Supplemental Material for Preference and Artifact Analysis for Video Transitions of Places

James Tompkin[1,4], Min H. Kim[2], Kwang In Kim[1], Jan Kautz[3], Christian Theobalt[1]
MPI für Informatik[1], KAIST[2], University College London[3], Intel VCI[4]

This supplemental material is exhaustive and is intended to be used as a reference if questions arise when reading the main paper. It contains:

- Per-transition analysis of preference and artifacts (Section 1).
- Per-set analysis of preference and artifacts (Section 2).
- Explainations of transition types used in our perceptual experiment, containing a historical review of its application, our technical method to achieve the transition, and a description of artifacts that may appear in the transition (Sections 3-8).
- Explainations of techniques and issues in common between transition types (Section 9).
- Images of the website used for the transition ranking experiment (Figures 29 and 30).

1 Per-transition analysis across sets

1.1 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 preferred 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 2b), and the absolute difference in preference between the nearest transitions is greater in the considerable case (Figure 1, 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.

1.2 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 5, 7 and 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.

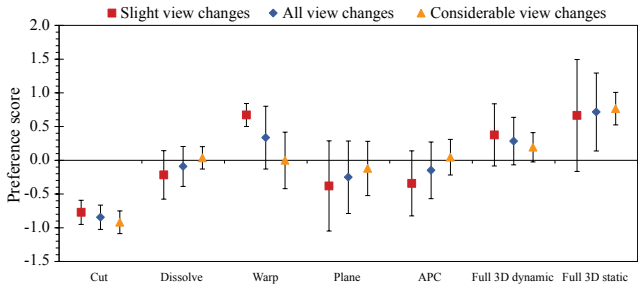

Figure 1: *Reproduced from main paper for reference. The result of preference scores for different transition types across all scenes. The scale is in the form of z-score, of which group mean is at 0, the y-value represents multiples of the group standard deviation as discriminal dispersion of the perceived image quality, and higher scores indicates more preference by participants.*

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

(a) *All scenes*

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

(b) *Slight view change*

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

(c) *Considerable view change*

Table 2: *Reproduced from the main paper. Pairwise significance tests with the p-values of the preference scores (α = 0.05 each). 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.*

1.3 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 1). 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 7, main paper). 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.

1.4 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.

1.5 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 7).

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 22). This case is actually a failure for APC: the generated result is incorrect. 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.

1.6 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 transition start and end anchor frames (at which the full 3D static transition starts and ends). The anchor 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.

1.7 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 [Pérez et al. 2003].

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.

2 Per-set analysis across transitions

We present perceptual scales for each set, and specific features/artefacts are cross-referenced with comments to investigate the properties of the scene which have affected the result.

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.

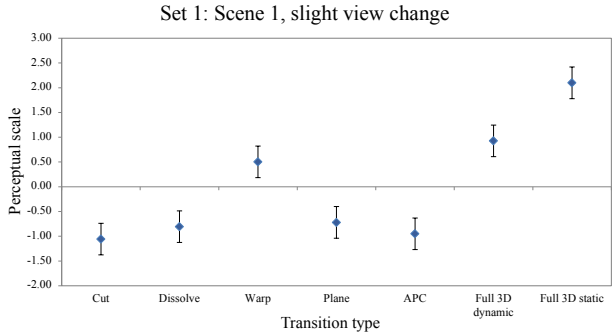

Figure 2: *Mean and 95% confidence error bars plotted on a perceptual scale for the different transition types for set 1. Perceptual scales are in z-score order, with the group mean at 0 and y-value representing multiples of the group standard deviation.*

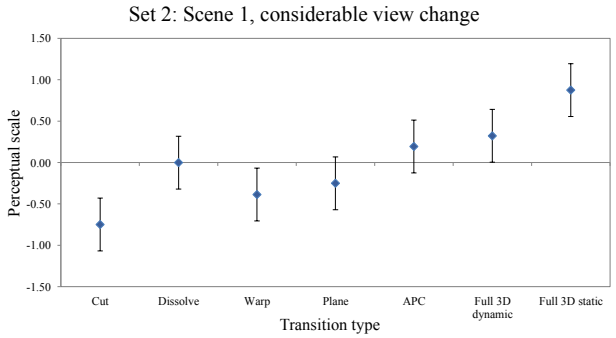

Figure 3: *Mean and 95% confidence error bars plotted on a perceptual scale for the different transition types for set 2. Perceptual scales are in z-score order, with the group mean at 0 and y-value representing multiples of the group standard deviation.*

Set 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 2 shows the perceptual scores.

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 22) — 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 3 shows the perceptual scores.

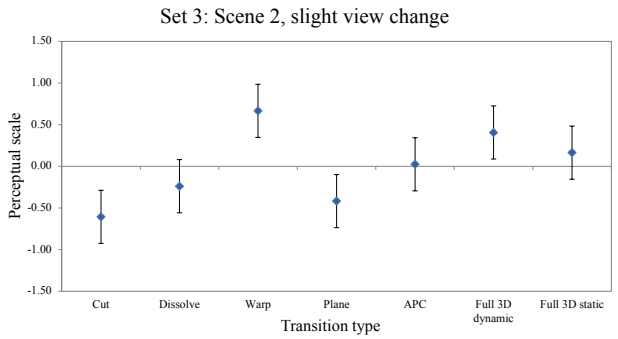

Figure 4: *Mean and 95% confidence error bars plotted on a perceptual scale for the different transition types for set 3. Perceptual scales are in z-score order, with the group mean at 0 and y-value representing multiples of the group standard deviation.*

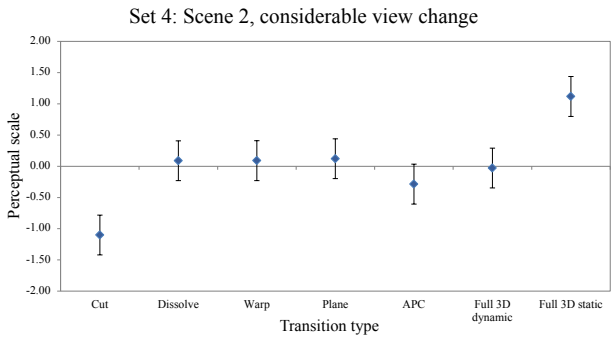

Figure 5: *Mean and 95% confidence error bars plotted on a perceptual scale for the different transition types for set 4. Perceptual scales are in z-score order, with the group mean at 0 and y-value representing multiples of the group standard deviation.*

Set 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 transition start and end anchor 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 4 shows the perceptual scores.

