Vision, Modeling, and Visualization (2016), pp. 1–9
M. Hullin, M. Stamminger, and T. Weinkauf (Eds.)

Swipe Mosaics from Video

Paper 1029

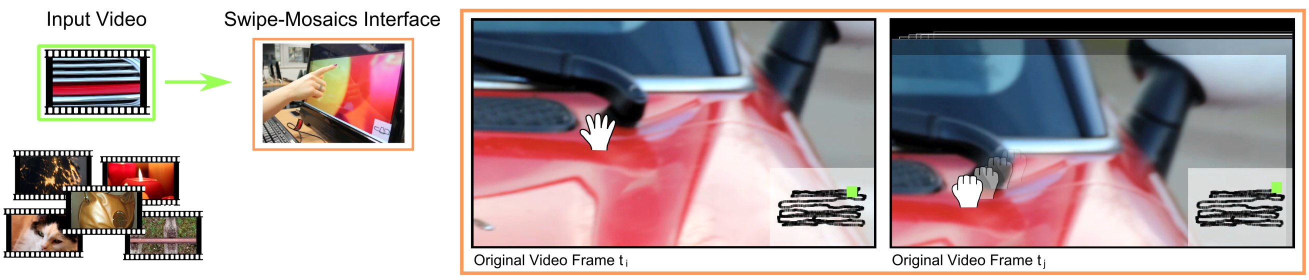


Figure 1: *Left:* Input videos containing "difficult" phenomena are used as inputs to our system. *Right:* The Swipe Mosaic interface allowing navigation over an image sequence. We propose Swipe Mosaics as an algorithm and associated interface which composites video frames into a content-centric navigable visualization. Some videos can already be browsed *spatially* by using existing mosaicing or IBR methods. Our system broadens the range of usable videos because it is trained to tolerate scene motion, parallax, repeated structure, and lack of texture.

Abstract

A panoramic image mosaic is an attractive visualization for viewing overlapping photos, but its images must be captured correctly to produce an acceptable composite. We propose Swipe Mosaics, an interactive visualization that places individual video frames on a 2D planar map that represents the layout of the physical scene. Compared to traditional panoramic mosaics, our capture *is easier because the user can both translate the camera center and film moving subjects. Processing and display degrade gracefully if the footage lacks distinct, overlapping, non-repeating texture. Our proposed visual odometry algorithm produces a distribution over (x,y) translations for image pairs. Inferring a* distribution *of possible camera motions allows us to better cope with parallax, lack of texture, dynamic scenes, and other phenomena that hurt deterministic reconstruction techniques. Robustness is obtained by training on synthetic scenes with known camera motions. We show that Swipe Mosaics are easy to generate, support a wide range of difficult scenes, and are useful for documenting a scene for closer inspection.*

1. Introduction

The success of Microsoft's Photosynth [Pho12] demonstrates that people wish to capture environments for later navigation. In the case of a video it is intuitive to navigate *spatially*, rather than in the *temporal* order it was captured. The works of [GGC*08], [KWLB08], [DRB*08], and [NNL13] explored the *direct manipulation* of video. They map a user's click-drag strokes to a sequence of frames elsewhere in the timeline (with variations). The location of the click and the direction of the mouse indicate which pixels and what point or optical flow trajectory to query in the sequence as a whole. We seek a similar direct user interaction for spatial navigation of scenes, which preserves the film's points of view and the veracity of the images. For example, imagine needing to inspect the car in Fig. 1 to place a bid for it in an online auction, or to examine scratches after a crash. Our system allows casually captured video footage to be automatically converted, under some simple assumptions, into a navigable "Swipe Mosaic".

Image Based Rendering (IBR) techniques can be used both to composite static/dynamic mosaics [IAH95] and for 3D browsing [SSS06, GAF*10]. They depend on either accurate optical flow estimation or interests points, for 2D/3D pose estimation. Both optical flow and interest points require texture. However, many everyday scenes lack texture, or otherwise break the assumptions of current IBR and direct video manipulation systems. Our proposed IBR approach gracefully degrades for difficult scenes, maintaining both rendering quality and user interaction. Towards the objective of intuitively navigating video, we present the following contributions:

- A regressor trained with synthetic data, that learns the relationship between image pairs and their 2D Euclidean transform.
- A layout method that uses the probabilistic pairwise predictions from the regressor to produce a 2D location for each image, including detecting and optimizing loop-closures.
- A Swipe Mosaic interface, shown in Fig. 1, to display the video frames, allowing the user to perform content centric navigation and inspection by "swiping" scene elements. The interface runs

as either a native application or in a web browser, and can be used on a smart phone.

In contrast to regular panoramic image mosaicing approaches, our system can analyze and visualize hand-held camera footage with parallax, blur, textureless areas, specular areas, and moving subjects. The visualization quality degrades gracefully in the case of especially difficult scenes.

