixels and scaling the size of the point appropriately. Second, we render only those points in the cloud which do not lie on rays which intersect geometry. This occlusion sometimes causes view-dependent loss of density during the middle of the transition but it still maintains most of the streaking motion cues and hole fill in.

The final rendering is created as in the original paper, except the recovered geometry is projected with registered video (Section A.7.2) instead of static images. Figure A.10 shows an example transition.

### Artefacts

Ambient Point Clouds is an additive technique which aims to fill holes in recovered geometry at render time, rather than filling holes as a geometric model post-process. It also aims to provide motion cues to the viewer for wide baseline transitions. Thus, artefacts caused by incorrect geometry or inaccurate video registration are not specifically APC artefacts. However, APC and our video repurposing of APC do have their own set of artefacts.

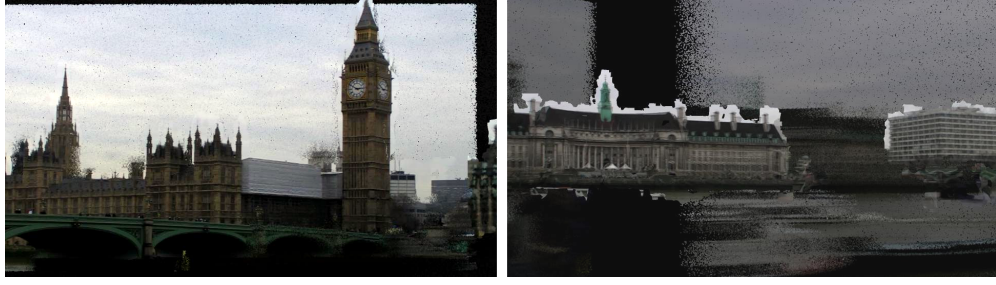
Goesele et al. [GAF<sup>+</sup>10, Section 7, paragraph 2] state that:

...if the virtual camera center does not lie on the line between the original two camera centers, i.e., if we allow for more general camera motion, the streaks from the two cameras may intersect at visible angles, diluting the illusion of coherent 3D motion.

We directly interpolate camera positions to create the virtual camera motion (Section A.7.3). While this is the same camera interpolation method as used in the APC paper for photographs, it does not produce motion along a line in world space in our video case. This is because we sequentially pairwise interpolate between a sequence of camera positions representing video frames from two clips. Our virtual camera motions lie along arbitrary paths in world space. While they may lie along a line, they also often deviate due to camera shake and movement in the start and end video clips (Figure A.17). This is one price we pay for not implementing ‘APC for video’; however, in practice, we do not often encounter these streak intersection artefacts: Large changes in camera position in each individual video are required during the blended portions of the start and end clips to generate the deviations from the theoretical line necessary to cause these *visible angle* streak intersection effects. As our clips are captured at slow speeds on foot, there is very little parallax during the (commonly) 30 or 60 frames of a video used during a transition, and so visible angle streak intersection has not been a major problem. This may be a problem if the clips were captured on a car or aeroplane, and may be more visually distracting if the parallax occurs perpendicular to the horizon.

Another artefact from not implementing ‘APC for video’ is areas of virtual transition with no content (see Section A.7.4). APC attempts to fill these empty holes, but the situation is worse with video. As we implement only a single APC per start and end video clip, it is possible for one or both clips to undergo rotations during the transition. These rotations are interpolated into the virtual camera poses, causing the virtual view to rotate away from the area of world space covered by the intersecting APCs. This most frequently causes a vertical or horizontal strip of black at the edges of the virtual image. Figure A.11 demonstrates this undesirable effect. APC does not guarantee a rendered image with no empty areas; however, our specific implementation (or lack thereof of ‘video APC’) makes these effects more common for start and end clips under rotation. In the worst case, our APCs do not intersect at all, and this rare case can also be seen in Figure A.11. This has occurred because both start and end clips were undergoing rotations in opposite directions: the portal frames in the middle of the transition matched, but the frames where the transition starts and ends, and from which the APCs are generated, share no common scene elements. Thus, their APCs do not intersect, and we see large separations of empty space in the final transition rendering. Still, even in this case, APC provides motion cues.

A more minor artefact is speckle noise over time. As the APC begins to streak in the virtual view, very small black holes appear in the background where no point sample is visible. These holes appear and disappear per frame as the virtual view moves, and become less noticeable during the middle of the transition where large motion is visible. The temporal speckle noise reappears as the end APC streaks come to reform the end clip of the transition. Pepper noise is also visible at the edges of the APC. This is where the APC is less sampled in depth due to view dependence, and so many more of these holes are visible (see Figure A.11, *right*). Finally, often an image is formed within the APC during slight view changes, and this image has the appearance of a noisy plane. This sometimes causes a double image effect where the geometry is not quite in line with the image formed from a slightly different viewpoint within the APC.



**Figure A.11:** Left: Black strips to the top and left caused by using a single APC for all frames of the end clip during a transition. With an APC for each frame, this effect would be less pronounced. Recovered geometry without supporting APC can be seen within the black strip to the right. Right: The rare case where the generated APCs do not intersect at all. The black strip running through the image is the separation between the two APCs. Motion cues are still provided by the visible streaking, but holes in geometry are unsuccessfully filled. The pepper noise at APC edges is clearly visible in this screenshot.

## A.6 Full 3D

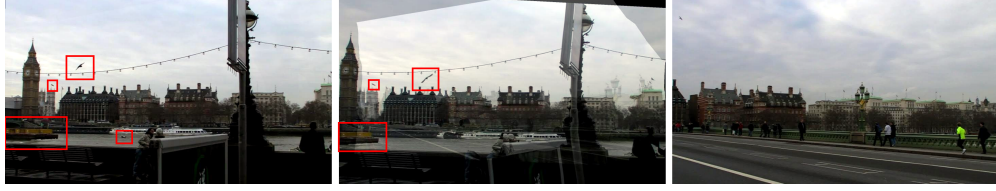
### Background

Full 3D transitions are closest to movie-style computer-generated visual effects transition, where detailed near-field geometry is layered and composited with painted or rendered backgrounds set on planes. Such compositions have now been used in movies for 30 years. Ideally, a full 3D transition would have accurate geometry for each pixel in the rendered virtual view. For movies, this would be expensive as geometry is often created by hand, and so distant objects are replaced with planes. For our system, automatic geometry recovery methods cannot currently recover full scene geometry (Section A.7.1), and certain areas such as the sky will likely always need special treatment.

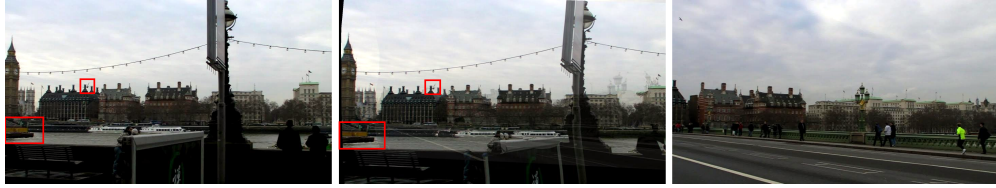
In our full 3D transitions, we use recovered geometry for all pixels for which it is available, and then approximate geometry for every remaining pixel. As in APC, we start with the recovered and processed geometry (Section A.7.1) which fills some portion of the screen in the virtual transition view. Typically, the sky is not recovered, nor are ground pixels perpendicular to the horizon, such as roads and pavements, and so we are often left with the upright portions of buildings, static vehicles, and street furniture. We approximate the real depth so that we can provide a crisp continuous projection, even if the depth is wrong, onto proxy geometry.

### Technical Method

We use planes as proxy geometry: we place one sky plane just behind all existing geometry, and one ground plane below all existing geometry. In this experiment, we added these planes by hand, but this is easily automated: First fit a plane as in the *plane* transition to the 3D feature points that intersect both cameras. Translate this plane into the scene until it is the farthest piece of geometry from both portal cameras. This plane acts as a proxy surface for the sky. Then, fit a second plane perpendicular to the first which extends from the lowest extracted 3D feature point on the first plane to a point below the farthest centre of projection of any cameras used in the transition. This plane acts as a proxy for any missing ground geometry.



**Figure A.12:** Left: *Transition start frame*. Middle: *Full 3D dynamic rendering transition frame*. Right: *Transition end frame*. Objects outlined in red are dynamic and have moved by the middle frame: two birds in flight (one disappears out of view), a boat on the river, and flying flags.



**Figure A.13:** Left: *Transition start frame*. Middle: *Full 3D static rendering transition frame*. Right: *Transition end frame*. Objects outlined in red are dynamic objects (a bird and a boat) that should and have moved by the middle frame but have not in the frozen time static rendering.

We separate our full 3D transitions into two types: *dynamic*, where the registered video projection (Section A.7.2) continues across the transition and scene objects move; and *static*, where the video projection pauses and scene objects do not move.

*Dynamic* is as other transitions (Figure A.12). The two videos projected from both sets of registered cameras are blended onto the geometry and the weights in the blend for each video as the transition progresses are as in Equation A.1.

*Static* transitions are useful when either of the start or end clips undergo significant camera shake. In this case, video registration to the recovered geometry is often bad and would cause visible shake even if the virtual camera path were very smooth. The videos are paused and only show one frame each during the transition (Figure A.13). These frames are blended with the same weights as in the dynamic case. The effect is that the world appears to stop moving during the transition. The static case is included to see whether, for one particular transition, participants prefer this freeze-frame transition style. The static case is also included to see whether participants prefer the freeze-frame style specifically when video registration is bad and causes artefacts in the dynamic full 3D transition.

### Artefacts

Full 3D transitions suffer all artefacts that exist in the recovered geometry, such as errors caused by specular, transparent/translucent, or moving surfaces; however, projective texturing usually hides the majority of these artefacts. Full 3D transitions suffer the same artefacts as all dissolve-type transitions as the projected video is blended across the transition, where dynamic objects mysteriously fade out of view or merge into one another. Dynamic full 3D transitions suffer these artefacts while the dynamic objects continue to move, whereas static full 3D transitions have these objects frozen in time as they dissolve into one another.





**Figure A.14:** Top: Two frames from a full 3D dynamic transition which contain artefacts caused by incorrect geometry. Bottom: Zoomed sections highlighting specific issues. From left to right: multiple geometry features (three/four on the tower, then three on the dome) and inaccurate dome geometry with sky recovered as geometry causing a halo effect; split pedestrian, where his head and shoulders are projected onto scene geometry and his torso onto the ground plane; double projection of a runner, again one onto the scene geometry and one onto the ground plane.

Full 3D transitions also suffer the same artefacts as plane transitions in those portions of the image that are not filled by recovered geometry. For instance, wide-angle plane transitions of buildings will show significant skew of straight lines where the geometry is incorrect; however, this is usually not a problem for full 3D transitions because these buildings typically have accurate recovered geometry. Broadly, we only have these problems in the sky and on the ground. This is not a problem for sky regions as these areas are filled with featureless regions or pseudo-random cloud patterns — dissolving these regions is rarely objectionable. However, it is more of a problem for the ground, as the added plane can be an inaccurate proxy if there is height variation on the ground. In our database, this caused artefacts when one video clip in the transition was on a bridge, and the ground has two or more heights.

Likewise, if the video registration and/or the geometry is inaccurate, then the projected videos do not line up with the geometry and the projection may stray onto the added planes, causing further ghosting. This isn't a problem in APC, for instance, because even though the point cloud acts as a projection surface, the cloud has no identifiable shape. However, it is a problem in the full 3D case. In the worst case, multiple examples of scene features can exist: for example, one from the coloured geometry, one from the inaccurately registered start video clip projecting onto a proxy plane instead of the geometry, and again another from the inaccurately registered end video clip projecting onto a proxy plane instead of the geometry.

Finally, the dynamic case tends to reveal more empty areas than the static case (Section A.7.4). This is because the projection onto geometry and the proxy planes is subject to the continuing motion in the start and end videos. Here, a pan in either clip will reveal an empty area in the virtual camera view. This effect does not happen in the static case. All these artefacts may be seen in Figure A.14.

## A.7 Transition Issues in Common

### A.7.1 Geometry Recovery and Processing

Recovered geometry is shown in the Ambient Point Cloud and full 3D transition types. This geometry is recovered for each portal using mostly off-the-shelf techniques; however, there is interest in the particular frames of video that we use for this reconstruction, in the pre-processing of these images before reconstruction, in the parameters chosen for the reconstruction, and in the geometry post-processing to create better renderings.

In the portal identification stage, we find the support set of video frames for each portal (Section 5.5.1). The support set contains similar visual information to the portal, but also includes frames which do not conform to the specific portal selection criteria. This can be thought of as a clustering of the candidate portal frames. We aim for the support set to contain as many different views of the scene as possible from which to recover geometry for each portal.

We first tried to recover geometry for a particular portal from frames which temporally surround the portal — each of these portal frame matches had been verified holistically, geometrically (Section 5.3) and contextually (Section 5.4). We chose frames before and after each portal frame for four seconds in time from each video associated with a portal. However, this is a very simple approach and it did not produce usable geometric models. Even though some baseline existed between the portal views, many times there was insufficient baseline for good geometric reconstruction, and what geometry was generated was largely planar and not a good proxy. Although the videos undergo some parallax in these four seconds, it is insufficient to estimate the geometry of the scene, particularly when looking at buildings in the middle distance as from city squares, parks or across rivers.

Another problem was dynamic objects such as people and vehicles. In the case where a portal has very few views for geometry reconstruction, any outliers (such as images with dynamic objects in front of buildings) were relatively influential. Ideally, these outliers would not exist in the reconstruction; however, with small numbers of views this is possible. Equally, dynamic objects could cause large regions of important structure to fail to be reconstructed at all as no correspondence is found to other views in these areas.

Finally, some videos contain significant camera shake, motion blur and over- or under-exposed saturated or starved regions. These capture artefacts cause problems for geometry reconstruction, and should be treated as outliers. In portals with few videos for reconstruction, or in portals with many videos which suffer these effects, these video frames cause problems in the geometry reconstruction. This final problem is a large stumbling block for geometric reconstruction methods that rely on unstructured video as input. An approach to mitigate this risk by exploiting the connections present in the Videoscape seemed more likely to generate good results.

As such, we generate and employ the support set of frames to solve these problems. This more varied set of frames often a) contains more frames, b) contains frames with wider baselines for common content, and c) contains more dynamic object variation. Using the support set as input to geometry reconstruction produces higher quality geometric reconstructions: with less noise, producing a qualitatively



better representation of the scene, more robust to individual dynamic objects corrupting the reconstruction, and more robust to capture artefacts such as shake and blur. This support set reconstruction allows us to recover geometry for portals where the constituent videos would not return usable geometry.

### Common Reconstruction Failures

Sky is never reconstructed, and this is the expected result: the ground truth geometry for these regions is undefined. Occasionally, objects in the sky (such as the edges of clouds which contrast with the sky) are reconstructed with inaccurate geometry and appear as sparse, low-density clusters in the reconstructed point cloud. In our experience, these appear at arbitrary depths relative to the camera and so can obstruct correctly recovered geometry from different view angles during rendered transitions. Thus, these clusters should not appear in the final geometry used for rendering, and if they are included they often cause visible artefacts. These points are removed in Section [A.7.1](#).

Ground geometry is infrequently recovered. Ground surfaces often have sparse features and are largely viewed at grazing angles where even typically diffuse surfaces exhibit specularity due to Fresnel reflection. This appearance variance to view angle makes it impossible to accurately recover the geometry. Likewise, objects which cast specular highlights and reflections and objects which are translucent or transparent fail to be reconstructed.

Buildings with repeated structure sometimes cause problems in the reconstruction, because correspondence has been incorrectly found between different parts or sides of the same building in different views. Context refinement (Section [5.4](#)) can help reduce these errors if other buildings are in view, but cannot help to disambiguate some cases. We found this problem on two buildings in particular, due to the way in which people take videos of these buildings. Big Ben and the London Eye are tall structures which are often filmed from below looking up. This commonly leaves only small amounts of the tops of other buildings in the frame, and it makes it particularly hard to correctly match the side of the building when forming correspondences. The recovered incorrect pose then creates ambiguity for the geometry reconstruction. Recent works attempt to solve this problem [[ZKP10](#), [HS12](#)], but this difficult problem is yet solved and we leave it for future work.

Moving objects are a problem in that they obscure content we wish to reconstruct, but they are not usually a problem in that they appear in the reconstruction. The support set strategy successfully mitigates the problem of dynamic occluders appearing in the geometry. These objects would otherwise require explicit modelling if the graph structure were not exploited and geometry was recovered just from camera ego-motion. In an unstructured video collection, as the videos are often captured at different times and dates, it is very uncommon for the same dynamic object (such as a pedestrian or car) to be viewed in different videos. Where this does become a problem is with multiple time-synchronized views of a dynamic object. In this case, there are sufficient examples of correspondence for the dynamic objects to be reconstructed. Some recent approaches explicitly handle these cases [[BBPP10](#), [TBP11](#)], but they make the assumption that the foreground dynamic object is always present in all videos. The integration of these different ideas and techniques is also left for future work.

## Image Distortion Correction

With an large unstructured video collection, we assume that it is not possible to calibrate and discover the intrinsic parameters of each camera beforehand. As such, radial distortion parameters (across zoom levels) are unknown. Snavely et al. [SSS06] attempts to estimate two radial distortion parameters for each image; however, in practice, the results are often significantly incorrect due to scene ambiguities. In trials, where we undistorted each input image based on these parameters, the resulting reconstructed geometry was qualitatively worse for our databases than if we simply did not estimate these parameters and undistort in the first place. Removing these parameters from the pose optimization creates a broadly more robust pose estimation and geometry reconstruction.

As such, we do not estimate radial distortion parameters nor undistort the images with these parameters before the multi-view stereo stage in our geometric recovery pipeline. These steps may be unnecessary for us because we use camcorders which produced no obvious visible distortion and we did not use wide-angle lenses. It may also be unnecessary because we do not require a metric reconstruction for our transitions (see next section).

## Fixed Focal Lengths

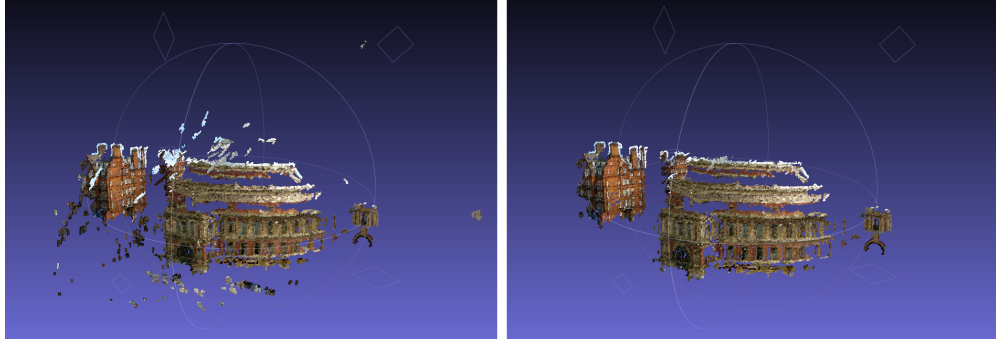
For some of our databases, we allowed the camera operators to zoom their cameras into interesting scene parts. This provides a wide variation of focal lengths in our databases, and none of these cameras are calibrated. We found that estimating focal lengths caused great variability in the quality of the reconstructions, and often recovered camera poses were wildly incorrect.