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 5 shows the perceptual scores.

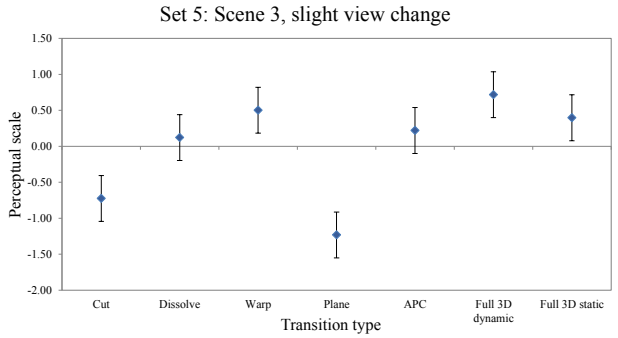

Figure 6: *Mean and 95% confidence error bars plotted on a perceptual scale for the different transition types for set 5. Perceptual scales are in z-score order, with the group mean at 0 and y-value representing multiples of the group standard deviation.*

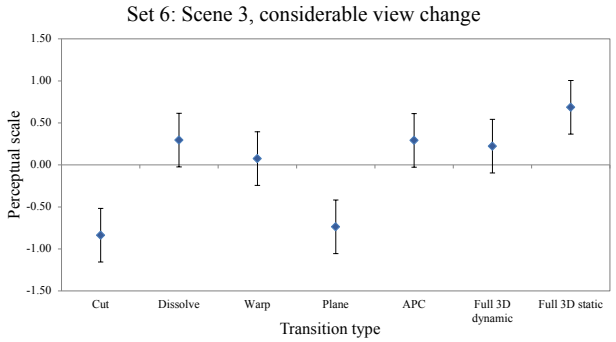

Figure 7: *Mean and 95% confidence error bars plotted on a perceptual scale for the different transition types for set 6. Perceptual scales are in z-score order, with the group mean at 0 and y-value representing multiples of the group standard deviation.*

Set 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 6 shows the perceptual scores.

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 7 shows the perceptual scores.

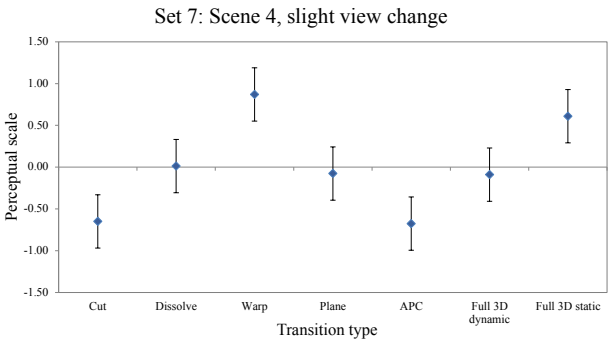

Figure 8: *Mean and 95% confidence error bars plotted on a perceptual scale for the different transition types for set 7. Perceptual scales are in z-score order, with the group mean at 0 and y-value representing multiples of the group standard deviation.*

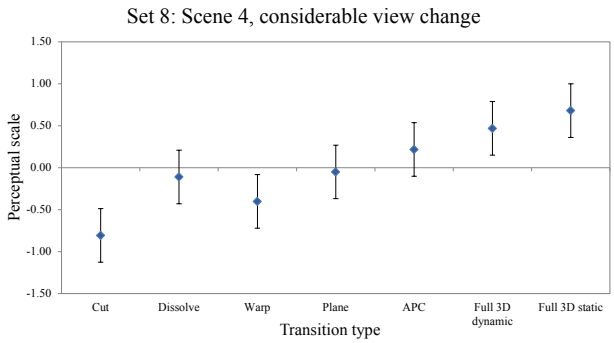

Figure 9: *Mean and 95% confidence error bars plotted on a perceptual scale for the different transition types for set 8. Perceptual scales are in z-score order, with the group mean at 0 and y-value representing multiples of the group standard deviation.*

Set 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 8 shows the perceptual scores.

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 9 shows the perceptual scores.

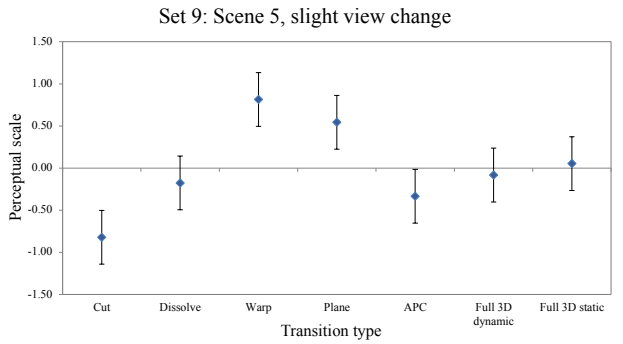

Figure 10: *Mean and 95% confidence error bars plotted on a perceptual scale for the different transition types for set 9. Perceptual scales are in z-score order, with the group mean at 0 and y-value representing multiples of the group standard deviation.*

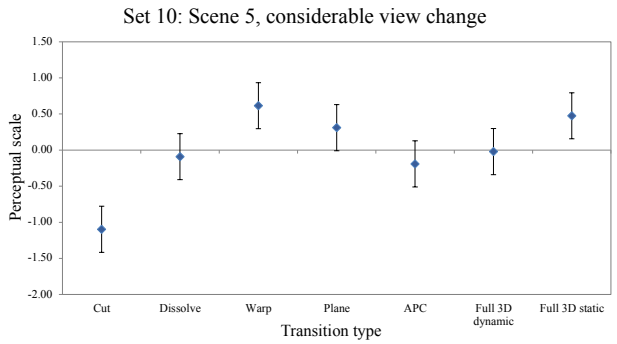

Figure 11: *Mean and 95% confidence error bars plotted on a perceptual scale for the different transition types for set 10. Perceptual scales are in z-score order, with the group mean at 0 and y-value representing multiples of the group standard deviation.*

Set 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 10 shows the perceptual scores.

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 11 shows the perceptual scores.