2. Related Work

Since the genesis of Image Based Rendering (IBR) for synthetic data [CW93], steady progress has been made toward beautiful and useful renderings of real world footage. Footage usually comes from multiple viewpoints, so progress is inherently dependent on having accurate estimates of relative camera poses. Here we summarize the most relevant interactive IBR approaches, starting with techniques for estimating the needed camera parameters.

Camera Poses A comprehensive summary of methods for converting video frames into planar and cylindrical mosaics is presented in [Sze96], while [SS97] cover spherical mosaics. They explain how stitching an image mosaic is easiest when all the images can be related to each other by homographies. Such registration benefits from either manual or interest-point based initialization, and assumes that the scene is textured. Textured scenes ensure convergence when minimizing the residual difference in the intensities of overlapping pixels. Texture can also help when mosaicing an image sequence, because optical flow is strongly correlated with visual odometry [Cam04]. Approaches such as DTAM [NLD01] estimate camera pose for textured scenes. We too benefit from texture in the scene, but are less reliant on it.

Initializing camera poses can be difficult in practice, even in textured scenes. Hardware attached to the camera can help [ATP*10], as demonstrated by [YN01] who fused visual cues with gyroscope data and [KD04] who used an inertial sensor to mitigate blur. [KUDC07] actively controlled the camera pose using a motorized telescope mount to stitch mosaics of thousands of photos. There are numerous other hybrid systems which fuse other data with images, but even [KUDC07], [WMLS10], and the Photosynth App [Pho12] rely on interest point matching to register their images. The SIFT detection and features of [Low04] remain the standard by which interest point detection and matching is measured [TM07]. Finding enough matching interest points in an image collection means that photos can be registered to each other, adjusted for exposure, and blended into a large mosaic [BL03]. At least four points must be matched to compute the projective transform between two images, but in practice 10's and 100's of points are used with RANSAC [FB81] to robustly calculate an answer. The same approach and inflated number of distinct interest points is normal for estimating the translation and rotation of the 2D Euclidean transform, even though two corresponding points is enough, and solutions with corresponding lines and curves also exist [HZ06]. The key issues are that large areas of real images have light or sparse texture, and that seemingly corresponding points may not represent the same 3D point in the world because of scene motion, motion blur, reflection, or repeated structures [Sze06].

When building mosaics or other IBR and multi-view scene models, camera pose estimation is overwhelmingly seen as a self-contained problem. Even [Dav98], whose system was designed to cope with moderately-sized moving objects and rotation-only cameras, performs global optimization by treating all the estimated pairwise camera-transforms as equally good. In contrast, our regressor (§3.1) reports high uncertainty for less textured or more dynamic scenes, and the subsequent layout computation (§3.2) incorporates this uncertainty. Swipe Mosaic visualizations can better cope with difficult (though typical) footage because we work with distributions rather than committing too early to interest-point matches or specific Euclidean transform parameters.

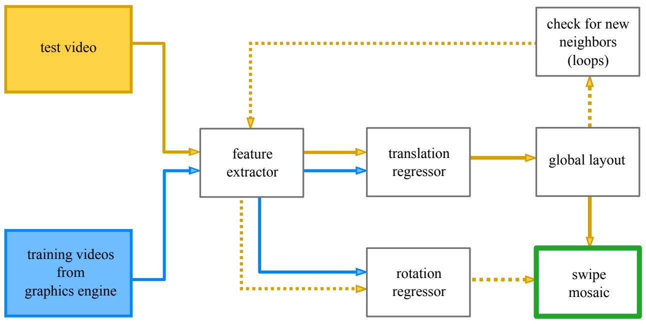


Figure 2: System diagram illustrating how a video is analyzed to generate a Swipe Mosaic. Blue lines indicate the offline training process. Dotted yellow lines indicate post processing steps, which take place after an initial layout is found.

Probabilistic distributions on locations have been applied before, such as to help a "teleporting" robot with a range sensor localize itself in a known floorplan [TFBD01]. Probabilistic models are increasingly employed in Structure from Motion (SfM) too [DRMS07]. SfM classically requires running RANSAC over more suggested interest-point matches than the Euclidean transform (five are needed at minimum). SfM then estimates 3D camera poses and 3D scene point locations, and finally optimizes these estimates globally using repeated steps of Bundle Adjustment (BA) [HZ06]. The stages of SfM are normally deterministic and notoriously computationally expensive, but we are particularly inspired by the recent work of [COSH11] who use a less costly optimization to compute an initialization for a single iteration of BA. They convert the deterministic pairwise estimates to probabilistic constraints on a graphical model, which they solve with Loopy Belief Propagation [MWJ99]. The probabilistic approach gives a principled method of incorporating other information, such as geotags. Instead of replacing the final half of the BA pipeline with a probabilistic system, we propose to model pose probabilistically from the beginning.