Instead, we decided to fix the focal length of all our cameras even during portions of the captured footage that were undergoing zooms. While this is patently incorrect, it creates qualitatively much better reconstructions. Frames from videos that were at zoomed focal lengths have recovered poses that are closer to the recovered geometry than was really the case. For our purpose, we accept this inaccuracy as we only use the poses to generate a virtual camera path for rendering a transition. The practical difference is that the distance covered during the transition is different from the true distance, and the angle of view change is also different from the true angle of view change. In cases where the focal length does not vary significantly, a forward/backward translation and a zoom are difficult to tell apart.

## Point Cloud Post-processing

The point cloud recovered from multi-view stereo often contains errors such as small clusters of isolated points which do not relate to scene objects, points incorrectly recovered from the sky or ground, and missing regions where windows once were. We can automatically fix these points using existing algorithms to remove points or fill holes [MKC07], but in our implemented system this part involves manually executing functions within MeshLab [CRC<sup>+</sup>11]. This is for two reasons: 1) MeshLab’s batch processing ‘server’ was unstable at the time of implementation, and 2) parameter estimation for density requires camera pose knowledge which MeshLab could not load at the time of implementation.

Point cloud clean up begins by estimating the local point spacing around each point from an estimate of the local density calculated with the closest  $n$  neighbours to each point (*Filters*→*Point Set*→*Estimate radius from density*). We use the default  $n$  of 16. Then, we perform a vertex selection conditioned upon



**Figure A.15:** Left: *Input multi-view stereo point cloud containing 2027195 vertices.* Right: *Cleaned and resampled version with sky and low-density clusters removed containing 137521 vertices.*

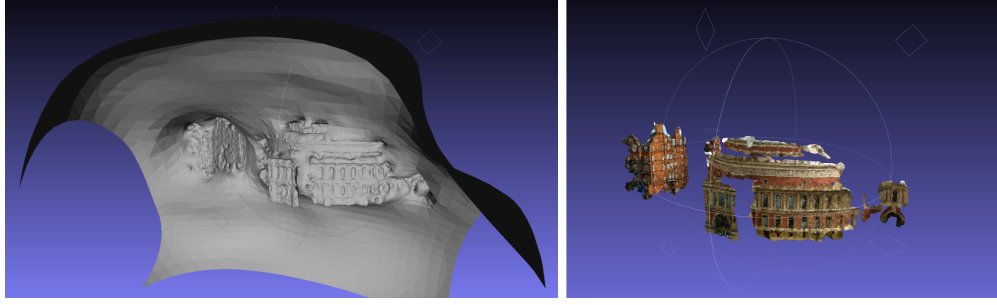
this radius (*Filters*→*Selection*→*Conditional Vertex Selection*). With the boolean expression  $rad < T$  we can select all points which have low density and then remove them.  $T$  should be defined by the expected density of the reconstruction, which at a minimum can be calculated from the size of the pixels as they project onto surfaces at the farthest scene depth. In scenes that span large depth ranges, this approach is not suitable as the expected density varies greatly across the depth range. Alternatively, Poisson surface reconstruction can be computed first [KBH06]. Typically, in these cases, isolated clusters will appear as small individual surfaces, and so can be easily removed with thresholding by volume.

Finally, the point cloud can be resampled to help speed up Poisson surface reconstruction. Point clouds with millions of vertices are often recovered, and these points were largely unnecessary for estimating the building surfaces in our databases (*Filters*→*Cleaning and Repairing*→*Merge Close Vertices*). An example cleaned point cloud is shown in Figure A.15.

### Mesh Post-processing

The cleaned point cloud is transformed into a mesh by Poisson surface reconstruction [KBH06], with octree depth parameter set to 12 and solver depth set to 8. All other parameters retain their defaults. Poisson surface reconstruction requires an oriented point cloud — vertices with normals. Fortunately, these are estimated by the multi-view stereo algorithm of Furukawa et al. [FCSS10]. As Poisson surface reconstruction attempts to extract an isosurface from the oriented point set, it tends to produce ‘hoods’ and often extends geometry into what would be the sky or ground (see Figure A.16). It also produces isolated isosurfaces for small clusters of oriented points.

To remove these unwanted geometries, we first remove small geometries (*Filters*→*Cleaning and Repairing*→*Remove isolated pieces (wrt diameter)*) with a percentage size less than 10% of the filled world space. Next, we want to both colour the mesh and remove the large-triangled hood: we first colour the mesh by transferring colour values to it from points in the cleaned point cloud which are close in world space (2%) to the mesh (*Filters*→*Sampling*→*Vertex Attribute Transfer*). Next, we remove uncoloured vertices (and their respective faces) from the mesh (*Filters*→*Selection*→*Conditional Vertex Selection*, boolean function  $r == 255$  and  $g == 255$  and  $b == 255$ ). Finally, we fill small holes. An example mesh is shown in Figure A.16.



**Figure A.16:** Left: *Poisson surface reconstruction of cleaned point cloud.* Right: *Coloured, clipped and filled version with 88881 vertices and 175354 faces.*

## Pipeline

With the above issues outlined, the final pipeline for geometry reconstruction with optional stages included is described in Algorithm 2.

## Computational Performance

For our databases, we generate support sets which extend recursively by two neighbourhoods, as this was a good compromise between speed of computation and extent of reconstruction (Section 5.5.1). This produced portal support sets with on average 70 frames for our London database. Given this, the reconstruction and tracking for 200+ portals took approximately two days, running in parallel on eight Xeon X5560 2.66GHz cores.

Even though we use state-of-the-art multi-view 3D reconstruction [FCSS10], the resulting geometry can be of poor quality, due to our database not being large enough to provide a sufficient baseline in some cases. In these cases, as motivated by our experiment in Section 6.3, we handle these problems by choosing a dissolve transition which does not require 3D geometry.

More recent work by Crandall et al. [COSH11] speeds up large-scale structure-from-motion 5-8x, increasing the possible number of image matches included in the support set of a portal, and in general the total number of possible videos in the collection. These solutions are drop-in replacements for our existing structure-from-motion estimation.

### A.7.2 Video Registration

The plane, APC and full 3D transitions all project video onto proxy or recovered geometry. To perform this projection, we need to know the camera poses for every video frame used within the projection. That is, a pose for each frame for the length of the transition from each of the start and end video clips registered against geometry in the same coordinate system.

If we employ the Snavely et al. [SSS06] method to each video frame individually, we do not generate camera poses which result in smooth motion as the video plays — frequently, the registration will produce a visible jump in the projection onto the geometry from frame to frame, and occasionally some frames will fail to be given valid poses at all. This causes large jarring artefacts in the transition such as temporally unstable reprojections, inaccurate interpolated virtual camera motions, and ‘dropped’ frames where a pose has failed to be recovered. Using this approach for video is expensive, brittle, and produces

---

**Algorithm 2:** Pipeline for geometry reconstruction including optional stages and default values for our databases.

---

**Data:** A Videoscape graph.

**Result:** Reconstructed geometry per portal.

**foreach** *portal* **do**

    Generate support set by Section 5.5.1;  
         *Optionally generate recursively; Default 2-neighbour deep;*  
     Compute camera poses by [SSS06];  
         *Optionally with fixed focal length; Default Yes;*  
         *Optionally with radial distortion parameters; Default No;*  
     **if** *radial distortion parameters estimated* **then**  
         | Undistort images;  
     Compute multi-view stereo *clusters* by [FCSS10];  
     **foreach** *cluster* **do**  
         | Compute multi-view stereo point cloud by [FP10];  
     Merge clusters by union operator;  
     Post-process point cloud by Section A.7.1;  
     Form point cloud into mesh by [KBH06];  
     Post-process mesh by Section A.7.1;  
     Add planes for full 3D transition types only by Section A.6;

---

jittery results as it does not exploit the temporal coherence within each video clip.

Instead, we combine the global registration between video clips at the portal frame with a local tracking across video frames either side of the portal frame. We first apply the Snavely et al. [SSS06] method to find good correspondences and 3D points between the portal views in each video. Next, for each video we find KLT feature points [TK91, ST94] over transition-frame-length windows in time around each portal frame, and robustly reject outliers with RANSAC [FB81] that suffer large reprojection errors after fitting a standard camera model with the eight-point algorithm [HZ04]. Bundle adjustment is performed after every 10 frames to optimize the pose and 3D locations of the 2D KLT feature points [TMHF00]. The KLT-based structure and motion recovery is computed by the Voodoo tracker [Tho06].

We now have many separate pose results in many different coordinate systems: one coordinate system for the portal frames, and one each for the 3D points recovered from KLT tracks for each video. Intuitively, we expect that there exists a transformation matrix relating these two coordinate spaces. However, it cannot be estimated by typical methods [Hor87, ELF97] as the scale transform is non-linear from the centre of projection of the camera, leading to incorrect alignments if iterative closest point methods are applied. One possible solution is to fix the focal lengths used in both the global and local approaches to remove it as a parameter from any optimization, and to attempt to align their corresponding point clouds [Zha94]. However, this also fails because, as previously stated, the point cloud from the video tracking is derived from clips with potentially little-to-no parallax, and so its points might be distributed on a plane or spherical sector. This leads to wildly inaccurate fitting, even with hand-helped initializations, and so produces unusable results.

To solve this problem, we couple the smooth, robust KLT tracks with the 2D-to-3D correspondences found at portal frames between clips. We find 2D KLT points from the local tracking which match 2D SIFT points from the global registration in the portal frames; that is, feature points which have sub-pixel Euclidean distances in the image plane. Given these, we follow the KLT tracks to neighbouring video frames and optimize new extrinsic parameters by the error in the reprojection of the 3D points which have been matched, via their 2D SIFT feature points, to 2D KLT feature points. We represent rotations using three Euler axis angles to minimize the number of optimization parameters. We solve this optimization using simulated annealing [KGV83], though other optimization methods are equally applicable [NW99].

We perform this optimization scheme bi-directionally out from the portal frames, and we chain translations and rotations from frame to frame. Should there be insufficient feature points to estimate our extrinsic parameters, then we increase the 2D reprojection error for matching SIFT and KLT points until there are a sufficient number of correspondences. While this increase of correspondence error does produce minor ghosting in the final result, importantly it still produces smooth motions. This is more pleasing than the brittle alternative approach described above.

In our experiments, KLT feature tracking worked well for videos that do not suffer heavy shake. Aligning the KLT features to feature points used in the 3D reconstruction yields cameras with sub-pixel reprojection errors. However, in the case of shaky video segments which might have rolling shutter artefacts, the quality deteriorates considerably and videos are no longer accurately aligned with the 3D

geometry, leading to ghosting artefacts in the 3D transitions (particularly in Scene 4).

For our databases, standard KLT tracking was sufficient for tracking around portals, but other databases may require exposure-compensated KLT tracking. This is a simple component swap and does not change any of the computation steps.

**Alternative Method** Ballan et al. [BBPP10] have a similar but subtly different problem of registering videos to geometry. They begin with a similar standard pose estimation [HZ04], but find that the accuracy for video is not sufficient. Personal correspondence with the authors confirmed that their initial approach caused ghosting. They propose a pose refinement strategy which exploits the geometry capture stage of their system, where photos of the scene are taken in a structured way specifically for multi-view geometry reconstruction. Only later on is the scene captured by their video cameras. This differs considerably from our case, where we try to reconstruct the geometry from the videos themselves, but we do have more videos to exploit. Their pose refinement proceeds [BBPP10, Section 3.1, paragraph 2]:

Treating the calibration so far as an initialization, we perform a second optimization of camera poses. We use particle filtering to minimize the sum-of-square-difference (SSD) between each [video frame] and our render-engine’s versions of the reconstructed and textured scene in different poses. In this case, the texture is obtained as the median reprojected texture from a temporal window of 1000 frames of the same camera A (subsampling for efficiency).

We implemented this technique (though with the different simulated annealing optimization strategy) and found it to be unsuitable for our case. Problems arise when the rendered view, to which we are comparing to the video frame via SSD, is incomplete and has empty regions. This is unavoidable in scenes where sky is present even if the rest of the scene has accurate geometry. These areas of no geometry must be drawn with some arbitrary colour (such as black or mid-grey) in the rendered version. The difference in the SSD computation between the original video frame and these empty regions becomes large and dwarfs the important difference between areas with textured geometry.

Using proxy planes for all unknown regions, as in the full 3D transition types, is also a problem in the general case. While sky regions are rendered reasonably well as they are effectively at infinity, the ground plane proxy is often wrong for complex environments. Also, it is difficult to ignore these regions in the SSD computation as, with unknown geometry accuracy or coverage, it is hard to say where in the video frame these regions lie. As such, we found this second post-process optimization method inapplicable for cases where the video frame has large regions of unrecovered or inaccurate geometry, and this is the case in some of our videos. All examples in the Ballan et al. [BBPP10] paper and supplementary video show complete or virtually complete geometry coverage, as they include a pre-processing step where scene geometry is recovered from photographs specifically taken for this purpose.

### A.7.3 Camera Interpolation

The plane, APC, and full 3D transitions generate frames by rendering a view from a virtual camera. This virtual camera interpolates in 3D space from a camera pose of a frame in the start clip to a camera pose of a frame in the end clip. The start and end frame poses are computed in Section A.7.2. Our goal is to



recover a seamless virtual camera motion path which blends the existing camera motions in the start and end video clips. For instance, if the start clip contained camera shake but the end clip did not, then the virtual view should include shake that fades out over the course of the transition. Likewise, any velocity and acceleration to which the camera is subjected should be smoothly interpolated across the transition: if the camera pans in either clip then the move from video frame pan to virtual camera pan should be seamless.

Linearly interpolating the recovered transition start- and end-point camera poses does not produce a convincing camera motion interpolation. Applying a Bezier-curve-based slow in and slow out, as is typically used in photo exploration interface [SSS06, GAF<sup>+</sup>10], does not correctly interpolate motion in the start and end video clips. This is especially true if the video clips contain camera shake, which is often the case for hand-held captured video. We not only need to transfer the broad motions, but also blend between higher frequency motions. We want the virtual camera to blend between the two styles of camera motion, but existing work on style transfer [KRE<sup>+</sup>] only transfers motion from one video to a virtual view, and does not blend between two styles.

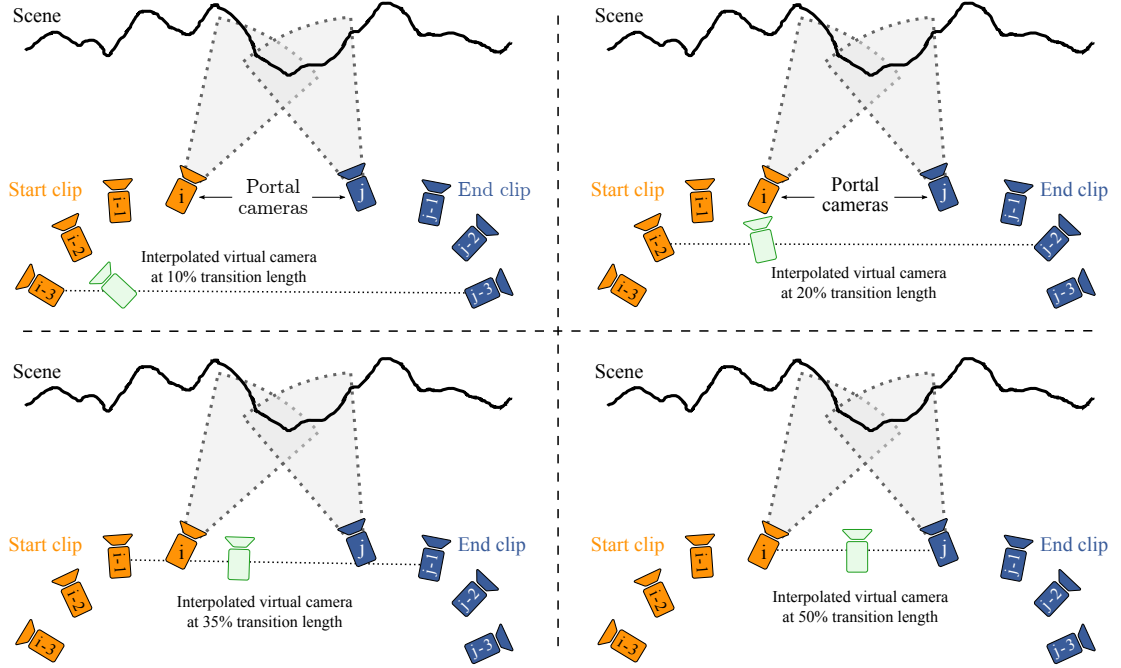
Instead, as we recover the pose for nearby frames in time to either side of each portal frame (Section A.7.2), we can directly interpolate the per-frame poses to generate the new pose for the virtual view (Figure A.17). We interpolate the position and three perpendicular vectors of rotation separately, then reform the transformation matrix to generate the new pose for the virtual view. The interpolation constant can be generated from a Bezier curve to apply slow in and slow out artistic effects to the virtual view, but we found the most convincing camera motion was with a linear interpolation constant. We believe this is because the camera poses being interpolated are already undergoing realistic velocity and acceleration for both position and rotation across the transition, and so no artificial slow in or slow out needs to be added. This approach also successfully interpolates camera shake.

Finally, the full 3D static transition still performs this per-frame pose interpolation even though the video clips do not play. This produces a much more realistic virtual camera motion than interpolating only between the two portal frame poses as there is no jolting change of acceleration when switching to the virtual view. In this static case, the progression of transition frames is slightly different. The interpolation begins by interpolating between the poses at frame  $i$  in the start clip and frame  $j - t$  in the end clip. The transition ends by interpolating between the poses at frame  $i + t$  in the start clip and frame  $j$  in the end clip. This ensures the transition starts and ends at the portal frames.

#### A.7.4 Empty Areas and Inpainting

All transition methods can create empty areas in the rendered virtual view transition sequence where no geometry or proxy exists (see Figure A.11). These areas contain no content and are filled with black. In image-to-image transitions, as in photo tourism applications, the off-putting appearance of these empty spaces is often mitigated by introducing a persistent black border around all images, or by viewing the images in a 3D space with a virtual camera that has a wider field of view than the original camera. In these cases, when transitioning, the empty spaces merge into the black border and so are less off-putting.

Ideally, these empty spaces would not be seen and would be filled, perhaps with a view of the



**Figure A.17:** Clockwise from top left: *Progression of interpolated virtual camera (green) position and rotations. Frames  $i$  and  $j$  are the middle frames of a transition in all but the warp and full 3D static transitions. The transition progression beyond frames  $i$  and  $j$  ( $+1, +2, +3$ , and so on) are not shown.*

correct part of the world rendered from a wider geometry reconstruction, or from a hallucination of what the sky may have looked like by inpainting. Ensuring a wider geometry reconstruction is difficult: we cannot guarantee in an unstructured video collection to have ever seen the parts of the world that may be revealed in a virtual view. Furthermore, even if we had seen it in a video, we cannot guarantee to reconstruct its geometry. Image inpainting techniques have recently advanced and can now cope with small areas of structure [BSFG09, BSGF10, PKVP09, KV10], but these often fail to generate plausible results in difficult examples (such as street scenes) over larger hole areas. These works have also yet to be fully extended to video which, from our experience on other projects, is a non-trivial extension [GTK<sup>+</sup>12, GKT<sup>+</sup>12]. If the quality of the results of such techniques does improve, we would need to pre-compute all transitions as typically these techniques do not work in real time.