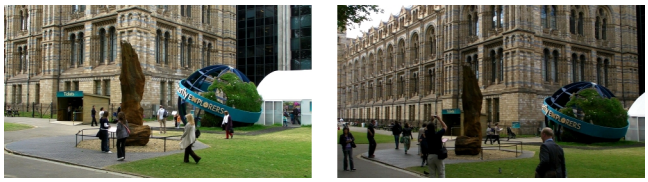

Figure 12: *An example of two frames which constitute a jump cut. We can see a slight camera position and rotation change and a temporal change.*

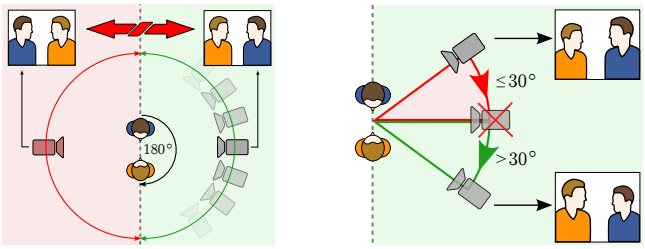

Figure 13: a) *The* 180° *rule. Cameras should not cross the dotted line when cutting between objects.* b) *The* 30° *rule. Cameras should be at least* 30° *apart relative to the scene when cutting. a) courtesy of [Wikipedia user: Grm_wnr 2011]; b) created from a).*

Figure 14: *Joining these two video frames would break the* 180° *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.*

3 Cut

3.1 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 12 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 [Dmytryk 1984; Murch 2001]. 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 to consider (McCurdy provides a summary [McCurdy 2007, 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 [Friedman 2003, 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 13 visualizes both the 30° and 180° rules.

3.2 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 decoding in tandem, the cut is simply a swap of the source of the output buffer from one clip to the next.

We 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 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.

3.3 Artifacts

Given that the aim of a seamless transition is to link sequences while providing a sense of orientation, we could say that the major artifact 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, in vision-based correspondence finding, symmetric buildings may break the 180° rule. If there is not sufficient context between frames to disambiguate the sides of symmetric buildings, the position of the camera may move between half-spaces (Figure 14). 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.

The 30° rule is commonly broken between videos of the same content within video collections. As such, many typical cut transitions between frames will be jump or axial cuts (see Figure 12). It would be possible to reject candidates with view change angles of less than 30° should the creator wish to use cut transitions. However, this is not explored as we focus on situations where a seamless transition provides orientation continuity.

Figure 15: *A frame in the middle of a dissolve transition (bottom) between two frames (top left and right).*

4 Dissolve

4.1 Background

A dissolve merges between two shots gradually over time (Figure 15). 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 [Dmytryk 1984, 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) [Wik 2010]. 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 [Linklater 2006, last transition in movie] or for when juxtaposition is undesirable [Stevens 1951]); 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.

4.2 Technical Method

We 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 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 $k = (1, 2, ..., t)$, new transition frame I_k^T equals:

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

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 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, we do 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.

4.3 Artifacts

Dissolve transitions, like cuts, are comfortable to watch. However, when interpreted literally they contain many artifacts: 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 artifacts can be planned away, but in arbitrary video collections these dynamic objects are difficult and costly to handle [Morvan and O'Sullivan 2009].

5 Warp

5.1 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 anarmorphosis to include warped elements in paintings, and mathematical biologist D'Arcy Thompson used image warping to describe natural variation in creatures [Thompson 1917]. 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 [Howard 1988] and music videos such as Michael Jackson's Black and White [Beier and Neely 1992].

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 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 [Lowe 2004; Sun et al. 2010; Lipski et al. 2010b] and perform interpolation [Igarashi et al. 2005; Schaefer et al. 2006; Lipski et al. 2010a]. For instance, the SIFT flow work of Liu et al. [Liu et al. 2008] 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. As we have many hundreds or thousands of transition possibilities within a video collection, any warping method must run in a short amount of time so that either preprocessing or real-time display is feasible.

5.2 Technical Method

We start the warp transition implementation explanation by first describing how to warp between the two frames only. We automatically discover image correspondences by extracting SIFT feature points and matching their descriptors [Lowe 2004]. From these correspondences, we robustly estimate a fundamental matrix with the normalized eight-point algorithm [Hartley and Zisserman 2004] and RANSAC [Fischler and Bolles 1981]. 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 frames with an interpolation scheme — we choose moving least squares (MLS) image warping [Schaefer et al. 2006]. 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 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 frame to the other. We compute this field bi-directionally. To generate a transition sequence which moves from source to target, we synthesize frames by blending fractional interpolations of the source-to-target and target-to-source warps. Formally, for start and end 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 $k = (1, 2, ..., t)$, and 2D vector fields $V_{source \rightarrow target}$, new transition frame I_k^T equals:

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

$$+\,\alpha\,\mathrm{invmap}(I_j^T, (1-\alpha)\, V_{I_j^T \rightarrow I_i^S}), \quad (3)$$

where $\alpha = \frac{k}{t+1}$ and $\mathrm{invmap}(I, V)$ applies an inverse (or backward/reverse) mapping. This allows us to dissolve between registered frames, and to synthesize new frames which move the view from one frame to the other (Figure 16).

Figure 16: *A frame in the middle of a warp transition (middle) between two frames (top left and right). This is the first stage of the warp transition effect, where the two videos are approximately aligned.*

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 frames, but instead wish to warp with pixels from frames surrounding the transition start and end anchor frames. To do this, we must find vector fields for each frame of each video clip to the corresponding anchor frame. We first find KLT feature points [Tomasi and Kanade 1991; Shi and Tomasi 1994] over small windows in time around each frame, and robustly reject outliers on dynamic objects as before [Fischler and Bolles 1981; Hartley and Zisserman 2004]. 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 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 anchor 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 anchor frames and leading up to the target anchor frame.

Formally, for notation as in Equation 3, new transition frame I_k^T equals:

$$I_k^T = (1-\alpha)\,\mathrm{invmap}(I_{ik}^S, \alpha\,\mathrm{lutbi}(V_{I_i^S \rightarrow I_{ik}^S}, V_{I_i^S \rightarrow I_j^T})) \\ +\alpha\,\mathrm{invmap}(I_{jk}^T, (1-\alpha)\,\mathrm{lutbi}(V_{I_j^T \rightarrow I_{jk}^T}, V_{I_j^T \rightarrow I_i^S})) \quad (4)$$

where ik and jk are as in Equation 1 and $\mathrm{lutbi}(V_1, V_2)$ 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 quadrat-

Figure 17: *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 5.3 contains artifact discussion and example images.*

ically 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. [Eisemann et al. 2008]. 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 17). 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 4, and W_k^S denoting a warped frame from the start clip for transition frame k, new transition frame I_k^T equals:

$$\begin{aligned} W_k^S &= \text{invmap}(I_{ik}^S, \alpha\, \text{lutbi}(V_{I_i^S \to I_{ik}^S}, V_{I_i^S \to I_j^T})), \\ W_k^E &= \text{invmap}(I_{jk}^T, (1-\alpha)\, \text{lutbi}(V_{I_j^T \to I_{jk}^T}, V_{I_j^T \to I_i^S})), \\ I_k^T &= (1-\alpha)\, \text{invmap}(W_k^S, \alpha\, V_{W_k^S \to W_k^E}) \\ &\quad + \alpha\, \text{invmap}(W_k^E, (1-\alpha) V_{W_k^E \to W_k^S}), \end{aligned} \tag{5}$$

where the vector fields between W_k^S and W_k^E are formed by optical flow [Brox et al. 2004].