Rendering & Interaction Much like the direct manipulation works mentioned already and our own interface, Dynamic Mosaics [GS12] prominently display for interaction a current frame from the input footage. Their rendering method occupies a middle ground between ours and that of classic image mosaics, in that they dynamically stitch onto that frame *some* spatially neighboring frames, choosing neighbors which share a large number of inlier correspondences. This obviously limits the variety of scenes which can be displayed, so they have an alternate mode based on the similarity transform, which requires somewhat fewer correspondences. We require no explicit correspondence points.

Interest-point based registration with subsequent Bundle Adjustment has allowed numerous interesting IBR prototypes to emerge. Panoramic Video Textures (PVT) [AZP*05] register and play video clips inside an otherwise static cylindrical panorama. A competing PVT system [RAPLP05] allows parts of the XYT-volume to be played back in different order, *e.g.* making explosions look like implosions. Also reliant on interest point matches but with an alternative optimization to BA [LGW*11] are able to stabilize shaky videos to follow different target trajectories.

The Lumigraph [GGSC96] and Light Field Rendering [LH96] cleverly allow the user to recombine the rays captured by an array of cameras. Interfaces allow users to navigate the plenoptic function spatially, and to simulate new focal lengths. [SH99] showed a hardware based system for capturing a reduced-size 3D plenoptic function. The recent system of [DLD12] massively simplifies the process of capturing light fields by giving fast feedback about what parts of the static scene have been adequately filmed. They employ the PTAM [KM07] real-time SfM system which registers their cameras if enough interest points are available, and the camera does the characteristic "SLAM wiggle" [HKM09].

[AAC*06] discuss the differences between strip panorama systems, and propose a multiviewpoint panorama which stitches together large regions of photos that were shot with a hand-held camera. The strength of their interface is that users can override the stitching to (de)emphasize perspective effects in different parts of the scene. Their system relies on the Bundler SfM system [SSS06] for camera registration. The Street Slide system of [KCSC10] shows another interface to multiviewpoint panoramas, which was part of our motivation for a 2D interface. The Photo Tourism work of [SSS06] and [SGSS08] was instrumental both for releasing Bundler and the insight that sufficiently large photo collections could be browsed in 3D. When images show the same objects or objects in-the-round, the viewer's transitions are rendered smoothly, and [GAF*10] offer especially smoothed transition effects for images that are very far apart in 3D. These systems prefer to cull low-texture and low-quality images, and endeavor to eliminate moving objects from their collections. In contrast, our users are filming video of something specific for interaction in a 2D swipe interface, need *that* sequence to work, and may not have the benefit of static scenes and distinct interest points.

3. Swipe Mosaic Construction

Our system takes as an input a video sequence or temporally ordered set of images $\{I_1, I_2, \ldots, I_N\}$ and presents them in a new type of interactive mosaic. Valid inputs to our system include scenes which could be used to create a panoramic mosaic, but also include scenes containing significant parallax and dynamic objects, so the Swipe Mosaic avoids trying to stitch all the inputs together seamlessly. As an overview of our approach, we first select pairs of images and make predictions of the *relative* camera motion for each pair, before combining those predictions using a global least squares optimization. Predictions are made with random regression forests, trained on synthetic data. The predictions form a distribution over possible camera motions. The layout algorithm locates the images on a 2D manifold so they can be visualized using our Swipe Mosaic interface. Finally, several postprocessing steps may be performed to further improve the viewing experience. The overall pipeline of our visual odometry regressor and layout system is shown in Fig. 2.

Pair selection generates a set $\mathcal{P} = \{(j_1, k_1), \ldots\}$, following which camera motion will be estimated between image pairs $\{(I_{j_1}, I_{k_1}), (I_{j_2}, I_{k_2}), \ldots\}$. A number of strategies can be employed to select pair indices – some selection is necessary as comparing $O(n^2)$ pairs is computationally infeasible for large sequences. It is possible to anticipate loops in the ordered set by finding image pairs which are temporally distant but show the same location. Possible techniques for modeling such similarity include SIFT matching [Low04], GIST scene descriptors [OT06], simple L2 intensity distance, or geodesic distance models such as Isomap [TDSL00]. We evaluated these methods but achieved superior results by initially picking only close temporal neighbors, and finding loop closures at a later stage (§3.3).