We could mitigate the effect of empty regions by pulling back the camera and creating an artificial border as in many photo tourism applications. However, this is unappealing for video that is often shot in a first-person style. One of the major benefits of this kind of video is the feeling of immersion, and this is made all the more so by viewing fullscreen video on a display device. Introducing a border might reduce immersion in the viewer. We could attempt to fill in the empty regions to provide an as-seamless-as-possible experience, but as previously stated this is difficult.

In the full 3D transitions, we place sky and ground planes in the scene to cover with proxy geometry all possible empty spaces in the virtual view. While this is sometimes a bad proxy, it does ensure that the maximum amount of screen space is filled with video-projected surface. The plane transition also ensures maximal coverage, but also suffers from motion in the projected video. Ambient Point Clouds

attempts to eliminate these empty spaces with ambiguous depth point clouds but, as discussed in Section A.5, this is not always successful and itself introduces temporal artefacts in the form of small flickering empty spaces. For warp transitions, we can perform a simple action in image space to inpaint some areas with moderate success. We repeat the colour of pixels along the frame edges to cover empty regions. This is simple with OpenGL using the `GL_REPEAT` texture parameter when compositing the different parts of the warp. This trick works well for sky regions as the content is composed of colour gradients and clouds with free-form structure, and repeating and blending these colours between two different frames often produces acceptable results. However, this works less well for structured areas. This repeating trick approach would also work on the sky and ground planes of the full 3D transitions. Finally, inpainting is not necessary in dissolve or cut transitions.

### A.7.5 Transition Timing Differences

Given a pair of matched portal frames, we must decide where they appear within a transition. For instance, a dissolve transition could start with one portal frame and end with the other, meaning that the frames during the transition contain footage from after the portal frame in the start clip and before the portal frame in the end clip. However, as the camera shots may be performing arbitrary movements (such as pans), there could be very little visual link at all during parts of the dissolve transition. Instead, a dissolve transition should be set such that the middle frame is formed from 50% of each portal frame. This ensures that the clips visually match at some point during the transition.

This timing problem is apparent in all transitions, but its effect is more pronounced in others. As such, we describe the timings for each transition:

#### Cut

The cut has no timing difficulties: the start clip portal frame is followed immediately by the end clip portal frame.

#### Dissolve

As the exemplifier; the dissolve sets the portal frames to be in the middle of the transition.

#### Warp

The warp transition places the portal frames at the start and end of the transition. A warp transition should have the portal frames in the middle of the transition, but this has implementation implications which would require further SfM at the transition start and end frames to solve: Feature-point correspondences must be found in 2D across most areas of the two portal frames for our moving-least-squares warp to successfully transition between them. Were the portal frames to be in the middle of the transition, then we would have to reliably find feature point correspondences which would broadly cover the transition start and end frames. This is difficult because of the potentially arbitrary camera movement before and after portal frames.

#### Plane

In the plane transition, both videos will always project to somewhere on the plane as it has infinite dimension. However, if, over the course of the transition, the videos no longer visually intersect,

then regions of black will appear in the virtual view where no video is projected. This should be avoided; thus, we set the portal frames to the middle of the transition.

### Ambient Point Clouds

Ambient point cloud transitions place the portal frame in the middle of the transition. The location of the portal frame in the transition makes little difference to the ambient point clouds themselves as these are computed from the start and end transition frames regardless, but the 3D reconstructed geometry projected with video suffers as per the full 3D dynamic transition below.

### Full 3D

The full 3D static transition is simple as no video plays: the start clip portal frame begins the transition into virtual camera, and the end clip portal frame ends the transition back into video. This ensures visual similarity through the transition. In the full 3D dynamic case, if the camera motions in the start and end clips pull away from the virtual view, there is still correctly coloured geometry underneath if no projection is present. However, this is undesirable as the fidelity of the reproduction is significantly worse as the colour is per-vertex at the resolution of the mesh (see Figure. A.16) and there are no dynamic scene objects present. Hence, we set the portal frames to be in the middle of the transition.

## A.8 Video Stabilization

Often, hand-held video includes distracting camera shake which we may wish to remove. However, if we stabilize the videos with software before we perform our pre-processing, we jeopardize our vision-based matching (Chapter 5) and reconstruction (Chapter 6) as software stabilization alters the camera's geometric properties, such as the centre of projection, by translating and scaling within the video frame to remove shake. Hardware stabilization, as either lens- or sensor-shift-based optical image stabilization, also changes the centre of projection and creates off-axis projections which are not supported in standard vision-based camera models.

Ideally, we would stabilize between portals in real time during interaction, but current software methods are too slow. One might think to smoothly 'turn off' stabilization as portals approach in time, but this leaves critical parts of the video unstabilized. Instead, we pre-compute 2D affine stabilization parameters as a per-frame crop region computed with a custom Deshaker build [Tha12]), but we do not apply them immediately or permanently to our input videos. Thus, we pass our input videos unaltered into our reconstruction pipeline. Then, when we view the videos in our explorer application, we apply the pre-computed stabilization parameters in real time in our renderer. During transitions, we interpolate the stabilization parameters across the transition. For geometry-based transitions with a virtual camera, we project the original unstabilized video footage and only stabilize the virtual camera view. This allows full stabilization at every video frame while not affecting the geometry reconstruction or reprojection.

One positive side-effect of this method is that stabilization can be turned on and off at any time by the viewer. It may be more appropriate for certain databases or particular videos within a database to not be stabilized. For instance, camera shake can be an important indicator of surface terrain when

in vehicles, or perhaps the camera operator intended for a video to have fast jerky movement for a particular expressive effect. Also, software stabilization is not flawless and some scenes currently cannot be stabilized without artefacts or without suffering rolling shutter wobble [LGJA09, LGW<sup>+</sup>11, GKE11]. Our approach allows the viewer control should these tricky stabilization cases arise.

## A.9 Transition Feature/Artefact Table

We collate and categorize all feature and artefact types in each transition in Table A.1. We will use this table to cross-reference comments from participants in the user study in Section 6.3.4. This is a repetition of 6.1 for convenience.

	Cut	Dissolve	Warp	Plane	APC	Full3DDyn	Full3DSta
<i>Feature</i>							
Registered scene			•	•	•	•	•
3D effect			○ <sup>1</sup>		•	•	•
Dynamic objects		•	•	•	•	•	
Smooth virtual camera (A.7.3)			○ <sup>2</sup>	•	•	•	•
Common familiarity	•	•					
Signifies change of time		•					
Explicit motion cues					•		
Frozen time							•
<i>Artefact</i>							
Ghosting (static objects)		•		• <sup>3</sup>			
Ghosting (dynamic objects)		•	•	•	•	•	
Orientation loss (A.1)	•	•					
Bad corresp. swirls (A.3)			•				
Frame edge flickering (A.3)			•				
Skewed scene (A.4)				•		○ <sup>4</sup>	○ <sup>4</sup>
Temporal pepper noise (A.5)					•		
Multiple scene elements (A.6)					○ <sup>5</sup>	•	○ <sup>6</sup>
Recovered geom. failures (A.7.1)					•	•	•
Empty black regions (A.7.4)				•	○ <sup>7</sup>	○ <sup>8</sup>	○ <sup>8</sup>

**Table A.1:** A table collating all features and artefacts for each transition type. Section numbers for text explaining each feature or artefact are included in parentheses. 1: Partial, only with good regular correspondence and flow correction. 2: Velocities only from feature-point tracks. 3: Although the scene is sparsely registered, ghosting is still present in almost all transitions because the plane is an inaccurate proxy to the true geometry. 4: On proxy planes only. 5: An image is formed within the APC as it appears as a noisy plane during slight view changes only. 6: Not as prominent as full 3D case as global registration at portal frames is better aligned to geometry, but still possible. 7: APC reduces, but not maximally, empty regions; introduces pepper noise. 8: Minimized as much as possible given video-video-geometry registration.

## **Appendix B**

### **Individual Set Perceptual Score Analysis**

Over the next ten pages, perceptual scales are shown for each set, and specific features/artefacts are cross-referenced with comments to investigate the properties of the scene which have affected the result. Scene descriptions and special features are from Figure [B.1](#) and Table [B.1](#).



Scene	Name	Set	S/C	Scene Description	Special Features
1	County Hall	1	S	View change along a bridge over a river into pan left, observing Edwardian Baroque County Hall on the bankside.	Camera shake in start video. Bird in flight in foreground, people in foreground to left.
		2	C	Pan left over a river from bank-side changes to pan left from bridge observing County Hall.	Travelling boats on river, people in foreground to left.
2	Palace of Westminster	3	S	Middle distance shot of Neo-Gothic Palace changes to farther distance shot in pan right.	Road traffic and pedestrians at bottom of frame.
		4	C	Palace in far shot in pan right changes to middle distance shot in pan right.	Many flying birds, people, travelling boats, distant road traffic and lamppost occluder.
3	Victoria Embankment	5	S	Pan left on bridge over river across Neo-Gothic buildings changes to pan right with bridge road in foreground.	Flying bird and pedestrians in foreground, distant road traffic and flags.
		6	C	Pan right across bankside changes to view of bridge surface in pan right.	Many flying birds, people, travelling boats, distant road traffic and lamppost occluder.
4	Royal Albert Hall	7	S	Translate right changes to translate left with full frame Neo-Romanesque building.	Significant camera shake, rolling shutter artefacts, road traffic in middle distance.
		8	C	Pan left changes to translation left with full frame building.	Significant shake in end video, rolling shutter artefacts, road traffic at frame bottom.
5	Millennium Bridge	9	S	Translate forward along modern glass/steel bridge changes to translate plus pan left.	Camera shake and many people in foreground.
		10	C	Pan left and zoom from bank-side changes to pan left upon suspension bridge.	Camera shake, travelling boat in middle distance and people in foreground.

**Table B.1:** All scenes breakdown with set number, common names, slight or considerable view change (S or C), contents and special features identified.



(a) Scene 1: County Hall, sets 1 & 2.



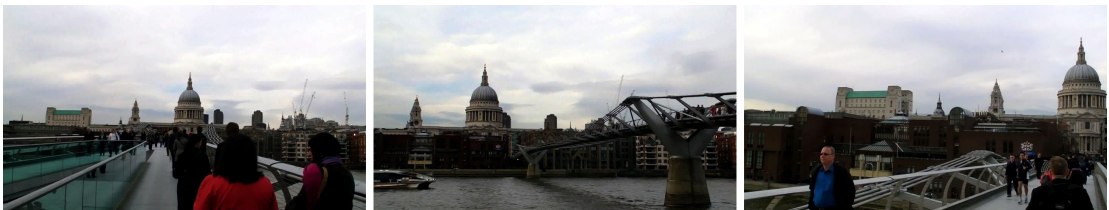
(b) Scene 2: Palace of Westminster, sets 3 & 4.



(c) Scene 3: Victoria Embankment, sets 5 & 6.

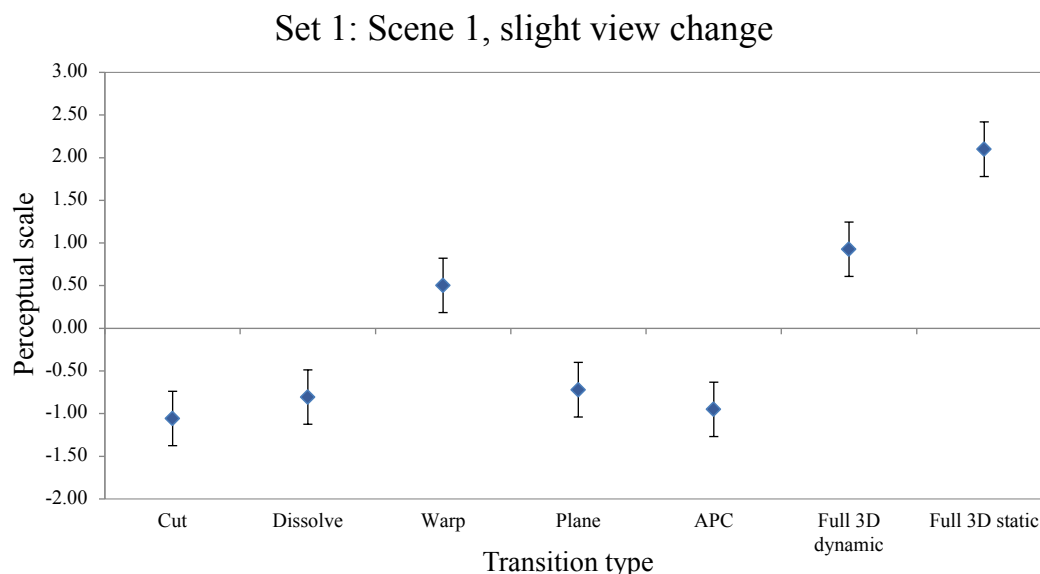


(d) Scene 4: Royal Albert Hall, sets 7 & 8.



(e) Scene 5: Millennium Bridge, sets 9 & 10.

**Figure B.1:** All scene slight and considerable view changes. Left: Start video frame for slight view change transition. Middle: Start video frame for considerable view change transition. Right: End video frame for both slight and considerable view change transitions. That is, the slight view change transition moves from the left column to the right column, and the considerable view change transition moves from the middle column to the right column in each case.



**Figure B.2:** Mean and 95% confidence error bars plotted on a perceptual scale for the different transition types for set 1. Perceptual scales are in z-score order, with the group mean at 0 and y-value representing multiples of the group standard deviation.

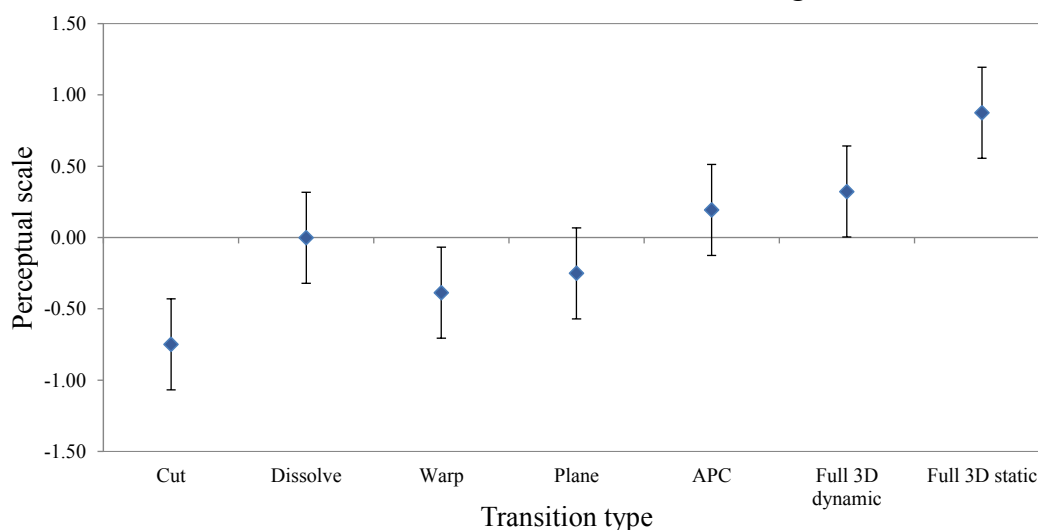
### Set 1: Scene 1, slight view change

*Description:* View change along a bridge over a river into pan left, observing Edwardian Baroque County Hall on the bankside.

*Features:* Camera shake in start video. Bird in flight in foreground, people in foreground to left.

Set 1 is a relatively simple transition with most of the frame taken up by a building. Dynamic objects and difficult geometry (London Eye) sit at the edges of the frame, and cause slight geometry errors. There is slight camera shake in the start clip and a slow pan in the end clip, but the video registration is good as no static ghosting is visible. As such, it is not surprising that the full 3D transitions perform well. The static and dynamic transitions are quite hard to tell apart: the dynamic transition has a ghosted flying bird near the end, but more importantly the slight shake and pan creates empty regions which are filled in the static transition. The warp is similarly convincing with no empty regions; however, the camera motion is not as smooth as in the full 3D transitions. Below the perceptual scale mean, the remaining four transitions are each different in appearance: the plane suffers slight skewing, APC suffers pepper noise and double images (one on the geometry, one in the APC due to slight view change), the dissolve suffers static ghosting due to the end clip pan, and the cut sees a sudden jump in the position of the building within the frame. Figure B.2 shows the perceptual scores.

### Set 2: Scene 1, considerable view change



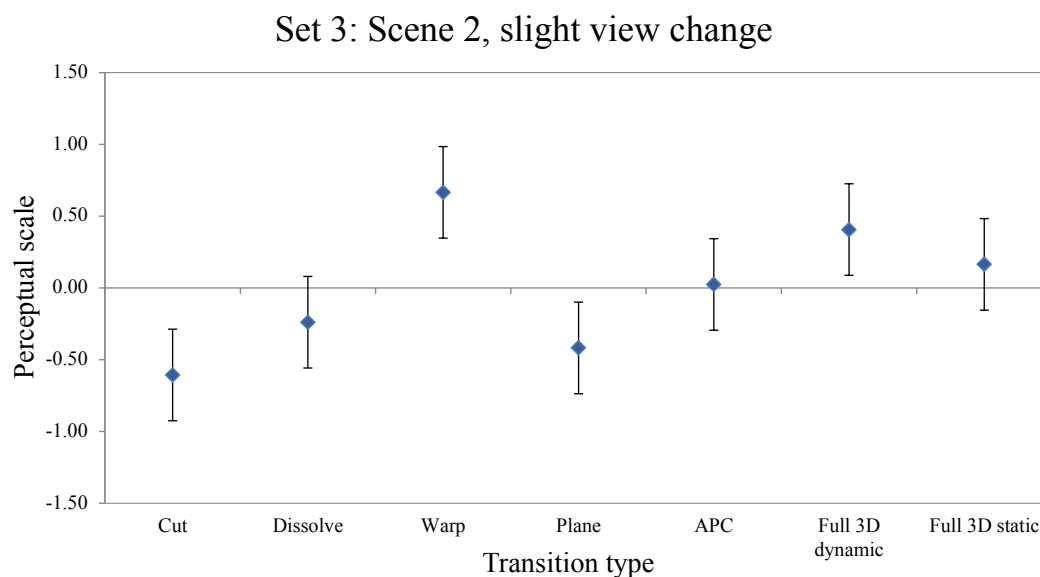
**Figure B.3:** Mean and 95% confidence error bars plotted on a perceptual scale for the different transition types for set 2. Perceptual scales are in z-score order, with the group mean at 0 and y-value representing multiples of the group standard deviation.

#### Set 2: Scene 1, considerable view change

*Description:* Pan left over a river from bankside changes to pan left from bridge observing County Hall.

*Features:* Travelling boats on river, people in foreground to left.

As a considerable view change, in comparison to set 1 we expect the warp transition to fare worse and the APC transition to fare better — this is exactly what we see. This start clip in this scene pans across a river, showing a moving boat. The two pans in the start and end clips move to the left, meaning that the dissolve transition has less static ghosting (and so its perceptual score is farther from the cut than in set 1). As expected, the plane transition suffers more shear than in the slight view change case. The warp transition loses all sense of camera motion for this considerable view change, and also suffers more flickering artefacts in the middle of the frame as the scene content is too different for the flow-based ghosting correction to overcome. APC is ranked much higher in set 2 than in set 1 even though the pans create the rare case where an APC separation is visible (Figure A.11) — comments from participants revealed that the added sense of motion helped improve the ranking in this case. Finally, the full 3D transitions are ranked first and second. As in set 1, there is some incorrect geometry at the frame edges, and the full 3D static and dynamic transitions are difficult to tell apart. Once again, the dynamic transition maintains the moving boat through the transition but also adds more empty areas. We suggest these empty area differences create a large perceptual preference difference. Figure B.3 shows the perceptual scores.