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.

Figure 18: *Artifacts in the warp transitions, taken as zoomed regions of Figure 17.* Left*: Swirl artifacts 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.*

5.3 Artifacts

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.

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 artifacts produced by MLS warping range from small swimming to large swirls in the worst case (Figure 18). Generally, any correspondence mismatch causes a significant visibly distracting artifact 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 18). 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 [Humayun et al. 2011]. We reduce the flicker by temporally smoothing the flow field with a 5-wide Gaussian kernel.

Unfortunately, these flicker artifacts often appear within the contrast sensitivity temporal frequency peak range of 5 to 10Hz [Wandell 1995, Figure 7.23-A, p. 223], and so are easy to notice. However, the 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. [Lipski et al. 2010b] 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 [Barnes et al. 2009; Barnes et al. 2010], though this has large caveats which will cause other artifacts such as no temporal

consistency and incorrectly inpainted structure causing other flow artifacts.

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 α to a curve which matched the accelerations.

6 Plane

6.1 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 [McCurdy et al. 2006; Snavely et al. 2006]. 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 [Microsoft 2008; Google 2008].

Early research into image-based graphics used cylinders or spheres to reproject omnidirectional imagery [Chen 1995; McMillan and Bishop 1995], 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 [Horry et al. 1997; Kang and Shin 2002]. 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 [Schaufler 1995]. Recent work has automated the billboard modelling of a single dynamic object at the centre of converging viewpoints for video-based viewpoint interpolation [Ballan et al. 2010], or has asked the user to draw silhouettes to define depth boundaries within a recovered point cloud [Chaurasia et al. 2011].

6.2 Technical Method

We choose to use the approach of Snavely et al. [Snavely et al. 2006, 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. [Snavely et al. 2006] first find SIFT feature points and candidate correspondences between images [Lowe 2004], then robustly find a fundamental matrix using RANSAC [Fischler and Bolles 1981] and the eight point algorithm [Hartley and Zisserman 2004]. 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 [Nocedal and Wright 1999]. 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 [Lourakis and Argyros 2004]. 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.

Figure 19: 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.*

Next, they estimate a common plane between the two views [Snavely et al. 2006, 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 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 9.8, 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. [Snavely et al. 2006] 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. [Snavely et al. 2006] 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 1, and interpolating a novel camera between the two views from which to render (Section 9.9). Figure 19 shows an example frame from slight and considerable view plane transitions.

6.3 Artifacts

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 artifacts such as static object ghosting as well as dynamic object ghosting.

Figure 20: Left*: Ghosting on static objects such as Big Ben.* Right*: Significant skewing in a considerable view change transition.*

These manifest as parallax or shear artifacts causing buildings to noticeably lean across wider view change transitions (Figure 20). This type of artifact is typical of plane transitions where the geometry proxy does not well represent the scene.

However, the plane transition is commonly used because its artifacts are less objectionable than those in other transitions. For instance, Snavely et al. [Snavely et al. 2006] compare it to triangulated morphs and find it preferable due to robustness. Vangorp et al. [Vangorp et al. 2011] study parallax effects caused by inaccurate geometry (among other IBR-related artifacts) 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 artifacts will not be as objectionable in our slight view change case.

7 Ambient Point Clouds

7.1 Background

Ambient Point Clouds (APC) is a recent technique developed by Goesele et al. [Goesele et al. 2010] to provide motion cues during transitions. It builds upon earlier work in producing geometric models through multi-view stereo from community image databases [Goesele et al. 2007]. 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.

7.2 Technical Method

The method is simple enough to summarize here, but for technical details, please refer to the original paper [Goesele et al. 2010]. APC starts by computing the minimum and maximum depths of any recovered geometry 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.

Figure 21: 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.*

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 6) 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 artifacts. Any video-based APC needs to generate temporally consistent results that do not introduce further artifacts.

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 9.8) instead of static images. Figure 21 shows an example transition.

7.3 Artifacts

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, artifacts caused by incorrect geometry or inaccurate video registration are not specifically APC artifacts. However, APC and our video repurposing of APC do have their own set of artifacts.

Goesele et al. [Goesele et al. 2010, 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 9.9). 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 28). This is one price we pay for not implementing 'APC for video'; however, in practice, we do not often encountered these streak intersection artifacts: 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 artifact from not implementing 'APC for video' is areas of virtual transition with no content (see Section 9.10). 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 22 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 22. This has occurred because both start and end clips were undergoing rotations in opposite directions: the anchor 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 artifact 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

Figure 22: 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.*

APC is less sampled in depth due to view dependence, and so many more of these holes are visible (see Figure 22, *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.

8 Full 3D

8.1 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 9.1), 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 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.

8.2 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 anchor 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

Figure 23: 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 24: 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.*

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.

We separate our full 3D transitions into two types: *dynamic*, where the registered video projection (Section 9.8) 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 23). 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 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 24). 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 artifacts in the dynamic full 3D transition.

8.3 Artifacts

Full 3D transitions suffer all artifacts 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 artifacts. Full 3D transitions suffer the same artifacts 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 artifacts while the dynamic objects continue to move, whereas static full 3D transitions have these objects frozen in time as they dissolve into one another.

Full 3D transitions also suffer the same artifacts 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

Figure 25: Top*: Two frames from a full 3D dynamic transition which contain artifacts 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.*

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 artifacts 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 9.10). 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 artifacts may be seen in Figure 25.

9 Transition Issues in Common

9.1 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 cam-

era 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 artifacts. These points are removed in Section 9.4.

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 [Tompkin et al. 2012] 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 [Zach et al. 2010; Hauagge and Snavely 2012], 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 [Tompkin et al. 2012] 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 [Ballan et al. 2010; Taneja et al. 2011], 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.

9.2 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. [Snavely et al. 2006] 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).

9.3 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.

9.4 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 [Marroquim et al. 2007], but in our implemented system this part involves manually executing functions within MeshLab [Cignoni et al. 2011]. 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 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 [Kazhdan et al. 2006]. 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 26.