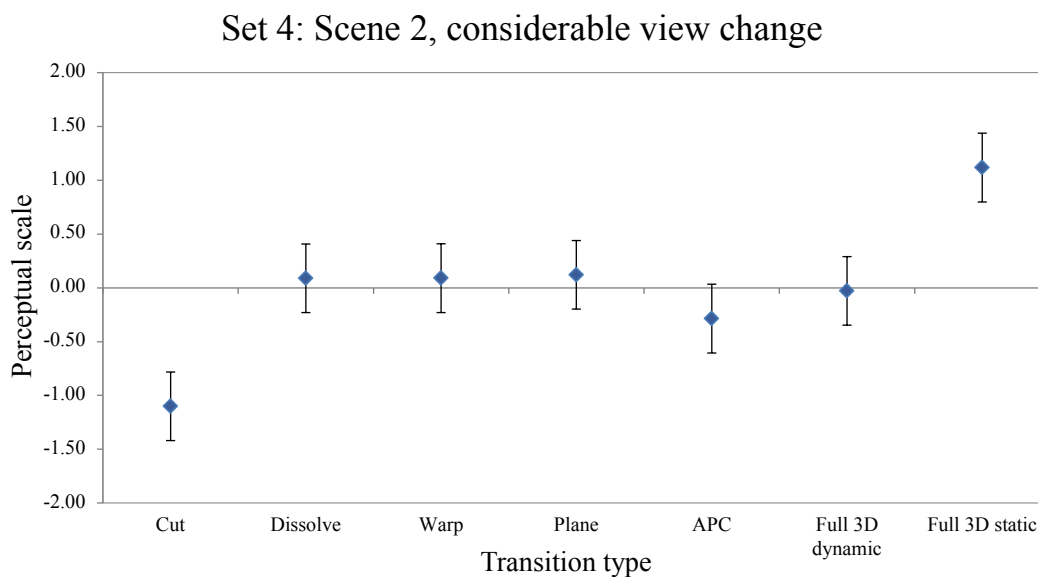
**Figure B.4:** Mean and 95% confidence error bars plotted on a perceptual scale for the different transition types for set 3. Perceptual scales are in z-score order, with the group mean at 0 and y-value representing multiples of the group standard deviation.

### Set 3: Scene 2, slight view change

*Description:* Middle distance shot of Neo-Gothic Palace changes to farther distance shot in pan right.

*Features:* Road traffic and pedestrians at bottom of frame.

Set 3 is almost entirely covered by building in both frames, and at the portal frames this building is in almost the same position in the frame (the end clip camera is zoomed in). As such, the cut in this set is a jumpcut, though perhaps an unusual one as the end clip pans to the right. This pan creates double images and static ghosting in the dissolve transition. As there is little change in view, the plane transition should perform well. However, there is considerable vertical axis skewing caused by a large scene depth range affecting the plane fitting. This small change in view is beneficial for the warp transition, which is top ranked. Minor flickering occurs on the thin Gothic features at the top of the building, but the transition is otherwise artefact free. The three transitions with recovered geometry all perform similarly: The zoom causes APC to appear quite noisy, but the smooth camera motion and accurate recovered geometry give a pleasing fluidity. The two full 3D transitions are hard to tell apart, but due to the dynamic video projection the full 3D dynamic case creates a smoother camera motion which we suggest accounts for its slightly higher score. Figure B.4 shows the perceptual scores.



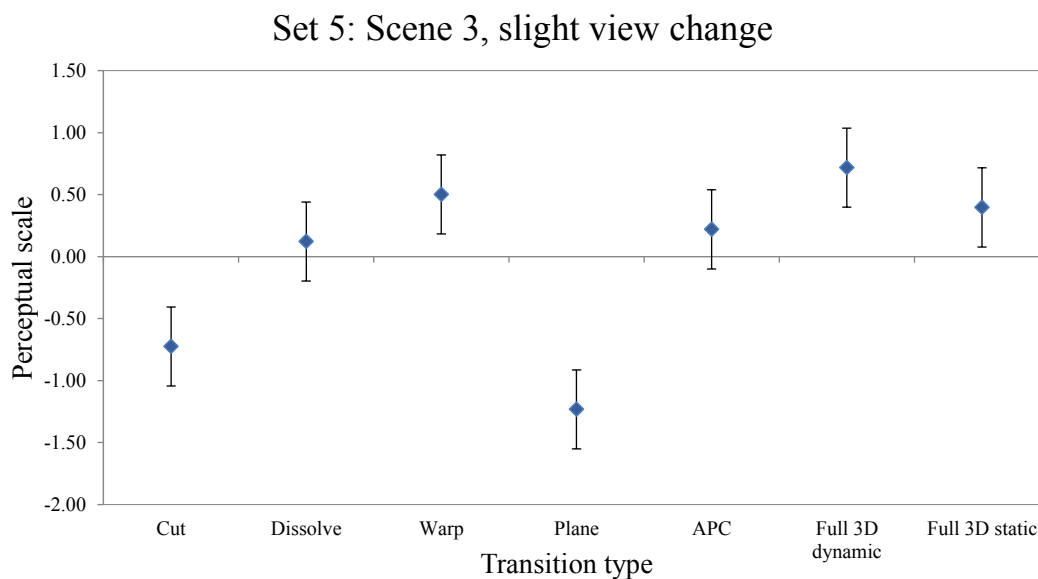
**Figure B.5:** Mean and 95% confidence error bars plotted on a perceptual scale for the different transition types for set 4. Perceptual scales are in z-score order, with the group mean at 0 and y-value representing multiples of the group standard deviation.

#### Set 4: Scene 2, considerable view change

*Description:* Palace in far shot in pan right changes to middle distance shot in pan right.

*Features:* Many flying birds, people, travelling boats, distant road traffic and lamppost occluder.

The set 4 perceptual scores, when compared to set 3, once again show that warps are not appropriate for considerable view changes and full 3D static transitions are appropriate. The considerable view change involves a large scene scale difference with foreground dynamic objects and occluders, as well as a matching camera pan in start and end clips. The dissolve transition fares comparatively well here as the pans are matched in direction and speed, and the scale change masks many of the usual static ghosting issues. The warp transition has very little static ghosting on the landmark building, but the flow correction creates large undulations in the foreground as the occluders warp and fade away. The plane transition performs relatively well in this case with no skewing, but is let down by static ghosting errors from inaccurate video registration. The full 3D dynamic case is equally let down by bad video registration, but otherwise looks largely identical to the plane transition — here, the proxy geometry appears as good as the recovered geometry as the major motion is a zooming and image-plane translation motion, not a rotating motion. We postulate that the video registration is confused by the dynamic objects in the foreground. With no such static ghosting errors caused by inaccurate video registration, the full 3D static transition is perceptually preferred by some margin. Figure B.5 shows the perceptual scores.



**Figure B.6:** Mean and 95% confidence error bars plotted on a perceptual scale for the different transition types for set 5. Perceptual scales are in z-score order, with the group mean at 0 and y-value representing multiples of the group standard deviation.

#### Set 5: Scene 3, slight view change

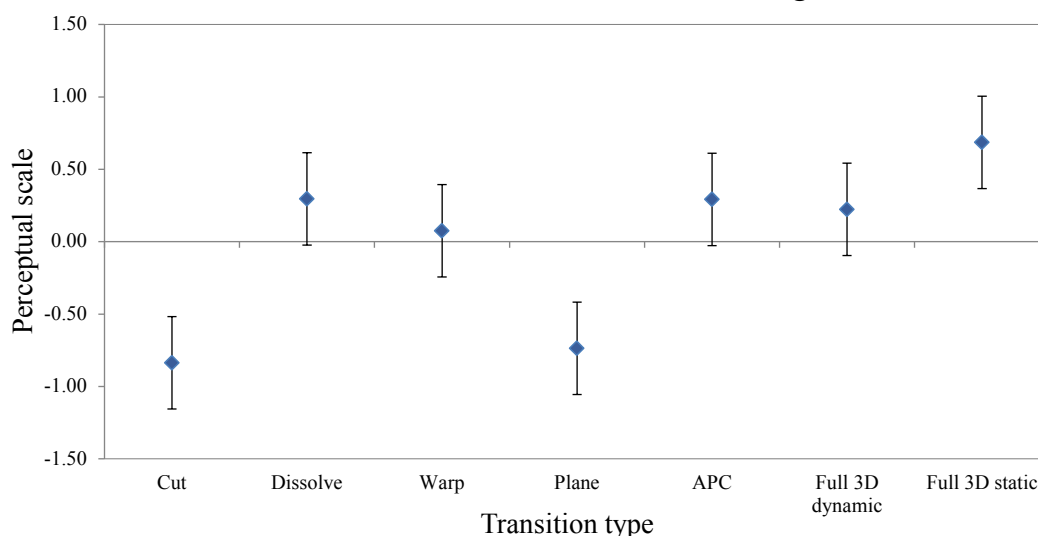
*Description:* Pan left on bridge over river across Neo-Gothic buildings changes to pan right with bridge road in foreground.

*Features:* Flying bird and pedestrians in foreground, distant road traffic and flags.

Set 5 is noted for having contrasting pans. Here, the most noticeable exception to our expectations is the plane transition which has a perceptual score lower than the cut. The plane transition suffers large skews which also creates large empty areas — approximately 36% of the frame is empty in the worst case. This is made worse because an empty area persists through to the end of the transition and causes a very large pop in as we move from virtual to video. The rest of the transitions (ignoring cut) are largely equivalent, though each have different minor artefacts: the warp transition has flickering from thin image features and inpainting, the APC transition has a slight double image from within the APC, and the full 3D transitions have noticeable pixel sliding from inaccurate geometry in the centre of the frame. We suggest that it would be difficult to rank these artefacts against one another for this set. Full 3D dynamic does have the largest perceptual score, and notably larger than full 3D static. In this case of contrasting pans, the full 3D static case has larger empty spaces than the dynamic case as the video projection does not fill the space introduced by the interpolated camera path. With a simpler virtual camera interpolation method, this empty region would not appear; however, then the camera motion would not match the start and end clips and so would jerk once when moving from video to virtual and then again when moving from virtual to video. Figure B.6 shows the perceptual scores.



### Set 6: Scene 3, considerable view change



**Figure B.7:** Mean and 95% confidence error bars plotted on a perceptual scale for the different transition types for set 6. Perceptual scales are in z-score order, with the group mean at 0 and y-value representing multiples of the group standard deviation.

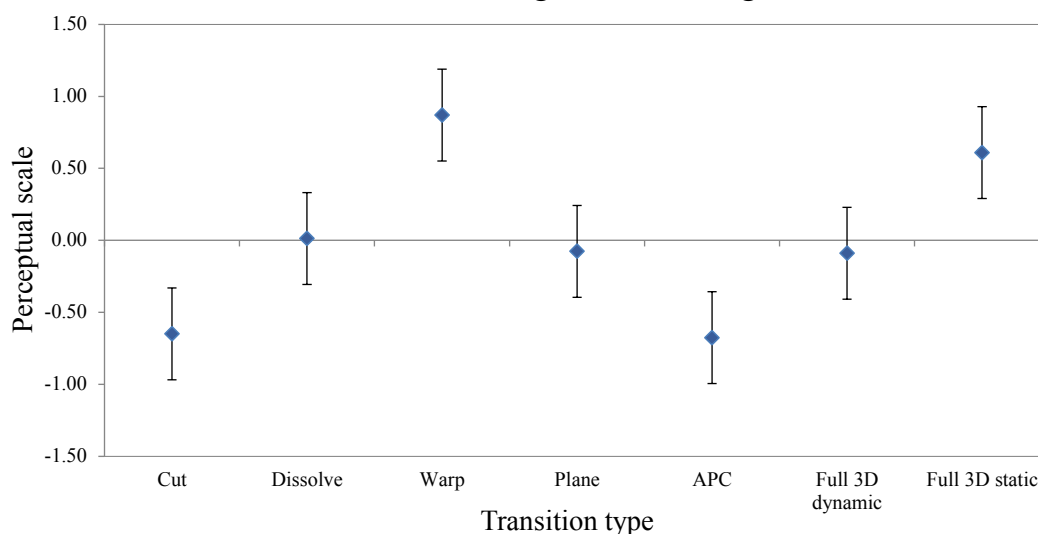
#### Set 6: Scene 3, considerable view change

*Description:* Pan right across bankside changes to view of bridge surface in pan right.

*Features:* Many flying birds, people, travelling boats, distant road traffic, and lamppost occluder.

Set 6 returns to matching pans. Again the plane transition suffers skew and empty areas — in both cases this scene is not well modelled by a plane as it contains perpendicular structures. As in set 5, the remaining transitions (ignoring cut) are largely equivalent, with full 3D static transitions scoring slightly higher. The corresponding pans help the dissolve, and the warp suffers artefacts from being unable to cope with new content revealed by the large angular change in view. In our opinion, the APC transition for set 6 is the most successful of all sets. The large angular view change allows the APC to spread out and this gives good motion cues, resulting in a comparatively large perceptual score for APC. Finally, the full 3D transitions: The matching pans now provide an advantage to the static transition, which suffers less empty areas. Otherwise, the dynamic transition maintains interesting moving objects through this transition (a boat, pedestrians, and most noticeably a bird in flight), but this seemed not to offset the added empty regions on the perceptual scale. Figure B.7 shows the perceptual scores.

### Set 7: Scene 4, slight view change



**Figure B.8:** Mean and 95% confidence error bars plotted on a perceptual scale for the different transition types for set 7. Perceptual scales are in z-score order, with the group mean at 0 and y-value representing multiples of the group standard deviation.

#### Set 7: Scene 4, slight view change

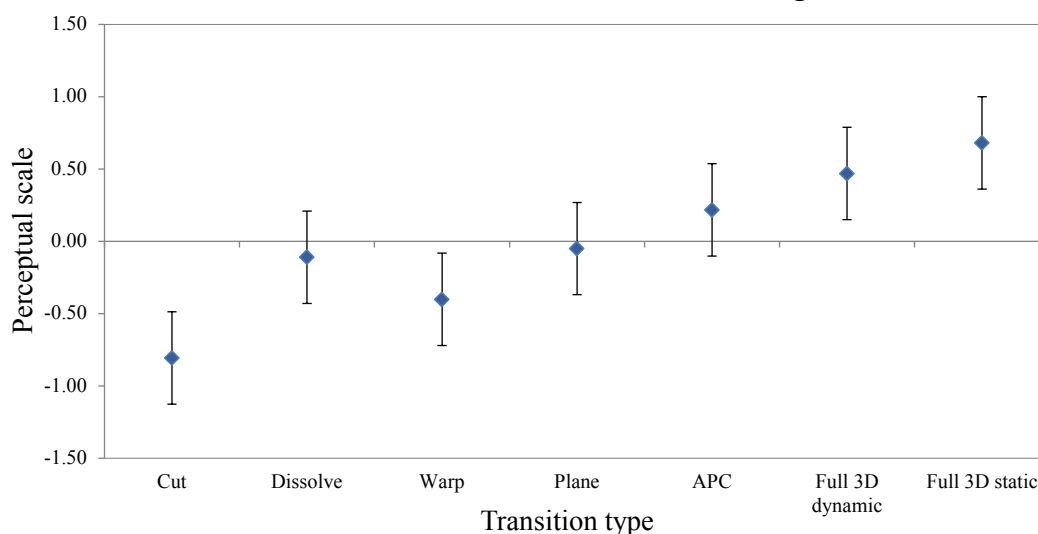
*Description:* Translate right changes to translate left with full frame Neo-Romanesque building.

*Features:* Significant camera shake, rolling shutter artefacts, road traffic in middle distance.

Set 7 has the most violent camera shake of all sets, causing roller shutter wobble artefacts in both start and end clips. In this difficult case, even though the geometry is good (as it is computed from the support set and not from the individual videos), the video registration is inaccurate. Given this, we would expect transitions which do not project with the video registration to perform better, as ghosting would be less and perceived shake would be reduced. Participant results are in line with that expectation, with warp and full 3D static transitions perceptually ahead. All transitions apart from cut and dissolve have artefacts.

APC surprises here as it has a similar score to the cut transition when we would expect it to be higher. The combination of static ghosting and noisy APC through a slight view change creates a very confusing visual impression, with very little of the structure in the scene providing a visual anchor. The plane and full 3D dynamic transitions appear very similar as the plane is a good proxy in this case, and so they have similar perceptual scores. The warp transition has undulating artefacts, but its stability removes the camera shake. Finally, the full 3D static transition has convincing geometry with no static ghosting, though there is one noticeable piece of missing geometry. However, our camera interpolation smoothly transitions between the motions in the start and end clips, and this includes the camera shake. Our virtual camera motion here is successful in interpolating the shakes, and as such the virtual motion still has shake. We suggest that this is why the warp has a higher perceptual scores. As participant 8 commented: “anything to reduce the camera shake”. Figure B.8 shows the perceptual scores.

### Set 8: Scene 4, considerable view change



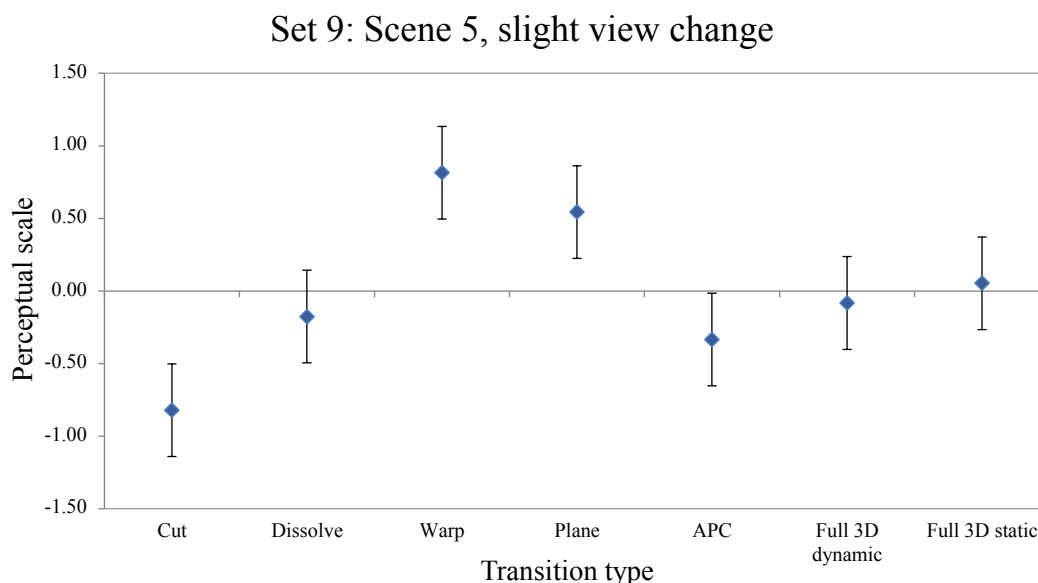
**Figure B.9:** Mean and 95% confidence error bars plotted on a perceptual scale for the different transition types for set 8. Perceptual scales are in z-score order, with the group mean at 0 and y-value representing multiples of the group standard deviation.

#### Set 8: Scene 4, considerable view change

*Description:* Pan left changes to translation left with full frame building.

*Features:* Significant shake in end video, rolling shutter artefacts, road traffic at frame bottom.

Set 8 has shake only in the end video, but introduces a pan instead. We see the expected drop in perceptual preference for the warp transition. The large rotation in this set produces the worst warp of all sets with many correspondence errors and unconvincing motion. However, it is still preferred over a cut. The dissolve, plane, APC and full 3D dynamic transitions all suffer static ghosting as either the frames are unregistered or the video registration is unsuccessful for the end video under shake. The plane and full 3D dynamic transitions additionally suffer large empty regions, but otherwise look quite similar — although the geometry is clearly better in the full 3D case with fewer shear artefacts, this improvement is difficult to spot through the shake and ghosting. The APC transition trades these empty regions for noisy point cloud, but as in set 7 this motion cue benefit is limited as there are few visual anchor points. Finally, the full 3D static transition has the highest perceptual score as it manages to remove static ghosting while still smoothly blending the camera motions from pan to shake. Figure B.9 shows the perceptual scores.



**Figure B.10:** Mean and 95% confidence error bars plotted on a perceptual scale for the different transition types for set 9. Perceptual scales are in z-score order, with the group mean at 0 and y-value representing multiples of the group standard deviation.

### Set 9: Scene 5, slight view change

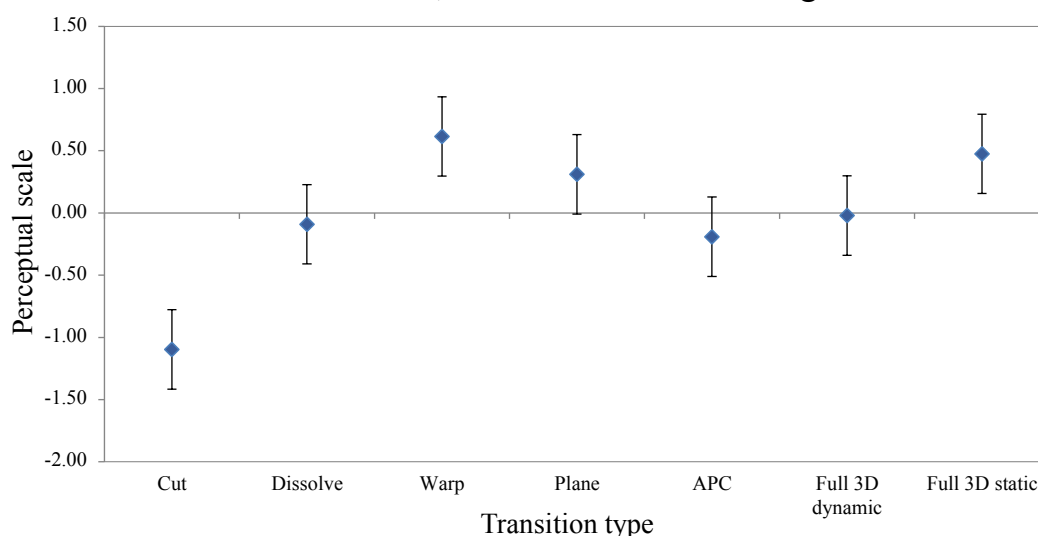
*Description:* Translate forward along modern glass/steel bridge changes to translate plus pan left.

*Features:* Camera shake and many people in foreground.

Sets 9 and 10 show our most complicated scene. Here, we have camera shake, complicated geometry, and dynamic objects in the foreground. Here, only the distant scene geometry is recovered. Given these conditions, we might expect a plane proxy to be equivalent to the distant recovered geometry. In set 9, the plane has a higher perceptual score than both full 3D and APC transitions — why is this? Under shake, we know that the video registration can be inaccurate, and this is true in this case. With geometry, this can lead to more than two examples of scene elements - one for the geometry, and potentially multiple for each projected video. In the plane case, even with inaccurate video registration, we only ever have a double image not a triple or quadruple image. Set 9 shows a skyline with spires. These cause considerably more noticeable static ghosting with triple images in the full 3D dynamic case than in the plane case as the spires contrast with the sky. The full 3D static case, while it has a slightly higher score, suffers from a geometry artefact around the edges of these spires which contrasts with the projected sky. We believe this causes it to be less preferred than the plane transition. The full 3D and APC transitions also suffer this geometry artefact, but the video registration inaccuracies dwarf this error.

Finally, the warp transition has the highest perceptual score as this transition can be well-estimated by an image zoom. Minor undulation occurs as the flow-based correction cannot entirely cope with the large zoom, but crucially it does not occur on the main visual anchor points of the skyline towers and dome as these are further into the distance. Figure B.10 shows the perceptual scores.

### Set 10: Scene 5, considerable view change



**Figure B.11:** Mean and 95% confidence error bars plotted on a perceptual scale for the different transition types for set 10. Perceptual scales are in z-score order, with the group mean at 0 and y-value representing multiples of the group standard deviation.

#### Set 10: Scene 5, considerable view change

*Description:* Pan left and zoom from bankside changes to pan left upon suspension bridge.

*Features:* Camera shake, travelling boat in middle distance and people in foreground.

Our final set swaps the shaky start clip from set 9 for a complicated pan and zoom shot which takes in a bridge across a river, upon which a boat is travelling. The view change represents a large translation and a more moderate rotation. The major scene buildings are still some way into the distance, but there is nearer complicated suspension bridge geometry. This distant scene geometry may explain why the warp transition performs so well in a considerable view change. Here, the distant visual anchors of the towers and dome are ghost-free throughout the warp transition. The rest of the scene undulates and dissolves unconvincingly, but this appears to be less important than the consistency of the most striking scene features. The motion in the warp also benefits from matching pans and does not look unnatural as in other cases. The plane transition also scores highly for the same reasons as in set 9. The APC and full 3D cases again suffer from geometry errors and so score less well. In set 10, we suggest that the full 3D static case performs slightly better than in set 9 because of the further view change. The added rotation allows the ghost-free geometry of the static case to better show the smooth camera motion, even though it still suffers from the geometry artefact around the edges of the towers. Figure B.11 shows the perceptual scores.

## Appendix C

# Transition Experiment Material

This section includes material from the transition experiment outlined in Section 6.3. Figures C.1 and C.2 visualize the website interface used by participants to rank video transitions. Figure C.3 shows the result sheet generated by the system for one participant, and Table C.1 shows the corresponding score for the example ranking. This score is used in the multi-dimensional scaling to translate the scores of all participants to a relative perceptual scale.

## Video Ranking Experiment

James Tompkin @ UCL, j.tompkin@cs.ucl.ac.uk

This experiment only works reliably in Firefox - if you are having trouble loading videos in another browser, please try with Firefox.  
This experiment also requires cookies.

You're using Firefox 6 on Windows!

## Scenario:

Imagine you wish to virtually tour a place, such as a city. You are using a new piece of software which can generate a video tour automatically. You select a path through the city on a map, and with one click the software produces a video tour which moves broadly along your chosen path. It does this by finding landmarks among a database of videos, and transitioning between the videos at these points.

Here is a short example of the kind of generated tour that you might see:



## Instructions:

In the following experiment, you are tasked with ranking a series of videos.  
Please rank the videos based on how often you would like to see each kind of transition in your tour.  
Placing a video at the top of the list means you would like to see it most often; the bottom of the list is least often.

Please play the video transitions, and change the ranking order by dragging and dropping the videos.  
There are 10 sets of videos to rank, and each set contains 7 5-second videos.  
Each set covers a different scene, and your judgement should be scene specific.  
If you prefer one transition over another for a particular scene, please rank it higher.

## Controls:

Use the next and previous buttons to move between sets.  
Each video has an associated symbol. These are there to help you remember which video is which.  
Please leave any comments for each set of videos in the box at the top of the page. These will be collected automatically.

When you've ranked the last set, press the 'Submit Result!' button to create an email to send me the results.  
Your rankings will not be lost as you move between sets.  
The experiment will take 30-60 minutes to complete.

**Note: there is an loading pause at the start and between sets - your browser has not crashed!**

Thank you very much!

Before you begin, please could you describe in a few words your level of expertise with computer graphics and media production:




**Figure C.1:** Image of the webpage which explains the experiment to participants. It includes an embedded video showing an example of the transitions that participants are likely to see (in this case, dissolve transitions between video clips that are unused elsewhere in the experiment). We also collect the self-assessed skill level of the participant in media production.



## Video Ranking Experiment

"How often would I like to see each of these video transitions in my automatic tour?"

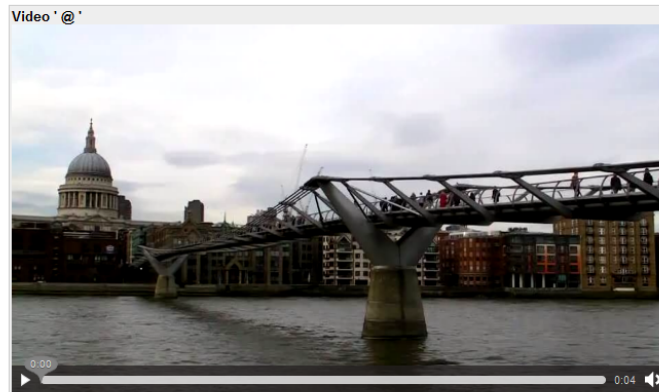
Ranking 1:

Any comments for this set?

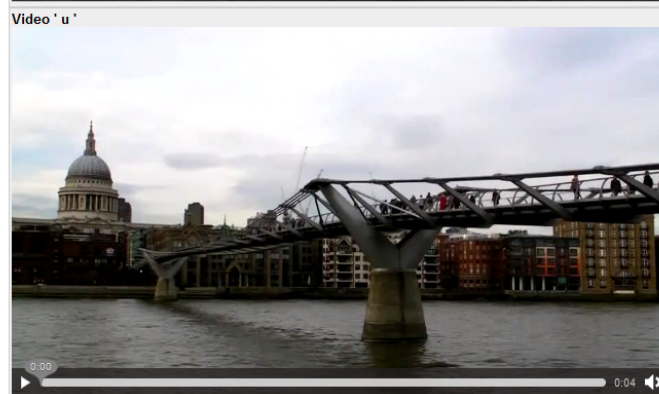
Previous Set Next Set

Previous Set Send Result! @ j.tompkin@cs.ucl.ac.uk

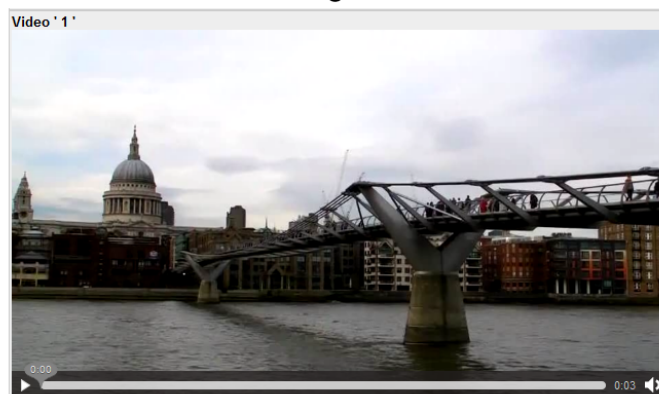
If the formed email doesn't work, please click Show Result and copy/paste into an email to me.



RANK 1 (Most often)



RANK 2



RANK 7 (Least often)

**Figure C.2:** Image of the webpage for ranking video transitions. Each transition type is randomly ordered onto the page. Participants can drag and drop the videos into order, using the ranking labels to the left to keep track. Comments can be left for each ranking, of which there are ten in total (five scenes, each with two view changes) shown in a random order. The region outlined in blue is replaced by the region outlined in red for the final ranking, allowing participants to submit their results remotely.

```

browser: Firefox 3.6 Windows
expertise: Average, not an expert in media production
setOrder: set9,set6,set2,set10,set8,set7,set3,set5,set1,set4
timeTaken(m): 35.891983333333336

set1: 3,6,2,7,5,4,1,
set1Time(m):2.44075
set1Comment: My favorite by far was '5' due to subtle change and
              pleasant feeling. [The full 3D static transition was
              visually labelled as video '5' in this case in the
              interface. '5' does not refer to the dissolve case.]

set2: 5,6,3,1,2,7,4,
set2Time(m):1.4257166666666667
set2Comment: As before.

set3: 7,6,4,3,2,5,1,
set3Time(m):0.9279333333333334
set3Comment: Any comments for this set?

set4: 3,7,1,5,2,4,6,
set4Time(m):2.0139333333333333
set4Comment: Any comments for this set?

set5: 5,6,7,2,3,1,4,
set5Time(m):1.21035
set5Comment: It's difficult to chose, I didn't like any of them.

set6: 3,2,5,7,6,1,4,
set6Time(m):1.86135
set6Comment: Skewing the video is really bad!

set7: 5,3,7,6,4,2,1,
set7Time(m):2.9397
set7Comment: Any comments for this set?

set8: 3,7,5,2,4,1,6,
set8Time(m):6.914083333333333
set8Comment: As before, pixelated and empty spaces are bad.

```

```

set9: 2,3,7,5,4,6,1,
set9Time(m):4.24735
set9Comment: I don't like seeing black areas, pixelated nor abrupt
              transitions. Fading from one video to another is more
              traditional approach, but I have to admit I like the
              3D effect of the other videos more to the top of my
              ranking.

set10: 3,2,5,7,4,6,1,
set10Time(m):2.3292333333333333
set10Comment: I really like the 3D effect of video.

```

**Figure C.3:** An example result sheet from the transition experiment (participant 5). Videos are labelled as follows: 1: Ambient Point Clouds. 2: Full 3D dynamic. 3: Full 3D static. 4: Plane. 5: Dissolve. 6: Cut. 7: Warp. Sets are paired into narrow then wide view changes. Sets 1 & 2: County Hall. Sets 3 & 4: Houses of Parliament. Sets 5 & 6: Portcullis House. Sets 7 & 8: Royal Albert Hall. Sets 9 & 10: Millennium Bridge.

Set	Time (m)	APC	Full3DDynamic	Full3DStatic	Plane	Blend	Cut	Warp
1	2.44	1	5	7	2	3	6	4
2	1.43	4	3	5	1	7	6	2
3	0.93	1	3	4	5	2	6	7
4	2.01	5	3	7	2	4	1	6
5	1.21	2	4	3	1	7	6	5
6	1.86	2	6	7	1	5	3	4
7	2.94	1	2	6	3	7	4	5
8	6.91	2	4	7	3	5	1	6
9	4.25	1	7	6	3	4	2	5
10	2.33	1	6	7	3	5	2	4
<b>Total</b>	<b>35.89</b>	<b>20</b>	<b>43</b>	<b>59</b>	<b>24</b>	<b>49</b>	<b>37</b>	<b>48</b>

**Table C.1:** This table shows an example scoring for the results from the participant shown in Figure C.3. The scoring is broken down by transition type, with each row for a particular set. The highest ranked transition type gains 7 points, reducing down to 1 for the lowest ranked transition type. All times are rounded to 2 decimal places. The total time is not necessarily equal to the sum of times of all individual rankings as the total time includes page loading times (each page loads seven videos, so depending on the bandwidth of the participant's Internet connection this time may not be short). In order of preference for this participant: Full 3D static, Blend, Warp, Full 3D dynamic, Cut, Plane, Ambient Point Clouds.

Appendix D

Spatial Awareness Experiment Material


This section includes material from the spatial awareness experiment outlined in Section 7.4.1. Figures D.1 and D.2 visualize the website interface used by participants in the experiment. Figure D.3 shows the result sheet generated by the system for one participant.

Video Experiment

James Tompkin @ UCL, j.tompkin@cs.ucl.ac.uk  
This experiment requires Javascript and cookies.

Task:


You will first see a map with a green pushpin and a triangle which represents a camera's field of view.



Next, a piece of video will play. The video contains two clips joined by a transition. The green triangle is the camera view from the first video clip just before the transition.

**Your task: Estimate the new position and view direction of the camera in the second video clip immediately after the transition.**

The map will return, and you must place a red pushpin where you think the second clip's camera was after the transition. Then, you must set the field of view for this camera.



We will first walk through an example to demonstrate the task.

[Go to example](#)

Final questions

Just a few questions before we finish the experiment:

You experienced two different interfaces: one with an initial green view (pushpin and triangle) and a 3D transition, and one with no initial green view (just a pushpin) and a cut transition.

**With which interface did you find it easiest to complete the task?**

☐ No green view/cut ☐ Green view/3D transition ☐ Both same

If one was easier, by how much?

☐ Slightly easier ☐ Easier ☐ Much easier ☐ Both same

**Which interface did you find provided the greatest spatial awareness and sense of orientation?**

☐ No green view/cut ☐ Green view/3D transition ☐ Both same

If one provided more, how much more?

☐ Slightly more ☐ More ☐ Much more ☐ Both same

**Were you with James when you completed this experiment?**

☐ Yes ☐ No

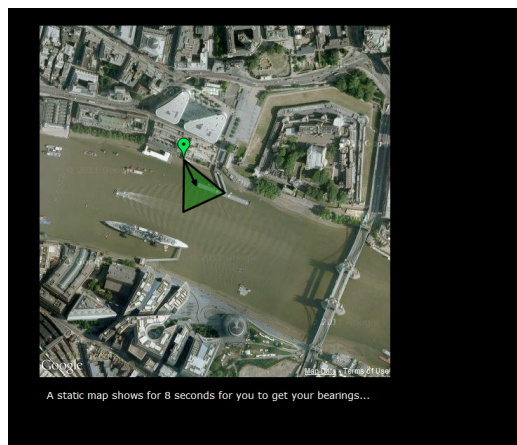
Finally, any other comments? *Please don't use line breaks in this box, sorry (no 'enter' key).*

Please click [Show Result](#) and copy/paste the contents into an email to me.

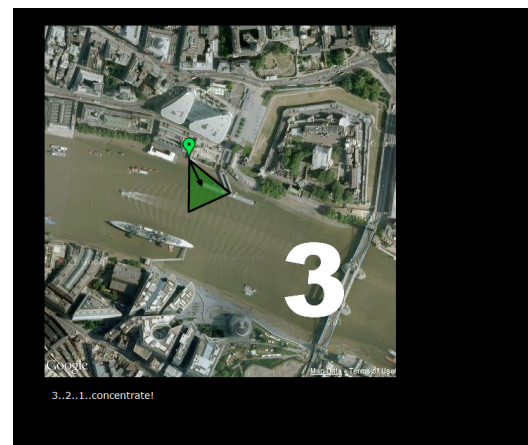
(a) Initial webpage with explanatory text.

(b) Questionnaire webpage.

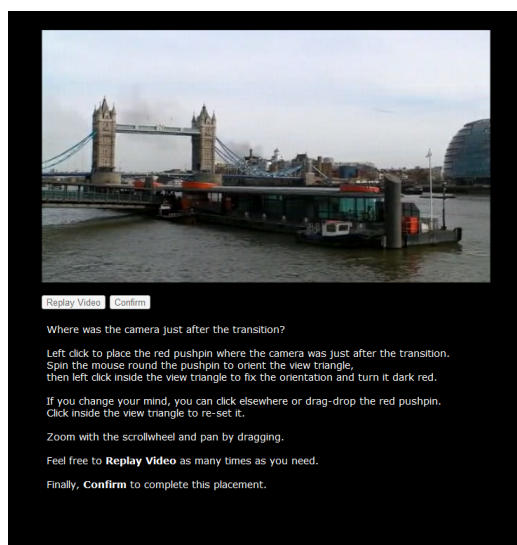
Figure D.1: Spatial awareness experiment website.