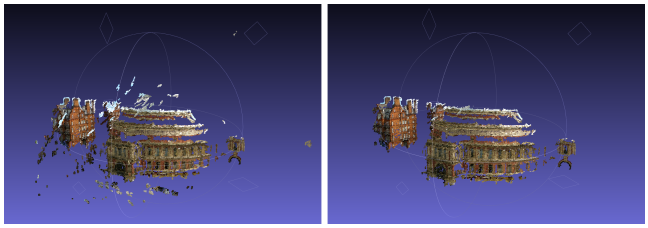

Figure 26: Left: *Input multi-view stereo point cloud containing 2027195 vertices.* Right: *Cleaned and resampled version with sky and low-density clusters removed containing 137521 vertices.*

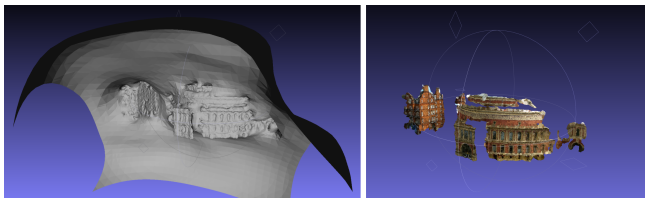

Figure 27: Left: *Poisson surface reconstruction of cleaned point cloud.* Right: *Coloured, clipped and filled version with 88881 vertices and 175354 faces.*

9.5 Mesh Post-processing

The cleaned point cloud is transformed into a mesh by Poisson surface reconstruction [Kazhdan et al. 2006], 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. [Furukawa et al. 2010]. 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 27). 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 27.

9.6 Pipeline

With the above issues outlined, the final pipeline for geometry reconstruction with optional stages included is described in Algorithm 1.

9.7 Computational Performance

Over a larger database of videos, the reconstruction and tracking for 200+ transitions 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 [Furukawa et al. 2010], the resulting geometry can sometimes be of poor quality. In these cases, as justified by our experiment in the main paper, we handle these problems by choosing a dissolve transition which does not require 3D geometry.

Algorithm 1: Pipeline for geometry reconstruction.

foreach *transition* **do**
  Compute camera poses by [Snavely et al. 2006];
    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 [Furukawa et al. 2010];
  foreach *cluster* **do**
    Compute multi-view stereo point cloud by [Furukawa and Ponce 2010];
  Merge clusters by union operator;
  Post-process point cloud by Section 9.4;
  Form point cloud into mesh by [Kazhdan et al. 2006];
  Post-process mesh by Section 9.5;
  Add planes for full 3D transition types only by Section 8.2;

More recent work by Crandall et al. [Crandall et al. 2011] speeds up large-scale structure-from-motion 5-8x, increasing the possible number of image matches within a video collection, 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.

9.8 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. [Snavely et al. 2006] 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 artifacts 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 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 anchor frame with a local tracking across video frames either side of the anchor frame. We first apply the Snavely et al. [Snavely et al. 2006] method to find good correspondences and 3D points between the anchor views in each video. Next, for each video we find KLT feature points [Tomasi and Kanade 1991; Shi and Tomasi 1994] over transition-frame-length windows in time around each anchor frame, and robustly reject outliers with RANSAC [Fischler and Bolles 1981] that suffer large reprojection errors after fitting a standard camera model with the eight-point algorithm [Hartley and Zisserman 2004]. Bundle adjustment is performed after every 10 frames to optimize the pose and 3D locations of the 2D KLT feature points [Triggs et al. 2000]. The KLT-based structure and

motion recovery is computed by the Voodoo tracker [Thormählen 2006].

We now have many separate pose results in many different coordinate systems: one coordinate system for the anchor 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 [Horn 1987; Eggert et al. 1997] 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 [Zhang 1994]. 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 anchor frames between clips. We find 2D KLT points from the local tracking which match 2D SIFT points from the global registration in the anchor 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 [Kirkpatrick et al. 1983], though other optimization methods are equally applicable [Nocedal and Wright 1999].

We perform this optimization scheme bi-directionally out from the transition start and end anchor 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 artifacts, the quality deteriorates considerably and videos are no longer accurately aligned with the 3D geometry, leading to ghosting artifacts in the 3D transitions (particularly in Scene 4).

For our databases, standard KLT tracking was sufficient, 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. [Ballan et al. 2010] have a similar but subtly different problem of registering videos to geometry. They begin with a similar standard pose estimation [Hartley and Zisserman 2004], but find 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 [Ballan et al. 2010, 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 (subsampled 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. [Ballan et al. 2010] 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.

9.9 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 9.8. 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 [Snavely et al. 2006; Goesele et al. 2010], 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

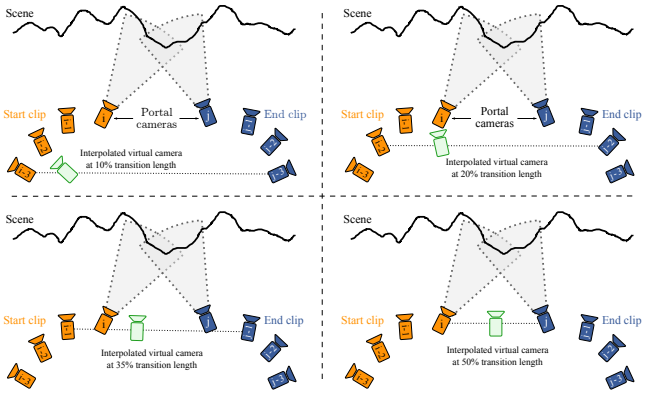

Figure 28: Clockwise from top left: *Progression of interpolated virtual camera (green) position and rotations. Frames i and j are the middle frames of a transition in all but the warp and full 3D static transitions. The transition progression beyond frames i and j ($+1$,$+2$,$+3$, and so on) are not shown.*

camera to blend between the two styles of camera motion, but existing work on style transfer [Kurz et al.] 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 transition start and end anchor frame (Section 9.8), we can directly interpolate the per-frame poses to generate the new pose for the virtual view (Figure 28). 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 anchor 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 anchor frames.

9.10 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 22). 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 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 [Barnes et al. 2009; Barnes et al. 2010; Prit et al. 2009; Kav-Venaki 2010], 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 [Granados et al. 2012b; Granados et al. 2012a]. 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 7.3, this is not always successful and itself introduces temporal artifacts 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.