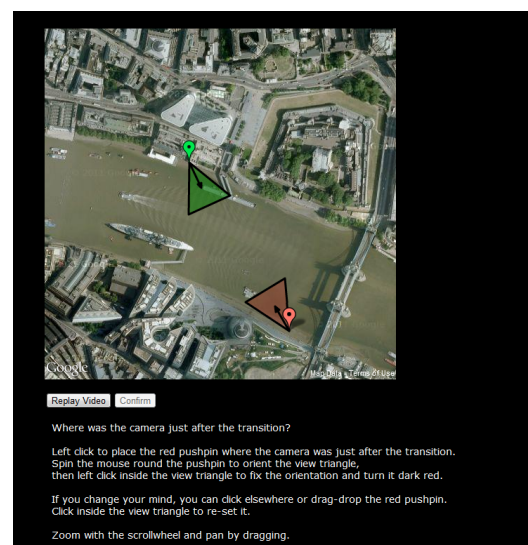
(a) Participants see a static map, pin, and view frustum for eight seconds as orientation.



(b) A countdown appears for three seconds.



(c) A video plays, then transitions into another video. This transports the viewer to a new world position and view direction. The two conditions in this experiment show either a cut or a 3D rendered transition.



(d) The participant marks on the map with the red pin/frustum from where they think the second video was taken.

**Figure D.2:** Spatial awareness experiment website.

```

VideoSpatialAwarenessExperiment JTompkin
browser: Chrome 14 Mac
name: << Anonymized >>
participantGroup: Group2::P.B::V.A
familiarity: Occasional visitor for many years.
              Now living in London (~3 weeks)
videoOrder: A2,B4,A3,A1,B3,B2,B1,A4

Example Positions: (51.50862467181381, -0.07809774337772524)
                  ::(51.50862467181381, -0.07809774337772524)
                  ::(51.50846440928814, -0.07824794708255922)
                  ::(51.50846440928814, -0.07824794708255922)
Example Bearings: NaN::-173.65974947159148::NaN::173.08873929137172
Example EntryTime: 1318341431805::1318341439124::1318341509214
                  ::1318341513020
Example StartTime: 1318341394362::1318341498585
Example EndTime: 1318341450513::1318341514313
Example ReplayCounter: 0
Example ReplayCounterTimes: 1318341406122::1318341417730

A1 Positions: (51.50098063568289, -0.12247466270446239)
              ::(51.50098063568289, -0.12247466270446239)
A1 Bearings: NaN::63.29417611132774
A1 EntryTime: 1318341835645::1318341839686
A1 StartTime: 1318341813553
A1 EndTime: 1318341841961
A1 ReplayCounter: 2
A1 ReplayCounterTimes: 1318341816018::1318341824594

<< Repeated for A2, A3, A4, B1, B2, B3, B4 >>

Question 1: Videoscapes
Question 1a: Easier
Question 2: Videoscapes
Question 2a: More
Comments: I know the South Bank and the Royal Albert Hall reasonably
          well so I don't know if that will have helped with the task.
SupervisedByJames: No

```

**Figure D.3:** An example result sheet from the spatial awareness experiment (participant 3).

## Appendix E

# Video Tour Experiment Material

This section includes material from the video tour summarization experiment outlined in Section 7.4.1. Figure E.1 visualizes the website interface used by participants in the experiment. Figure E.2 shows the result sheet generated by the system for one participant.



## Video Experiment

James Tompkin @ UCL, j.tompkin@cs.ud.ac.uk

This experiment requires Javascript and cookies.

## Task:

You will watch three short videos and answer a questionnaire about the presentation styles in those videos. These presentation styles are for summarizing video collections.

Each video shows a different summarization style. When watching the videos, try to concentrate on the **style of summarization** presented, not on any specific content, transitions or effects.

This experiment should take about 10 minutes.

Ready?

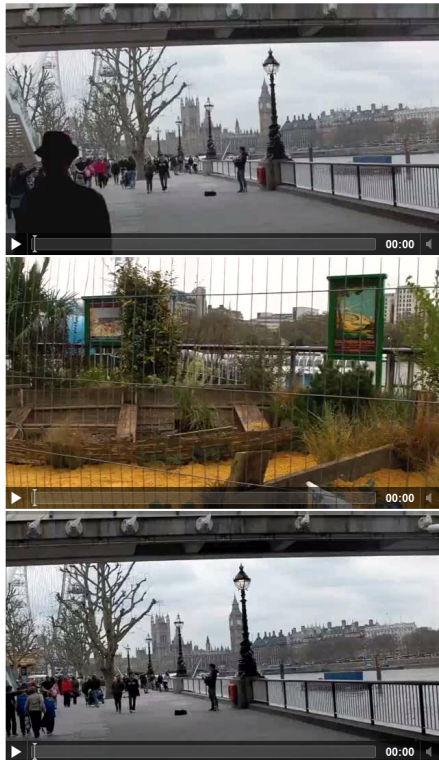
Please type your name (or an ID):



[Clear All Cookie Data \(Reset Experiment\)](#)

(a)

## Videos



(b)

## Questionnaire

You have just seen three videos (in a random order) which present different ways of summarizing videos and video collections:

- Smart fast-forward with normal speed at certain points (single video),
- Clips with similar content joined in sequence (many videos),
- Clips joined largely at random (many videos).

You are about to complete a questionnaire. Please try to ignore any specific contents or transitions in the videos that you may like or dislike, and to concentrate on the way in which the videos are summarized. **Please ignore the 'graphics' in the 'clips joined randomly' movie, such as the drawn images of Big Ben and the Leaning Tower of Pisa. Please also ignore any 'effects' on the videos themselves.**

Feel free to watch the videos as many times as you need.

1a) Which style of video summarization did you *most* prefer?

☐ Fast-forward ☐ Joined similar content ☐ Joined randomly

1b) Which style of video summarization did you *least* prefer?

☐ Fast-forward ☐ Joined similar content ☐ Joined randomly

2a) Which style of video summarization did you find *most* interesting?

☐ Fast-forward ☐ Joined similar content ☐ Joined randomly

2b) Which style of video summarization did you find *least* interesting?

☐ Fast-forward ☐ Joined similar content ☐ Joined randomly

3a) Which style of video summarization did you find provided the *best* sense of place?

☐ Fast-forward ☐ Joined similar content ☐ Joined randomly

3b) Which style of video summarization did you find provided the *worst* sense of place?

☐ Fast-forward ☐ Joined similar content ☐ Joined randomly

4a) Which style of video summarization did you find *most* spatially confusing?

☐ Fast-forward ☐ Joined similar content ☐ Joined randomly

4b) Which style of video summarization did you find *least* spatially confusing?

☐ Fast-forward ☐ Joined similar content ☐ Joined randomly

5a) Which style of video summarization would you use *most* often in your own video collections?

☐ Fast-forward ☐ Joined similar content ☐ Joined randomly

5b) Which style of video summarization would you use *least* often in your own video collections?

☐ Fast-forward ☐ Joined similar content ☐ Joined randomly

6a) Which style of video summarization would you view *most* often for online video collections (YouTube, etc.)?

☐ Fast-forward ☐ Joined similar content ☐ Joined randomly

6b) Which style of video summarization would you view *least* often for online video collections (YouTube, etc.)?

☐ Fast-forward ☐ Joined similar content ☐ Joined randomly

Finally, any other comments? All detail is welcome. Please don't use line breaks in this box, sorry (no 'enter' key).

Please click [Show Result](#) and copy/paste the contents into an email to me.

Thank you!

(c)

**Figure E.1:** Video tour summarization experiment website. (a) Initial webpage with explanatory text. (b) Videos which appear in a random order before the questionnaire. These can be replayed at will. (c) Questionnaire webpage.

VideoPreferenceSummarizationExperiment JTompkin

browser: Firefox 7 Windows

name: << Anonymized >>

order: 2,3,1 where 1=AutoMovie,2=Pongnumkul,3=Videoscapes

Question 1a: VS

Question 1b: AM

Question 2a: VS

Question 2b: AM

Question 3a: VS

Question 3b: AM

Question 4a: PK

Question 4b: VS

Question 5a: VS

Question 5b: AM

Question 6a: VS

Question 6b: AM

Comments: Depending upon how large the video collection is and how boring the individual videos are, I'm biased between fast-forward and joined similar content. For this particular example where videos were quite short, the joined similar content works brilliantly, but I can imagine a long boring wedding video where one might want to fast forward to the important moments and only see those in detail. On the other hand, the collage style random videos might be useful for achieving some sort of artistic or aesthetical purpose, but that's more a matter of taste rather than efficiency. Summing up, the best of all worlds in my vision would be to have a joined similar content video to give a story line to the whole collection but with the fast forward function added on top to skip to the highlights quickly.

**Figure E.2:** An example result sheet from the video tour summarization experiment (participant 10).

## Appendix F

# Video Browsing Experiment Material

This section includes material from the video browsing experiment outlined in Section 7.4.3. Figure F.1 visualizes the website interface used by participants in the experiment. Figure F.2 shows the result sheet generated by the system for one participant.

### Video Experiment

James Tompkin @ UCL, j.tompkin@cs.ucl.ac.uk

This experiment requires Javascript and cookies.

### Video Browsing Questionnaire

You have just been shown three different interfaces:

- iMovie,
- Yellow dots,
- Eyes with search.

You have also completed a short video browsing task to find related videos. Please would you answer the following questions:

Please type your name (or an ID):

1a) Which interface did you *most* prefer for completing the task of finding content?

☐ iMovie ☐ Yellow dots ☐ Eyes with search

1b) Which interface did you *least* prefer for completing the task of finding content?

☐ iMovie ☐ Yellow dots ☐ Eyes with search

2a) Which interface do you think you would *most* prefer for browsing content generally?

☐ iMovie ☐ Yellow dots ☐ Eyes with search

2b) Which interface do you think you would *least* prefer for browsing content generally?

☐ iMovie ☐ Yellow dots ☐ Eyes with search

3a) What did you think of the *iMovie* interface? How did you find the desired content?  
Please don't use line breaks (don't press 'return' or 'enter' in comment boxes).

3b) What did you think of the *yellow dots* interface? How did you find the desired content?

3c) What did you think of the *eyes with search* interface? How did you find the desired content?

4a) For the *eyes with search* interface, how useful were the eyes for the task?  
The eyes are placed 'above' the interesting location that they represent in the video collection.

☐ Very useful ☐ Somewhat useful ☐ Not useful ☐ Did not use

4b) In general, how useful do you think the eyes would be for browsing video collections?

☐ Very useful ☐ Somewhat useful ☐ Not useful

5a) For the *eyes with search* interface, how useful were the gray trails showing from where the video was taken?  
These are the direct path that the camera travelled.

☐ Very useful ☐ Somewhat useful ☐ Not useful ☐ Did not use

5b) In general, how useful do you think the gray trails would be for browsing video collections?

☐ Very useful ☐ Somewhat useful ☐ Not useful

6a) For the *eyes with search* interface, how useful were the white camera field of views showing from where the video was taken?  
These white triangles show the direction that the camera is pointing in along with the position.

☐ Very useful ☐ Somewhat useful ☐ Not useful ☐ Did not use

6b) In general, how useful do you think the white camera field of views would be for browsing video collections?

☐ Very useful ☐ Somewhat useful ☐ Not useful

7a) For the *eyes with search* interface, how useful was the search box for text searches? The search box allows keyword text search as well as image search to find similar content.

☐ Very useful ☐ Somewhat useful ☐ Not useful ☐ Did not use

7b) In general, how useful do you think the search box for text searches would be for browsing video collections?

☐ Very useful ☐ Somewhat useful ☐ Not useful

8a) For the *eyes with search* interface, how useful was the search box for image searches? The search box allows keyword text search as well as image search to find similar content.

☐ Very useful ☐ Somewhat useful ☐ Not useful ☐ Did not use

8b) In general, how useful do you think the search box for image searches would be for browsing video collections?

☐ Very useful ☐ Somewhat useful ☐ Not useful

9a) Do you think you would want to use the *eyes with search* interface for browsing your own personal video collections?

☐ Yes often ☐ Yes sometimes ☐ Yes rarely ☐ No

9b) Do you think you would want to use the *eyes with search* interface for browsing online video collections (Youtube, etc.)?

☐ Yes often ☐ Yes sometimes ☐ Yes rarely ☐ No

Finally, any other comments? Please don't use line breaks in this box, sorry (no 'enter' key).

Please click [Show Result](#) and copy/paste the contents into an email to me.

Thank you!

[Clear All Cookie Data](#)

**Figure F.1:** Questionnaire website for the video browsing experiment. Page appears as one column online.

VideoBrowseFindingContentExperiment JTompkin

browser: Chrome 15 Windows

name: << Anonymized >>

Question 1a: VS

Question 1b: IM

Question 2a: VS

Question 2b: IM

Question 3a: I think it depends on chance since you require the selected thumbnails to contain the image, otherwise it's necessary to scrub through which is slow.

Question 3b: Useful to have the thumbnails but again, they depend on the start of the video - having to wait for the video to play slows you down and no prior knowledge of viewing direction doesn't help.

Question 3c: The easiest since different methods of searching and also jumping to the correct frame dramatically increases the speed.

Question 4a: Very useful

Question 4b: Very useful

Question 5a: Somewhat useful

Question 5b: Somewhat useful

Question 6a: Very useful

Question 6b: Very useful

Question 7a: Very useful

Question 7b: Very useful

Question 8a: Very useful

Question 8b: Very useful

Question 9a: Yes often

Question 9b: Yes often

Comments: Obviously the search functions are very useful, I find the field of view particularly useful since it is often missing (for example the yellow dots).

**Figure F.2:** An example result sheet from the video browsing experiment (participant 3).

## Bibliography

- [AB91] Edward H. Adelson and James R. Bergen. The Plenoptic Function and the Elements of Early Vision. *Computational Models of Visual Processing*, 1991.
- [Ado12a] Adobe. Premiere, 2012.
- [Ado12b] Adobe. Premiere Elements, 2012.
- [ADS12] Juha Alakarhu, Damian Dinning, and Eero Salmelin. PureView Imaging Technology. *Nokia*, pages 1–10, 2012.
- [AFYC03] Daniel G. Aliaga, Thomas Funkhouser, Dimah Yanovsky, and Ingrid Carlbom. Sea of images. *IEEE Computer Graphics and Applications*, 23(6):22–30, 2003.
- [ALS07] Naveed Ahmed, Hendrik Lensch, and Hans-Peter Seidel. Seeing People in Different Light-Joint Shape, Motion, and Reflectance Capture. *IEEE Transactions on Visualization and Computer Graphics*, 13(4):663–674, 2007.
- [App11] Apple Inc. iMovie ’11, 2011.
- [App12] Apple Inc. Final Cut, 2012.
- [ASS<sup>+</sup>09] Sameer Agarwal, Noah Snavely, Ian Simon, Steven M Seitz, and Richard Szeliski. Building Rome in a Day. *2009 IEEE 12th International Conference on Computer Vision*, pages 72–79, September 2009.
- [ATS07] Naveed Ahmed, Christian Theobalt, and Hans-Peter Seidel. Spatio-temporal reflectance sharing for relightable 3D video. *Computer Vision/Computer Graphics Collaboration Techniques*, pages 47–58, 2007.
- [Aut08] Inc. Autodesk. ImageModeler. Software, 2008.
- [Avi12] Avid. Media Composer, 2012.
- [BBP04] Thomas Brox, Andrés Bruhn, and Nils Papenberg. High Accuracy Optical Flow Estimation based on a Theory for Warping. *European Conference on Computer Vision (ECCV)*, 4(May):25–36, 2004.

- [BBPP10] Luca Ballan, Gabriel J. Brostow, Jens Puwein, and Marc Pollefeys. Unstructured Video-based Rendering: Interactive Exploration of Casually Captured Videos. *ACM Transactions on Graphics*, 29(4):1, July 2010.
- [BG01] Samuel Boivin and Andre Gagalowicz. Image-based Rendering of Diffuse, Specular and Glossy Surfaces From a Single Image. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques - SIGGRAPH '01*, pages 107–116, New York, NY, USA, 2001. ACM.
- [BKC<sup>+</sup>10] Georges Baatz, Kevin Köser, David Chen, Radek Grzeszczuk, and Marc Pollefeys. Handling Urban Location Recognition as a 2D Homothetic Problem. In *Proc. ECCV*, pages 266–279. Springer, 2010.
- [BKM<sup>+</sup>07] David G. Bell, Frank Kuehnel, Chris Maxwell, Randy Kim, Kushyar Kasraie, Tom Gaskins, Patrick Hogan, and Joe Coughlan. NASA World Wind: Opensource GIS for Mission Operations. In *Proc. IEEE Aerospace Conference*, pages 1–9, March 2007.
- [BN92] Thaddeus Beier and Shawn Neely. Feature-based Image Metamorphosis. *ACM SIGGRAPH Computer Graphics*, 26(2):35–42, July 1992.
- [BSFG09] Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B. Goldman. PatchMatch: A Randomized Correspondence Algorithm for Structural Image Editing. In *ACM Transactions on Graphics (TOG)*, volume 28, page 24. ACM, 2009.
- [BSGF10] Connelly Barnes, Eli Shechtman, Dan B. Goldman, and Adam Finkelstein. The Generalized Patchmatch Correspondence Algorithm. *European Conference on Computer Vision (ECCV)*, pages 29–43, 2010.
- [BZS<sup>+</sup>07] Pravin Bhat, C. Lawrence Zitnick, Noah Snavely, Aseem Agarwala, Maneesh Agrawala, Michael Cohen, Brian Curless, and Sing Bing. Using Photographs to Enhance Videos of a Static Scene. *Eurographics Symposium on Rendering*, 2007.
- [CCMV07] Gustavo Carneiro, Antoni B. Chan, Pedro J. Moreno, and Nuno Vasconcelos. Supervised Learning of Semantic Classes for Image Annotation and Rretrieval. *IEEE TPAMI*, 29(3):394–410, 2007.
- [CCT<sup>+</sup>09] Tao Chen, Ming-Ming Cheng, Ping Tan, Ariel Shamir, and Shi-Min Hu. Sketch2Photo. *ACM Transactions on Graphics*, 28(5):1, December 2009.
- [CDF<sup>+</sup>04] Gabriella Csurka, Christopher R. Dance, Lixin Fan, Jutta Willamowski, and Cédric Bray. Visual Categorization with Bags of Keypoints. In *Proc. ECCV*, volume 1, pages 1–22, 2004.
- [CEJ<sup>+</sup>06] Charles-Félix Chabert, Per Einarsson, Andrew Jones, Bruce Lamond, Wan-Chun Ma, Sebastian Sylwan, Tim Hawkins, and Paul Debevec. Relighting Human Locomotion with

Flowed Reflectance Fields. *ACM SIGGRAPH 2006 Sketches on - SIGGRAPH '06*, page 76, 2006.

- [Che95] Shenchang Eric Chen. QuickTime VR: An Image-based Approach to Virtual Environment Navigation. In *SIGGRAPH '95: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 29–38, New York, NY, USA, 1995. ACM Press.
- [Chr06] Michael G. Christel. Evaluation and User Studies with Respect to Video Summarization and Browsing. *Proceedings of SPIE*, 6073:60730M–60730M–15, 2006.
- [Con11] Contour. Contour GPS Camera Software, 2011.
- [COSH11] David Crandall, Andrew Owens, Noah Snavely, and Dan Huttenlocher. Discrete-continuous Optimization for Large-scale Structure from Motion. *Computer Vision and Pattern Recognition*, 286(26):3001–3008, 2011.
- [CRC<sup>+</sup>11] Paulo Cignoni, Guido Ranzuglia, Marco Callieri, Massimiliano Corsini, Matteo Dellepiane, Fabio Ganovelli, Nico Pietroni, and Marco Tarini. MeshLab, 2011.
- [CSDI11] Gaurav Chaurasia, Olga Sorkine, George Drettakis, and Reves Inria. Silhouette-aware Warping for Image-Based Rendering. In *Eurographics Symposium on Rendering*, volume 30, 2011.
- [CTMS03] Joel Carranza, Christian Theobalt, Marcus Magnor, and Hans-Peter Seidel. Free-viewpoint Video of Human Actors. In *SIGGRAPH '03: Proceedings of the 30th Annual Conference on Computer Graphics and Interactive Techniques*, 2003.
- [CW93] Shenchang Eric Chen and Lance Williams. View Interpolation for Image Synthesis. *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques - SIGGRAPH '93*, pages 279–288, 1993.
- [Cya93] Cyan Worlds. Myst, 1993.
- [Cyt12] Cytoscape Consortium. Cytoscape: An Open Source Platform for Complex Network Analysis and Visualization, 2012.
- [Deb81] Ryoichiro Debuchi. Frozen Time, 1981.
- [Deb97] Paul Debevec. The Campanile Movie, 1997.
- [Dij59] Edsger W. Dijkstra. A Note on Two Problems in Connexion with Graphs. *Numerische Mathematik*, 1(1):269–271, 1959.
- [Dir11] Tim Dirks. Film Milestones in Visual/Special Effects, 2011.
- [DJLW08] Ritendra Datta, Dhiraj Joshi, Jia Li, and James Ze Wang. Image Retrieval: Ideas, Influences, and Trends of the New Age. *ACM Computing Surveys*, 40(2), 2008.



- [Dmy84] Edward Dmytryk. *On Film Editing*. 1984.
- [DTM96] Paul E. Debevec, Camillo J. Taylor, and Jitendra Malik. Modeling and Rendering Architecture from Photographs: A Hybrid Geometry- and Image-based Approach. In *SIGGRAPH*, pages 11–20, 1996.
- [DYB98] Paul Debevec, Yizhou Yu, and George Borshukov. Efficient View-dependent Image-based Rendering with Projective Texture-mapping. In *Rendering Techniques 98: Proceedings of the Eurographics Workshop*, number CSD-98-1003, page 14, Vienna, Austria, 1998. University of California at Berkeley.
- [EDM<sup>+</sup>08] Martin Eisemann, Bert De Decker, Marcus Magnor, Philippe Bekaert, Edilson de Aguiar, Naveed Ahmed, Christian Theobalt, and Anita Sellent. Floating Textures. *Computer Graphics Forum*, 27(2):409–418, April 2008.
- [ELF97] David W. Eggert, Adele Lorusso, and Robert B. Fisher. Estimating 3-D Rigid Body Transformations: A Comparison of Four Major Algorithms. *Machine Vision and Applications*, 9(5-6):272–290, March 1997.
- [Eve09] Everyscape.com. Everyscape, 2009.
- [Far03] Gunnar Farneback. Two-frame Motion Estimation based on Polynomial Expansion. In *Proc. SCIA*, pages 363–370, 2003.
- [FB81] Martin A. Fischler and Robert C. Bolles. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Communications of the ACM*, 24(6):381–395, 1981.
- [FCSS10] Y. Furukawa, B. Curless, Steven M. Seitz, and R. Szeliski. Towards Internet-scale Multi-view Stereo. In *Proc. IEEE CVPR*, pages 1434–1441, June 2010.
- [FGG<sup>+</sup>10] Jan-Michael Frahm, Pierre Georgel, David Gallup, Tim Johnson, Rahul Raguram, Changchang Wu, Yi-Hung Jen, Enrique Dunn, Brian Clipp, Svetlana Lazebnik, Marc Pollefeys, and Pierre Fite-Georgel. Building Rome on a Cloudless Day. In *European Conference on Computer Vision (ECCV)*, pages 368–381, 2010.
- [Fin02] David Fincher. *Panic Room*, 2002.
- [FP10] Yasutaka Furukawa and Jean Ponce. Accurate, Dense, and Robust Multi-view Stereopsis. *IEEE TPAMI*, 32(8):1362–1376, August 2010.
- [FPL<sup>+</sup>10] Jan-Michael Frahm, Marc Pollefeys, Svetlana Lazebnik, David Gallup, Brian Clipp, Rahul Raguram, Changchang Wu, Christopher Zach, and Tim Johnson. Fast Robust Large-scale Mapping from Video and Internet Photo Collections. *ISPRS Journal of Photogrammetry and Remote Sensing*, 65(6):538–549, 2010.

- [Fri03] Doron Friedman. *Knowledge-based Formalization of Cinematic Expression and its Application to Animation*. PhD thesis, 2003.
- [FSN<sup>+</sup>95] Myron Flickner, Harpreet Sawhney, Wayne Niblack, Jonathan Ashley, Qian Huang, Byron Dom, Monika Gorkani, Jim Hafner, Denis Lee, Dragutin Petkovic, David Steele, and Peter Yanker. Query by Image and Video Content: The QBIC System. *Computer*, 28:23–32, 1995.
- [GAF<sup>+</sup>10] Michael Goesele, Jens Ackermann, Simon Fuhrmann, Carsten Haubold, and Ronny Klowsky. Ambient Point Clouds for View Interpolation. *ACM Transactions on Graphics (TOG)*, 29(4):1–6, 2010.
- [GBQV09] Stephan Gammeter, Lukas Bossard, Till Quack, and Luc Van Gool. I Know What You Did Last Summer: Object-level Auto-annotation of Holiday Snaps. In *Proc. ICCV*, 2009.
- [Ger39] Alexander Gershun. The Light Field. *Journal of Mathematics and Physics*, 18(2):51–151, 1939.
- [GFT11] Abhijeet Ghosh, Graham Fyffe, and Borom Tunwattanapong. Multiview Face Capture Using Polarized Spherical Gradient Illumination. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia)*, 30(6):1–10, 2011.
- [GGSC96] Steven J. Gortler, Radek Grzeszczuk, Richard Szeliski, and Michael F. Cohen. The Lumigraph. *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques - SIGGRAPH '96*, pages 43–54, 1996.
- [GH10] Jean-Yves Guillemaut and Adrian Hilton. Joint Multi-layer Segmentation and Reconstruction for Free-Viewpoint Video Applications. *International Journal of Computer Vision*, 93(1):73–100, December 2010.
- [GKE11] Matthias Grundmann, Vivek Kwatra, and Irfan Essa. Auto-directed Video Stabilization with Robust L1 Optimal Camera Paths. *Computer Vision and Pattern Recognition*, (1):225–232, June 2011.
- [GKT<sup>+</sup>12] Miguel Granados, Kwang In Kim, James Tompkin, Jan Kautz, and Christian Theobalt. Background Inpainting for Videos with Dynamic Objects and a Free-moving Camera. In *European Conference on Computer Vision (ECCV)*, 2012.
- [Glo09] GlobalVision. Steetview Switzerland, 2009.
- [Goo07] Google. Street View, 2007.
- [Goo08] Google. Panoramio Look Around, 2008.
- [Goo10] Google. Quiksee, 2010.
- [Goo12a] Google. Google Maps, 2012.

- [Goo12b] Google. YouTube, 2012.
- [Gra04] Oliver Grau. 3D Sequence Generation from Multiple Cameras. In *Multimedia Signal Processing, IEEE 6th Workshop on*, pages 391–394, 2004.
- [Gra06] Oliver Grau. Multi-camera Radiometric Surface Modelling for Image-Based Re-lighting. In *DAGM-Symposium*, pages 667–676, 2006.
- [Gra11] Oliver Grau. Fast Volumetric Visual Hull Computation. Technical Report November, BBC R&D, 2011.
- [GS11] Andreas Girgensohn and Frank Shipman. Adaptive Clustering and Interactive Visualizations to Support the Selection of Video Clips. *Proceedings of the 1st ACM International Conference on Multimedia Retrieval*, 2011.
- [GSC<sup>+</sup>07] Michael Goesele, Noah Snavely, Brian Curless, Hugues Hoppe, and Steven M. Seitz. Multi-view Stereo for Community Photo Collections. *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8, 2007.
- [GTK<sup>+</sup>12] Miguel Granados, James Tompkin, Kwang In Kim, Oliver Grau, Jan Kautz, and Christian Theobalt. How Not to Be Seen - Object Removal from Videos of Crowded Scenes. *Computer Graphics Forum*, 31(2pt1):219–228, May 2012.
- [Gui54] Joy Paul Guilford. *Psychometric Methods*. McGraw-Hill Book Company, 2nd edition, 1954.
- [HAA97] Youichi Horry, Ken-ichi Anjyo Anjyo, and Kiyoshi Arai. Tour Into The Picture: Using a Spidery Mesh Interface to make Animation from a Single Image. In *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques*, pages 225–232. ACM Press/Addison-Wesley Publishing Co., 1997.
- [Haw07] Hawk-eye Innovations Ltd. Hawk-eye, 2007.
- [HGK<sup>+</sup>11] Adrian Hilton, Jean-Yves Guillemaut, Joe Kilner, Oliver Grau, and Graham Thomas. 3D-TV Production From Conventional Cameras for Sports Broadcast. *IEEE Transactions on Broadcasting*, 57(2):462–476, June 2011.
- [HGO<sup>+</sup>10] Kyle Heath, Natasha Gelfand, Maks Ovsjanikov, Mridul Aanjaneya, and Leonidas J. Guibas. Image Webs: Computing and Exploiting Connectivity in Image Collections. In *Proc. IEEE CVPR*, pages 3432–3439. IEEE, 2010.
- [Hit48] Alfred Hitchcock. Rope, 1948.
- [Hit63] Alfred Hitchcock. The Birds, 1963.
- [HMB11] Ahmad Humayun, Oisín Mac Aodha, and Gabriel J. Brostow. Learning to Find Occlusion Regions. In *Computer Vision and Pattern Recognition*, 2011.

- [Hor87] Berthold K. P. Horn. Closed-form Solution of Absolute Orientation using Unit Quaternions. *Journal of the Optical Society of America A*, 4(4):629, April 1987.
- [How88] Ron Howard. Willow, 1988.
- [HRT<sup>+</sup>09] Nils Hasler, Bodo Rosenhahn, Thorsten Thorm, Michael Wand, Juergen Gall, and Hanspeter Seidel. Markerless Motion Capture with Unsynchronized Moving Cameras. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2009)*, Miami Beach, Florida, 2009.
- [HS81] Berthold K. P. Horn and Brian G. Schunck. Determining Optical Flow. *Artificial Intelligence*, 17(1-3):185–203, August 1981.
- [HS92] Jim Hollan and Scott Stornetta. Beyond Being There. In R M Baecker, editor, *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 119–125. ACM Press, 1992.
- [HS93] Berthold K. P. Horn and B.G. Schunck. Determining optical flow: a retrospective. *Artificial Intelligence*, 59(1-2):81–87, February 1993.
- [HS04] Adrian Hilton and Jonathan Starck. Multiple View Reconstruction of People. In *3D Data Processing, Visualization and Transmission, 2004. 3DPVT 2004. Proceedings. 2nd International Symposium on*, pages 357–364, 2004.
- [HS12] Daniel Cabrini Hauagge and Noah Snavely. Image Matching using Local Symmetry Features. *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 206–213, June 2012.
- [HZ04] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004.
- [IBL00] Yuri A. Ivanov, Aaron F. Bobick, and John Liu. Fast Lighting Independent Background Subtraction. *International Journal on Computer Vision*, 37(2):7–199, 2000.
- [IBR<sup>+</sup>09] INA, BBC, RAI, JRS, B&G, and ORF. EU FP7 Project: PrestoPRIME, 2009.
- [IMH05] Takeo Igarashi, Tomer Moscovich, and John F. Hughes. As-rigid-as-possible Shape Manipulation. *ACM Transactions on Graphics*, 24(3):1134, July 2005.
- [Kat92] Toshikazu Kato. Database Architecture for Content-based Image Retrieval. *Proc. SPIE*, pages 112–123, 1992.
- [KBH06] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson Surface Reconstruction. In *Proc. Eurographics Symposium on Geometry Processing*, pages 61–70, New York, NY, USA, 2006. Eurographics Association.

- [KCS10] Johannes Kopf, Billy Chen, and Richard Szeliski. Street Slide: Browsing Street Level Imagery. *ACM Transactions on Graphics*, 2010.
- [KF01] Don Kimber and J Foote. Flyabout: Spatially Indexed Panoramic Video. *ACM Transactions on Multimedia Computing, Communications, and Applications*, pages 339–347, 2001.
- [KGV83] Scott Kirkpatrick, C. Daniel Gelatt, and Mario P. Vecchi. Optimization by Simulated Annealing. *Science*, 220(4598):671–680, 1983.
- [KL02] Risi I. Kondor and John Lafferty. Diffusion Kernels on Graphs and Other Discrete Structures. In *Proc. ICML*, pages 315–322, 2002.
- [KN08] Lyndon Kennedy and Mor Naaman. Generating Diverse and Representative Image Search Results for Landmarks. In *Proc. WWW*, pages 297–306, 2008.
- [KN09] Lyndon Kennedy and Mor Naaman. Less Talk, More Rock: Automated Organization of Community-contributed Collections of Concert Videos. In *Proc. of WWW*, pages 311–320, 2009.
- [KNC<sup>+</sup>08] Johannes Kopf, Boris Neubert, Billy Chen, Michael Cohen, Daniel Cohen-Or, Oliver Deussen, Matt Uyttendaele, and Dani Lischinski. Deep Photo: Model-based Photograph Enhancement and Viewing. *ACM Transactions on Graphics*, 27(5):1, 2008.
- [KRE<sup>+</sup>] Christian Kurz, Tobias Ritschel, Elmar Eisemann, Torsten Thormahlen, and Hans-Peter Seidel. Camera Motion Style Transfer. In *Visual Media Production (CVMP), 2010 Conference on*, pages 9–16. IEEE.
- [KS02] Hyung Woo Kang and Sung Yong Shin. Tour Into The Video: Image-based Navigation Scheme for Video Sequences of Dynamic Scenes. In *Proceedings of the ACM Symposium on Virtual Reality Software and Technology*, pages 73–80. ACM, 2002.
- [KSV98] Takeo Kanade, Hideo Saito, and Sundar Vedula. The 3D Room: Digitizing Time-varying 3D Events by Synchronized Multiple Video Streams. Technical report, Carnegie-Mellon University, 1998.
- [KTT<sup>+</sup>12] Kwang In Kim, James Tompkin, Martin Theobald, Jan Kautz, and Christian Theobald. Match Graph Construction for Large Image Databases. In *European Conference on Computer Vision (ECCV)*, 2012.
- [KV10] Eitam Kav-Venaki. Feedback Retargeting. In *Media Retargeting Workshop at ECCV*, 2010.
- [KWO10] Michael Kroepfl, Y Wexler, and E Ofek. Efficiently Locating Photographs in Many Panoramas. *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems - GIS '10*, (c), 2010.

- [LA04] Manolis I. A. Lourakis and Antonis A. Argyros. The Design and Implementation of a Generic Sparse Bundle Adjustment Software Package based on the Levenberg-Marquardt Algorithm. *ICSFORTH Technical Report TR*, 340(340), 2004.
- [Lau94] Aldo Laurentini. The Visual Hull Concept for Silhouette-based Image Understanding. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(2):150–162, 1994.
- [LFDF07] Anat Levin, Rob Fergus, Frédo Durand, and William T. Freeman. Image and Depth from a Conventional Camera with a Coded Aperture. *ACM Transactions on Graphics*, 26(3):70, July 2007.
- [LGJA09] Feng Liu, Michael Gleicher, Hailin Jin, and Aseem Agarwala. Content-preserving Warps for 3D Video Stabilization. *ACM Transactions on Graphics*, 28(3):1, July 2009.
- [LGW<sup>+</sup>11] Feng Liu, Michael Gleicher, Jue Wang, Hailin Jin, and Aseem Agarwala. Subspace Video Stabilization. *ACM Transactions on Graphics*, 1(212):1–10, 2011.
- [LH96] Marc Levoy and Pat Hanrahan. Light Field Rendering. *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques - SIGGRAPH '96*, 30(Annual Conference Series):31–42, 1996.
- [LH05] Marius Leordeanu and Martial Hebert. A Spectral Technique for Correspondence Problems using Pairwise Constraints. In *Proc. ICCV*, pages 1482–1489, 2005.
- [Lib07] LiberoVision AG. LiberoVision, 2007.
- [Lin06] Richard Linklater. A Scanner Darkly, 2006.
- [Lip80] Andrew Lippman. Movie-maps: An Application of the Optical Videodisc to Computer Graphics. *ACM SIGGRAPH Computer Graphics*, 14(3):32–42, 1980.
- [LLB<sup>+</sup>10] Christian Lipski, Christian Linz, Kai Berger, Anita Sellent, and Marcus Magnor. Virtual Video Camera: Image-based Viewpoint Navigation Through Space and Time. *Computer Graphics Forum*, 29(8):2555–2568, 2010.
- [LLN<sup>+</sup>10] Christian Lipski, Christian Linz, Thomas Neumann, Markus Wacker, and Marcus Magnor. High Resolution Image Correspondences for Video Post-Production. In *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, pages 33–39. IEEE, 2010.
- [LM01] Thomas Leung and Jitendra Malik. Representing and Recognizing the Visual Appearance of Materials using Three-dimensional Textons. *IJCV*, 43(1):29–44, 2001.
- [Low04] David G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60(2):91–110, November 2004.

- [LSDJ06] Michael S. Lew, Nicu Sebe, Chabane Djeraba, and Ramesh Jain. Content-based Multimedia Information Retrieval: State of the Art and Challenges. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 2006.
- [LSP06] Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories. In *Proc. IEEE CVPR*, pages 2169–2178, 2006.
- [LTK12a] Philippe Levieux, James Tompkin, and Jan Kautz. Interactive Viewpoint Video Textures. In *Visual Media Production (CVMP), 2012 Conference on*, 2012.
- [LTK12b] Henrik Lieng, James Tompkin, and Jan Kautz. Interactive Multi-perspective Imagery from Photos and Videos. *Computer Graphics Forum*, 31(2pt1):285–293, May 2012.
- [LWZ<sup>+</sup>08] Xiaowei Li, Changchang Wu, Christopher Zach, Svetlana Lazebnik, and Jan-Michael Frahm. Modeling and Recognition of Landmark Image Collections using Iconic Scene Graphs. In *Proc. ECCV*, pages 427–440, 2008.
- [LYT<sup>+</sup>08] Ce Liu, Jenny Yuen, Antonio Torralba, Josef Sivic, and William T. Freeman. Sift Flow: Dense Correspondence Across Different Scenes. *European Conference on Computer Vision (ECCV)*, pages 28–42, 2008.
- [Mac80] Tim Macmillan. Early Work 1980-1994, 1980.
- [MB95] Leonard McMillan and Gary Bishop. Plenoptic Modeling: An Image-based Rendering System. In *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques*, volume 95, pages 39–46. ACM, 1995.
- [MB06] Harvey J. Motulsky and Ronald E. Brown. Detecting Outliers When Fitting Data with Nonlinear Regression - A New Method based on Robust Nonlinear Regression and the False Discovery Rate. *BMC Bioinformatics*, 7:123, January 2006.
- [McC94] Scott McCloud. *Understanding Comics*. Harper Perennial, 1994.
- [McC07] Neil J. McCurdy. *RealityFlythrough: A System for Ubiquitous Video*. Ph.d., University of California, San Diego, 2007.
- [MCG05] Neil J. McCurdy, Jennifer N. Carlisle, and William G. Griswold. Harnessing Mobile Ubiquitous Video. *CHI '05 Extended Abstracts on Human Factors in Computing Systems - CHI '05*, page 1645, 2005.
- [MCUP04] Jiri Matas, Ondrej Chum, Martin Urban, and Tomáš Pajdla. Robust Wide-baseline Stereo from Maximally Stable Extremal Regions. *Image and Vision Computing*, 22(10):761–767, September 2004.