9.11 Transition Timing Differences

Given a pair of matched anchor frames, we must decide where they appear within a transition. For instance, a dissolve transition could start with one anchor frame and end with the other, meaning that the frames during the transition contain footage from after the anchor frame in the start clip and before the anchor 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 anchor 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 anchor frame is followed immediately by the end clip anchor frame.

Dissolve
As the exemplifier; the dissolve sets the anchor frames to be in the middle of the transition.

Warp
The warp transition places the anchor frames at the start and end of the transition. A warp transition should have the anchor 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 anchor frames for our moving-least-squares warp to successfully transition between them. Were the anchor 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 anchor 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 anchor frames to the middle of the transition.

Ambient Point Clouds
Ambient point cloud transitions place the anchor frame in the middle of the transition. The location of the anchor 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 anchor frame begins the transition into virtual camera, and the end clip anchor 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. 27) and there are no dynamic scene objects present. Hence, we set the anchor frames to be in the middle of the transition.

10 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 reconstruction 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 videos in real time during interaction, but current software methods are too slow. Instead, we pre-compute 2D affine stabilization parameters as a per-frame crop region computed with a custom Deshaker build [Thalin 2012]), 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 artifacts or without suffering rolling shutter wobble [Liu et al. 2009; Liu et al. 2011; Grundmann et al. 2011]. Our approach allows the viewer control should these tricky stabilization cases arise.

11 Transition Feature/Artefact Table

We collate and categorize all feature and artifact types in each transition in Table 3. This is a repetition of Table 1 in the main paper.

12 Transition Experiment Material

This section includes material from the transition experiment outlined in the main paper. Figures 29 and 30 visualize the website interface used by participants to rank video transitions.

References

BALLAN, L., BROSTOW, G. J., PUWEIN, J., AND POLLEFEYS, M. 2010. Unstructured Video-based Rendering: Interactive Exploration of Casually Captured Videos. *ACM Transactions on Graphics 29*, 4 (July), 1.

BARNES, C., SHECHTMAN, E., FINKELSTEIN, A., AND GOLDMAN, D. B. 2009. PatchMatch: A Randomized Correspondence Algorithm for Structural Image Editing. In *ACM Transactions on Graphics (TOG)*, ACM, vol. 28, 24.

BARNES, C., SHECHTMAN, E., GOLDMAN, D. B., AND FINKELSTEIN, A. 2010. The Generalized Patchmatch Correspondence Algorithm. *European Conference on Computer Vision (ECCV)*, 29–43.

BEIER, T., AND NEELY, S. 1992. Feature-based Image Metamorphosis. *ACM SIGGRAPH Computer Graphics 26*, 2 (July), 35–42.

BROX, T., BRUHN, A., AND PAPENBERG, N. 2004. High Accuracy Optical Flow Estimation based on a Theory for Warping. *European Conference on Computer Vision (ECCV) 4*, May, 25–36.

CHAURASIA, G., SORKINE, O., DRETTAKIS, G., AND INRIA, R. 2011. Silhouette-aware Warping for Image-Based Rendering. In *Eurographics Symposium on Rendering*, vol. 30.

Video Ranking Experiment

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 generated 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:

Start

Clear All Data (Reset Rankings)

Figure 29: *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.*

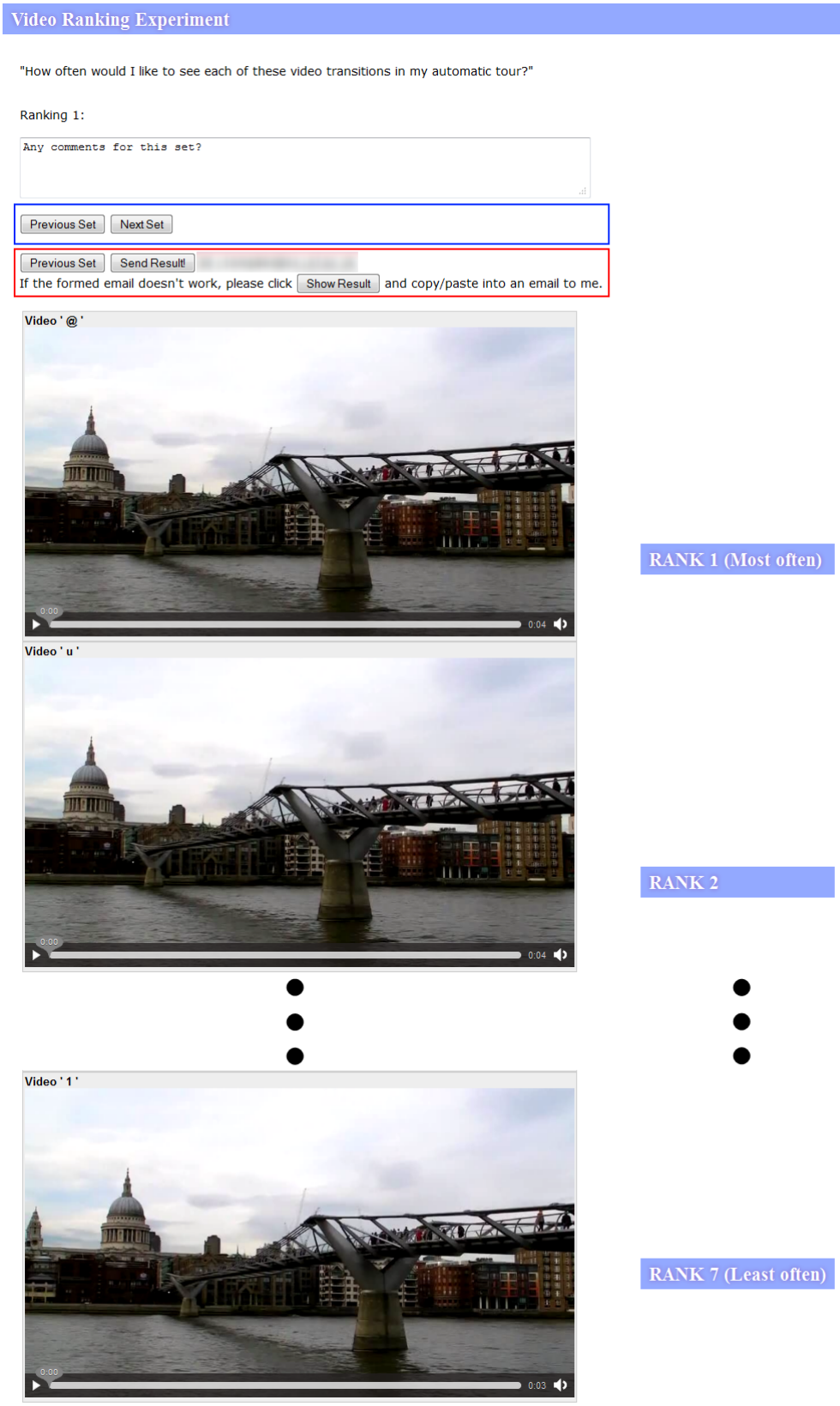

Figure 30: *Image of the webpage for ranking video transitions. Each transition type is randomly ordered onto the page. Participants can drag and drop the videos into order, using the ranking labels to the left to keep track. Comments can be left for each ranking, of which there are ten in total (five 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.*