- [MGL06] Neil J. McCurdy, William Griswold, and Leslie Lenert. A Robust Abstraction for First-Person Video Streaming: Techniques, Applications, and Experiments. *8th IEEE International Symposium on Multimedia (ISM'06)*, pages 235–244, 2006.
- [MHS05] Gregor Miller, Adrian Hilton, and Jonathan Starck. Interactive Free-Viewpoint Video. In *Visual Media Production (CVMP), 2005 Conference on*, pages 52–61, 2005.
- [Mic08] Microsoft. Photosynth, 2008.
- [Mic11] Microsoft. Bing Maps Streetside, 2011.
- [Mic12] Microsoft. Windows Live Movie Maker, 2012.
- [MKC07] Ricardo Marroquim, Martin Kraus, and Paulo Roma Cavalcanti. Efficient Point-based Rendering using Image Reconstruction. *Eurographics Symposium on Point-Based Graphics*, 2007.
- [MNBN07] Francesc Moreno-Noguer, Peter N. Belhumeur, and Shree K. Nayar. Active refocusing of images and videos. *ACM Transactions on Graphics*, 26(3):67, July 2007.
- [MO09] Yann Morvan and Carol O’Sullivan. Handling Occluders in Transitions from Panoramic Images: A Perceptual Study. *ACM Trans. Applied Perception*, 6(4):1–15, 2009.
- [Mok06] Mok3 Inc. SuperTour Travel, 2006.
- [MPC<sup>+</sup>05] Marcus Magnor, Marc Pollefeys, German Cheung, Wojciech Matusik, and Christian Theobalt. Video-based Rendering. SIGGRAPH Course, 2005.
- [MS04] Krystian Mikolajczyk and Cordelia Schmid. Scale & Affine Invariant Interest Point Detectors. *International Journal of Computer Vision*, 60(1):63–86, 2004.
- [MSH06] G. Miller, J. Starck, and a. Hilton. Projective surface refinement for free-viewpoint video. *3rd European Conference on Visual Media Production (CVMP 2006). Part of the 2nd Multimedia Conference 2006*, pages 153–162, 2006.
- [Mur01] Walter Murch. *In The Blink Of An Eye*. Silman-James Press, 2001.
- [Nai91] Michael Naimark. VBK - A Moviemap of Karlsruhe. Interactive Instillation, 1991.
- [Nai94] Michael Naimark. SEE BANFF! Interactive Installation, 1994.
- [Nai96] Michael Naimark. Paris VideoPlan. Interactive Installation, 1996.
- [NIS12] NIST. TREC Video Retrieval Evaluation, 2012.
- [NW99] Jorge Nocedal and Stephen J. Wright. Numerical Optimization (Springer Series in Operations Research). *Analysis*, 1999.

- [NW05] Neil J. McCurdy and William G. Griswold. A systems architecture for ubiquitous video. In *Proc. International Conference on Mobile Systems, Applications, and Services*, pages 1–14, 2005.
- [OAS<sup>+</sup>09] Paul Over, George Awad, Alan F. Smeaton, Colum Foley, and James Lanagan. Creating a Web-scale Video Collection for Research. *Proceedings of the 1st Workshop on Web-scale Multimedia Corpus - WSMC '09*, page 25, 2009.
- [OCDD01] Byong Mok Oh, Max Chen, Julie Dorsey, and Frédo Durand. Image-based Modeling and Photo Editing. In *SIGGRAPH '01: Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, pages 433–442, New York, NY, USA, 2001. ACM Press.
- [OT06] Aude Oliva and Antonio Torralba. Building the Gist of a Scene: The Role of Global Image Features in Recognition. *Progress in Brain Research*, 155:23–36, January 2006.
- [PGB03] Patrick Pérez, Michel Gangnet, and Andrew Blake. Poisson image editing. *ACM SIGGRAPH 2003 Papers on - SIGGRAPH '03*, page 313, 2003.
- [Phy] NIST Physics Laboratory. Common View GPS Time Transfer.
- [PKVP09] Yael Prit, Eitam Kav-Venaki, and Shmuel Peleg. Shift-map Image Editing. *2009 IEEE 12th International Conference on Computer Vision*, (c):151–158, September 2009.
- [Pre01] Presto Studios. *Myst III: Exile*, 2001.
- [PSZ11] James Philbin, Josef Sivic, and Andrew Zisserman. Geometric Latent Dirichlet Allocation on a Matching Graph for Large-scale Image Datasets. *IJCV*, 95(2):138–153, 2011.
- [PTMD07] Pieter Peers, Naoki Tamura, Wojciech Matusik, and Paul Debevec. Post-production Facial Performance Relighting using Reflectance Transfer. *ACM Trans. Graph.*, 26(3):52, 2007.
- [PWC08] Suporn Pongnumkul, Jue Wang, and Michael Cohen. Creating Map-based Storyboards for Browsing Tour Videos. *Proceedings of the 21st Annual ACM Symposium on User Interface Software and Technology - UIST '08*, page 13, 2008.
- [Red06] Red Bee Media Ltd. *Piero*, 2006.
- [SB11] Klaus Schoeffmann and Laszlo Boeszoermenyi. Image and Video Browsing with a Cylindrical 3D Storyboard. *Proceedings of the 1st ACM International Conference on Multimedia Retrieval - ICMR '11*, pages 1–2, 2011.
- [Sch95] Gernot Schaufler. Dynamically Generated Impostors. In *GI Workshop on Modeling Virtual Worlds*, pages 129–135, 1995.

- [SFP10] Olivier Saurer, Friedrich Fraundorfer, and Marc Pollefeys. OmniTour: Semi-automatic Generation of Interactive Virtual Tours from Omnidirectional Video. In *Proc. Int. Symp. on 3D Data Processing, Visualization and Transmission (3DPVT2010)*, pages 1–8, 2010.
- [SGdA<sup>+</sup>10] Carsten Stoll, Juergen Gall, Edilson de Aguiar, Sebastian Thrun, and Christian Theobalt. Video-based Reconstruction of Animatable Human Characters. *ACM Transactions on Graphics*, 29(6):1, December 2010.
- [SGSS08] Noah Snavely, Rahul Garg, Steven M. Seitz, and Richard Szeliski. Finding Paths Through the World’s Photos. *ACM Transactions on Graphics*, 27(3):1, August 2008.
- [SH07] Jonathan Starck and Adrian Hilton. Surface Capture for Performance-Based Animation. *Computer Graphics and Applications, IEEE*, 27(3):21–31, 2007.
- [Shu07] Heung-Yeung Shum. *Image-based Rendering*. Springer, 2007.
- [Sin11] Singular Software. PluralEyes, 2011.
- [SK00] Heung-Yeung Shum and Sing Bing Kang. A Review of Image-based Rendering Techniques. *Proceedings of IEEE/SPIE Visual Communications and Image Processing*, pages 2–13, 2000.
- [SMV09] Pavel Serdyukov, Vanessa Murdock, and Roelof Van Zwol. Placing Flickr Photos on a Map. In *Proc. SIGIR*, pages 484–491, 2009.
- [SMW06] Scott Schaefer, Travis McPhail, and Joe Warren. Image Deformation using Moving Least Squares. *ACM Transactions on Graphics*, 25(3):533, July 2006.
- [Spo98] Sportsvision Inc. SportsVision, 1998.
- [SRB10] Deqing Sun, Stefan Roth, and Michael J. Black. Secrets of Optical Flow Estimation and their Principles. In *IEEE Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2010.
- [SRS11] SRSLY Inc. Switchcam, 2011.
- [SS11] Alex Simons and Steven Sinofsky. Improvements in Windows Explorer, 2011.
- [SSH02] Gregory G. Slabaugh, Ronald W. Schafer, and Mat C. Hans. Image-based Photo Hulls. *Proceedings. 1st International Symposium on 3D Data Processing Visualization and Transmission*, pages 704–862, 2002.
- [SSS06] Noah Snavely, Steven M. Seitz, and Richard Szeliski. Photo Tourism: Exploring Photo Collections in 3D. In *ACM Transactions on Graphics (TOG)*, volume 25, pages 835–846. ACM, 2006.
- [SSS08] Noah Snavely, Steven M. Seitz, and Richard Szeliski. Skeletal Graphs for Efficient Structure from Motion. *IEEE Conference on Computer Vision and Pattern Recognition*, pages(Section VII):1–8, 2008.

- [SSSE00] Arno Schödl, Richard Szeliski, David H. Salesin, and Irfan Essa. Video Textures. *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques - SIGGRAPH '00*, pages 489–498, 2000.
- [ST94] Jianbo Shi and Carlo Tomasi. Good Features to Track. In *Computer Vision and Pattern Recognition*, pages 593–600. IEEE, 1994.
- [Sta03] JR Starck. *Human modelling from multiple views*. PhD thesis, 2003.
- [Ste51] George Stevens. A Place In The Sun, 1951.
- [STMR09] Jennifer G. Sheridan, James Tompkin, Abel Maciel, and George Roussos. DIY Design Process for Interactive Surfaces. *Proceedings of 23rd Conference on Human Computer Interaction*, 2009.
- [SWS<sup>+</sup>00] Arnold W. M. Smeulders, Marcel Worring, Santini Santini, Amarnath Gupta, and Ramesh Jain. Content-based Image Retrieval at the End of the Early Years. *IEEE TPAMI*, 22(12):1349–1380, 2000.
- [SZ03] Josef Sivic and Andrew Zisserman. Video Google: A Text Retrieval Approach to Object Matching in Videos. *Proceedings Ninth IEEE International Conference on Computer Vision*, (ICCV):1470–1477 vol.2, 2003.
- [TAL<sup>+</sup>07] Christian Theobalt, Naveed Ahmed, Hendrik Lensch, Marcus Magnor, and Hans-Peter Seidel. Seeing People in Different Light — Joint Shape, Motion, and Reflectance Capture. *IEEE Transactions on Visualization and Computer Graphics*, 13(4):663–74, 2007.
- [TBP11] Aparna Taneja, Luca Ballan, and Marc Pollefeys. Modeling Dynamic Scenes Recorded with Freely Moving Cameras. *Computer Vision — ACCV 2010*, pages 613–626, 2011.
- [Tha12] Gunnar Thalin. *Deshaker*, 2012.
- [The10] The Register. Google acquires 'video Street View' startup, 2010.
- [The12] The Register. App of the Week iOS: Vyclone, 2012.
- [Tho17] D'arcy Wentworth Thompson. *On Growth And Form*. Cambridge University Press, 1917.
- [Tho06] Thorsten Thormählen. *Zuverlässige Schätzung der Kamerabewegung aus einer Bildfolge*. PhD thesis, Universität Hannover, 2006.
- [Tho07] Graham A. Thomas. White Paper 146: Real-time Camera Pose Estimation for Augmenting Sports Scenes. Technical Report January, BBC Research & Development, 2007.
- [Tho12] Graham Thomas. White Paper 220: Sports TV Applications of Computer Vision. Technical Report February, BBC Research & Development, 2012.

- [TI05] Robby T. Tan and Katsushi Ikeuchi. Separating Reflection Components of Textured Surfaces using a Single Image. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (ICCV):178–193, 2005.
- [TK91] Carlo Tomasi and Takeo Kanade. Detection and Tracking of Point Features. *Carnegie-Mellon University Technical Report CMU-CS-91-132*, (April), 1991.
- [TKKT12] James Tompkin, Kwang In Kim, Jan Kautz, and Christian Theobalt. Videoscapes: Exploring Sparse, Unstructured Video Collections. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 31(4), 2012.
- [TLRA03] Kentaro Toyama, Ron Logan, Asta Roseway, and Padmanabhan Anandan. Geographic Location Tags on Digital Images. In *ACM Transactions on Multimedia Computing, Communications, and Applications*, pages 156–166, 2003.
- [TMHF00] Bill Triggs, Philip F. McLauchlan, Richard I. Hartley, and Andrew W. Fitzgibbon. Bundle Adjustment — A Modern Synthesis. *Vision Algorithms: Theory and Practice*, pages 153–177, 2000.
- [TMJ<sup>+</sup>12] James Tompkin, Samuel Muff, Stanislav Jakushevskij, Jim Mccann, Jan Kautz, Marc Alexa, and Wojciech Matusik. Interactive Light Field Painting. In *SIGGRAPH 2012 Emerging Technologies*, 2012.
- [Tor58] Warren S. Torgerson. *Theory and Methods of Scaling*. Wiley, New York, NY, USA, 1958.
- [TPSK11] James Tompkin, Fabrizio Pece, Kartic Subr, and Jan Kautz. Towards Moment Imagery: Automatic Cinemagraphs. *Visual Media Production (CVMP), 2011 Conference on*, 2011.
- [UCK<sup>+</sup>03] Matthew Uyttendaele, Antonio Criminisi, Sing Bing Kang, Simon Winder, Richard Hartley, and Richard Szeliski. High-quality Image-based Interactive Exploration of Real-World Environments. *Microsoft Research MSR-TR-2003-61*, 2003.
- [UCK<sup>+</sup>04] M. Uyttendaele, A. Criminisi, S.~B. Kang, S. Winder, R. Szeliski, and R. Hartley. Image-based interactive exploration of real-world environments. *IEEE Computer Graphics and Applications*, 24(3):52–63, 2004.
- [VCL<sup>+</sup>11] Peter Vangorp, Gaurav Chaurasia, Pierre-Yves Laffont, Roland W. Fleming, and George Drettakis. Perception of Visual Artifacts in Image-based Rendering of Façades. In *Eurographics Symposium on Rendering*, volume 30, 2011.
- [Vid06] VideoSurf Inc. VideoSurf, 2006.
- [Vim12] Vimeo LLC. Vimeo, 2012.
- [VMK<sup>+</sup>10] Eduardo Veas, Alessandro Mulloni, Ernst Kruijff, Holger Regenbrecht, and Dieter Schmalstieg. Techniques for View Transition in Multi-camera Outdoor Environments. In *Proc. Graphics Interface*, pages 193–200, 2010.

- [Von07] Ulrike Von Luxburg. A Tutorial on Spectral Clustering. *Statistics and Computing*, 17(4):395–416, 2007.
- [Vyc12] Vyclone. Vyclone, 2012.
- [Wan95] Brian A. Wandell. *Foundations of Vision*. Sinauer Associates Inc, first edition, 1995.
- [Wik10] Wikipedia - Lap Dissolve, 2010.
- [Wik11] Wikipedia user: Grm\_wnr. Wikipedia - 180 degree rule, 2011.
- [WL11] Tobias Weyand and Bastian Leibe. Discovering Favorite Views of Popular Places with Iconoid Shift. In *Proc. ICCV*, 2011.
- [WTA11] Jason Wither, Yun-Ta Tsai, and Ronald Azuma. Indirect Augmented Reality. *Computers & Graphics*, 35(4):810–822, August 2011.
- [Wu07] Changchang Wu. SiftGPU: A GPU Implementation of Scale Invariant Feature Transform (SIFT), 2007.
- [WW99] A Wachowski and L Wachowski. *The Matrix*, 1999.
- [WWC<sup>+</sup>05] Michael Waschbüsch, Stephan Würmlin, Daniel Cotting, Filip Sadlo, and Markus Gross. Scalable 3D Video of Dynamic Scenes. *The Visual Computer*, 21(8-10):629–638, August 2005.
- [XLS<sup>+</sup>11] Feng Xu, Yebin Liu, Carsten Stoll, James Tompkin, Gaurav Bharaj, Qionghai Dai, Hans-Peter Seidel, Jan Kautz, and Christian Theobalt. Video-based Characters - Creating New Human Performances from a Multi-view Video Database. In *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 2011.
- [XT97] Yalin Xiong and Ken Turkowski. Creating Image-based VR using a Self-calibrating Fish-eye Lens. In *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*, pages 237–243, 1997.
- [YDTS10] Beste F. Yuksel, Michael Donnerer, James Tompkin, and Anthony Steed. A Novel Brain-computer Interface using a Multi-touch Surface. *Proceedings of the 28th International Conference on Human Factors in Computing Systems - CHI '10*, page 855, 2010.
- [YDTS11] Beste F. Yuksel, Michael Donnerer, James Tompkin, and Anthony Steed. Novel P300 BCI Interfaces to Directly Select Physical and Virtual Objects. In *5th International Brain-Computer Interface Conference*, pages 5–8, 2011.
- [You12] YouTube. F.A.Q., 2012.
- [Zha94] Zhengyou Zhang. Iterative Point Matching for Registration of Free-form Curves and Surfaces. *International Journal of Computer Vision*, 13(2):119–152, October 1994.

- [Zha04] Cheng Zhang. *Practice and Theory of Extracting 3D Information from Fish-eye Pictures*. PhD thesis, Shandong University, 2004.
- [ZKP10] Christopher Zach, Manfred Klopschitz, and Marc Pollefeys. Disambiguating Visual Relations using Loop Constraints. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1426–1433. IEEE, 2010.
- [ZKU<sup>+</sup>04] C Lawrence Zitnick, Sing Bing Kang, Matthew Uyttendaele, Simon Winder, and Richard Szeliski. High-quality Video View Interpolation using a Layered Representation. *ACM Trans. Graph.*, 23(3):600–608, 2004.
- [ZLC<sup>+</sup>10] Bo Zhang, Qinlin Li, Hongyang Chao, Bill Chen, Eyal Ofek, and Ying-Qing Xu. Annotating and Navigating Tourist Videos. *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems - GIS '10*, (c):260, 2010.
- [ZPB07] Christopher Zach, Thomas Pock, and Horst Bischof. A Globally Optimal Algorithm for Robust TV-L1 Range Image Integration. *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8, 2007.
- [ZS10] Amir Roshan Zamir and Mubarak Shah. Accurate Image Localization based on Google Maps Street View. *Computer Vision ECCV 2010*, pages 255–268, 2010.
- [ZZS<sup>+</sup>09] Yantao Zheng, Ming Zhao, Yang Song, Hartwig Adam, Ulrich Buddemeier, Alessandro Bissacco, Fernando Brucher, Tat-Seng Chua, Harmut Neven, and Jay Yagnik. Tour the World: Building a Web-scale Landmark Recognition Engine. In *Proc. CVPR*, pages 1085–1092, 2009.