	Cut	Dissolve	Warp	Plane	APC	Full3DDyn	Full3DSta
Feature							
Registered scene			●	●	●	●	●
3D effect			○[1]		●	●	●
Dynamic objects		●	●	●	●	●	
Smooth virtual camera (9.9)			○[2]	●	●	●	●
Common familiarity	●	●					
Signifies change of time		●					
Explicit motion cues					●		
Frozen time							●
Artefact							
Ghosting (static objects)		●		●[3]			
Ghosting (dynamic objects)		●	●	●	●	●	
Orientation loss (3.3)	●	●					
Bad corresp. swirls (5.3)			●				
Frame edge flickering (5.3)			●				
Skewed scene (6.3)				●		○[4]	○[4]
Temporal pepper noise (7.3)					●		
Multiple scene elements (8.3)					○[5]	●	○[6]
Recovered geom. failures (9.1)					●	●	●
Empty black regions (9.10)				●	○[7]	○[8]	○[8]

Table 3: *A table collating all features and artifacts for each transition type. Section numbers for text explaining each feature or artifact 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 anchor 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.*

CHEN, S. E. 1995. 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*, ACM Press, New York, NY, USA, 29–38.

CIGNONI, P., RANZUGLIA, G., CALLIERI, M., CORSINI, M., DELLEPIANE, M., GANOVELLI, F., PIETRONI, N., AND TARINI, M., 2011. MeshLab.

CRANDALL, D., OWENS, A., SNAVELY, N., AND HUTTENLOCHER, D. 2011. Discrete-continuous Optimization for Large-scale Structure from Motion. *Computer Vision and Pattern Recognition 286*, 26, 3001–3008.

DMYTRYK, E. 1984. *On Film Editing*.

EGGERT, D. W., LORUSSO, A., AND FISHER, R. B. 1997. Estimating 3-D Rigid Body Transformations: A Comparison of Four Major Algorithms. *Machine Vision and Applications 9*, 5-6 (Mar.), 272–290.

EISEMANN, M., DECKER, B. D., MAGNOR, M., BEKAERT, P., DE AGUIAR, E., AHMED, N., THEOBALT, C., AND SELLENT, A. 2008. Floating Textures. *Computer Graphics Forum 27*, 2 (Apr.), 409–418.

FISCHLER, M. A., AND BOLLES, R. C. 1981. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Communications of the ACM 24*, 6, 381–395.

FRIEDMAN, D. 2003. *Knowledge-based Formalization of Cinematic Expression and its Application to Animation*. PhD thesis.

FURUKAWA, Y., AND PONCE, J. 2010. Accurate, Dense, and Robust Multi-view Stereopsis. *IEEE TPAMI 32*, 8 (Aug.), 1362–1376.

FURUKAWA, Y., CURLESS, B., SEITZ, S. M., AND SZELISKI, R. 2010. Towards Internet-scale Multi-view Stereo. In *Proc. IEEE CVPR*, 1434–1441.

GOESELE, M., SNAVELY, N., CURLESS, B., HOPPE, H., AND SEITZ, S. M. 2007. Multi-view Stereo for Community Photo Collections. *2007 IEEE 11th International Conference on Computer Vision*, 1–8.

GOESELE, M., ACKERMANN, J., FUHRMANN, S., HAUBOLD, C., AND KLOWSKY, R. 2010. Ambient Point Clouds for View Interpolation. *ACM Transactions on Graphics (TOG) 29*, 4, 1–6.

GOOGLE, 2008. Panoramio Look Around.

GRANADOS, M., KIM, K. I., TOMPKIN, J., KAUTZ, J., AND THEOBALT, C. 2012. Background Inpainting for Videos with Dynamic Objects and a Free-moving Camera. In *European Conference on Computer Vision (ECCV)*.

GRANADOS, M., TOMPKIN, J., KIM, K. I., GRAU, O., KAUTZ, J., AND THEOBALT, C. 2012. How Not to Be Seen - Object Removal from Videos of Crowded Scenes. *Computer Graphics Forum 31*, 2pt1 (May), 219–228.

GRUNDMANN, M., KWATRA, V., AND ESSA, I. 2011. Auto-directed Video Stabilization with Robust L1 Optimal Camera Paths. *Computer Vision and Pattern Recognition*, 1 (June), 225–232.

HARTLEY, R., AND ZISSERMAN, A. 2004. *Multiple View Geometry in Computer Vision*, second ed. Cambridge University Press, ISBN: 0521540518.

HAUAGGE, D. C., AND SNAVELY, N. 2012. Image Matching using Local Symmetry Features. *2012 IEEE Conference on Computer Vision and Pattern Recognition* (June), 206–213.

HORN, B. K. P. 1987. Closed-form Solution of Absolute Orientation using Unit Quaternions. *Journal of the Optical Society of America A 4*, 4 (Apr.), 629.

HORRY, Y., ANJYO, K.-I. A., AND ARAI, K. 1997. 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*, ACM Press/Addison-Wesley Publishing Co., 225–232.

HOWARD, R., 1988. Willow.

HUMAYUN, A., MAC AODHA, O., AND BROSTOW, G. J. 2011. Learning to Find Occlusion Regions. In *Computer Vision and Pattern Recognition*.

IGARASHI, T., MOSCOVICH, T., AND HUGHES, J. F. 2005. As-rigid-as-possible Shape Manipulation. *ACM Transactions on Graphics 24*, 3 (July), 1134.

KANG, H. W., AND SHIN, S. Y. 2002. 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*, ACM, 73–80.

KAV-VENAKI, E. 2010. Feedback Retargeting. In *Media Retargeting Workshop at ECCV*.

KAZHDAN, M., BOLITHO, M., AND HOPPE, H. 2006. Poisson Surface Reconstruction. In *Proc. Eurographics Symposium on Geometry Processing*, Eurographics Association, New York, NY, USA, 61–70.

KIRKPATRICK, S., GELATT, C. D., AND VECCHI, M. P. 1983. Optimization by Simulated Annealing. *Science 220*, 4598, 671–680.

KURZ, C., RITSCHEL, T., EISEMANN, E., THORMAHLEN, T., AND SEIDEL, H.-P. Camera Motion Style Transfer. In *Visual Media Production (CVMP), 2010 Conference on*, IEEE, 9–16.

LINKLATER, R., 2006. A Scanner Darkly.

LIPSKI, C., LINZ, C., BERGER, K., SELLENT, A., AND MAGNOR, M. 2010. Virtual Video Camera: Image-based Viewpoint Navigation Through Space and Time. *Computer Graphics Forum 29*, 8, 2555–2568.

LIPSKI, C., LINZ, C., NEUMANN, T., WACKER, M., AND MAGNOR, M. 2010. High Resolution Image Correspondences for Video Post-Production. In *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, IEEE, 33–39.

LIU, C., YUEN, J., TORRALBA, A., SIVIC, J., AND FREEMAN, W. T. 2008. Sift Flow: Dense Correspondence Across Different Scenes. *European Conference on Computer Vision (ECCV)*, 28–42.

LIU, F., GLEICHER, M., JIN, H., AND AGARWALA, A. 2009. Content-preserving Warps for 3D Video Stabilization. *ACM Transactions on Graphics 28*, 3 (July), 1.

LIU, F., GLEICHER, M., WANG, J., JIN, H., AND AGARWALA, A. 2011. Subspace Video Stabilization. *ACM Transactions on Graphics 1*, 212, 1–10.

LOURAKIS, M. I. A., AND ARGYROS, A. A. 2004. The Design and Implementation of a Generic Sparse Bundle Adjustment Software Package based on the Levenberg-Marquardt Algorithm. *ICSFORTH Technical Report TR 340*, 340.

LOWE, D. G. 2004. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision 60*, 2 (Nov.), 91–110.

MARROQUIM, R., KRAUS, M., AND CAVALCANTI, P. R. 2007. Efficient Point-based Rendering using Image Reconstruction. *Eurographics Symposium on Point-Based Graphics*.

MCCURDY, N. J., GRISWOLD, W., AND LENERT, L. 2006. A Robust Abstraction for First-Person Video Streaming: Techniques, Applications, and Experiments. *8th IEEE International Symposium on Multimedia (ISM'06)*, 235–244.

MCCURDY, N. J. 2007. *RealityFlythrough: A System for Ubiquitous Video*. Ph.d., University of California, San Diego.

MCMILLAN, L., AND BISHOP, G. 1995. Plenoptic Modeling: An Image-based Rendering System. In *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques*, ACM, vol. 95, 39–46.

MICROSOFT, 2008. Photosynth.

MORVAN, Y., AND O'SULLIVAN, C. 2009. Handling Occluders in Transitions from Panoramic Images: A Perceptual Study. *ACM Trans. Applied Perception 6*, 4, 1–15.

MURCH, W. 2001. *In The Blink Of An Eye*. Silman-James Press.

NOCEDAL, J., AND WRIGHT, S. J. 1999. Numerical Optimization (Springer Series in Operations Research). *Analysis*.

PÉREZ, P., GANGNET, M., AND BLAKE, A. 2003. Poisson image editing. *ACM SIGGRAPH 2003 Papers on - SIGGRAPH '03*, 313.

PRIT, Y., KAV-VENAKI, E., AND PELEG, S. 2009. Shift-map Image Editing. *2009 IEEE 12th International Conference on Computer Vision*, c (Sept.), 151–158.

SCHAEFER, S., MCPHAIL, T., AND WARREN, J. 2006. Image Deformation using Moving Least Squares. *ACM Transactions on Graphics 25*, 3 (July), 533.

SCHAUFLER, G. 1995. Dynamically Generated Impostors. In *GI Workshop on Modeling Virtual Worlds*, 129–135.

SHI, J., AND TOMASI, C. 1994. Good Features to Track. In *Computer Vision and Pattern Recognition*, IEEE, 593–600.

SNAVELY, N., SEITZ, S. M., AND SZELISKI, R. 2006. Photo Tourism: Exploring Photo Collections in 3D. In *ACM Transactions on Graphics (TOG)*, ACM, vol. 25, 835–846.

STEVENS, G., 1951. A Place In The Sun.

SUN, D., ROTH, S., AND BLACK, M. J. 2010. Secrets of Optical Flow Estimation and their Principles. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, IEEE.

TANEJA, A., BALLAN, L., AND POLLEFEYS, M. 2011. Modeling Dynamic Scenes Recorded with Freely Moving Cameras. *Computer Vision — ACCV 2010*, 613–626.

THALIN, G., 2012. Deshaker.

THOMPSON, D. W. 1917. *On Growth And Form*. Cambridge University Press.

THORMÄHLEN, T. 2006. *Zuverlässige Schätzung der Kamerabewegung aus einer Bildfolge*. PhD thesis, Universität Hannover.

TOMASI, C., AND KANADE, T. 1991. Detection and Tracking of Point Features. *Carnegie-Mellon University Technical Report CMU-CS-91-132*, April.

TOMPKIN, J., KIM, K. I., KAUTZ, J., AND THEOBALT, C. 2012. Videoscapes: Exploring Sparse, Unstructured Video Collections. *ACM Trans. Graph. (Proc. SIGGRAPH) 31*, 4.

TRIGGS, B., MCLAUCHLAN, P. F., HARTLEY, R. I., AND FITZGIBBON, A. W. 2000. Bundle Adjustment — A Modern Synthesis. *Vision Algorithms: Theory and Practice*, 153–177.

VANGORP, P., CHAURASIA, G., LAFFONT, P.-Y., FLEMING, R. W., AND DRETTAKIS, G. 2011. Perception of Visual Artifacts in Image-based Rendering of Façades. In *Eurographics Symposium on Rendering*, vol. 30.

WANDELL, B. A. 1995. *Foundations of Vision*, first ed. Sinauer Associates Inc.

2010. Wikipedia - Lap Dissolve.

WIKIPEDIA USER: GRM_WNR, 2011. Wikipedia - 180 degree rule.

ZACH, C., KLOPSCHITZ, M., AND POLLEFEYS, M. 2010. Disambiguating Visual Relations using Loop Constraints. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, IEEE, 1426–1433.

ZHANG, Z. 1994. Iterative Point Matching for Registration of Free-form Curves and Surfaces. *International Journal of Computer Vision 13*, 2 (Oct.), 119–152.