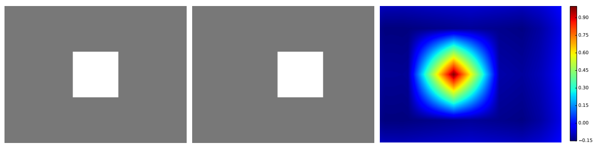
3.1. Learning and Inference on Image Pairs

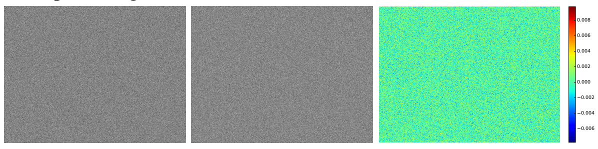
We seek a probabilistic estimate of the camera motion between a pair of images. To that end, we use Regression Random Forests (RRF) [CSK11], in turn based on Random Forests (RF) [Bre01, Ho95]. Other supervised learning algorithms could have been used, but RRFs produce inherently probabilistic multivariate output making them an excellent fit. Testing on unseen data produces a distribution of predictions, one from each tree. We fit a Gaussian to these predictions to obtain a parametric distribution, but in principle, the raw distribution could be used. As well as their probabilistic nature, RRFs train and test quickly, can handle high dimensional feature vectors, and are trivially parallelizeable. RF algorithms have been successfully applied in a range of applications, including human pose recognition [SFC*11] and supervised mesh segmentation [KHS10]. Relative interframe motion is modeled here using the 2D Euclidean transform, so whether training or testing, the label-space consists of three degrees of freedom: two for translation and one for rotation. In practice, we build an RRF for translation and an essentially identical RRF for rotation to reduce the amount of training data needed. The rotational RRF is trained to predict small camera rotations around the optical axis and is used for postprocessing. In both cases the training is with synthetic data, as documented in the appendix. Differences between the two RRFs are highlighted in §3.3 and §4.

3.1.1. Feature Computation

A 3599 dimensional feature vector is extracted from each image pair by encoding the responses of many Normalized Cross Correlation (NCC) comparisons between different segments of the images. NCC was chosen because it concisely describes many properties of image pairs. Images containing similar, distinctive, localizable content produce unimodal NCC responses (Fig. 3a). Textureless or uniform input images produce approximately flat NCC responses (Fig. 3b). Images with repeated structure produce periodic NCC responses (Fig. 3c). Our feature extraction aims to detect and encode all these situations, allowing the RRF to learn the mapping between NCC responses and camera movement.

For image index pair $(j,k) \in \mathcal{P}$ we take I_j as the template image and I_k as the search image. A pyramid of patches is defined by placing regular $1\times1, 2\times2, 4\times4, 6\times6$ and 8×8 grids onto each image (the 6×6 case is shown in Fig. 4). This approach of taking


(a) Image pair (left/middle) with a strong texture, producing a unimodal NCC response (right).


(b) Textureless image pair (left/middle) producing flat NCC response (right).

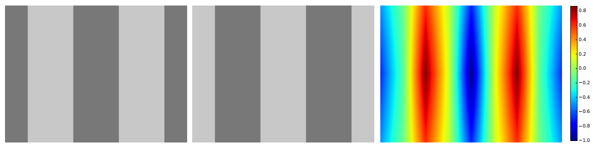

(c) Image pair (left/middle) with repeated structure producing multimodal NCC response (right).

Figure 3: Representative types of image pair we may see (left, middle), along with their corresponding NCC response (right).

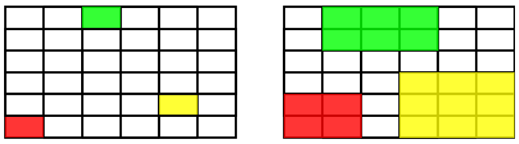


Figure 4: Template (left) and search (right) patches representing some combinations of image regions on which NCC is run (6×6 grid level). Color reflects correspondence. Note the edge truncation behavior.

patches at different scales and areas at each grid resolution in the template image is compared using NCC to a region of the search image creating a response image N. To allow for scene movement between the images, each template patch is compared to a larger region in the search image, by expanding out 1 patch in each direction unless the edge of the image prevents this. In Fig. 4 the colored patches indicate representative examples of regions that would be compared. This first step (FEATEXTRACTIMG in the pseudo-code, found in the appendix) produces 121 different NCC responses, each of which is then encoded to a few numbers, the concatenation of which forms our full feature vector.

A strongly peaked NCC response indicates a likely offset for the scene content (*i.e.* this portion of the scene contains localizable texture). In this case providing the location of this offset to our machine learning system is crucial, as (for example) if every NCC comparison contained a strongly peaked offset to the left, this is strong evidence that the camera has moved to the right. If the response is relatively flat, *i.e.* the peak value is close to the mean value, then the patches compared are likely textureless and so no definitive decision can be made about the camera motion. Note that this absence of certainty is an equally important input to our machine learning technique (it may make the RRF more likely to output a large variance). The NCC image containing a "ridge" (peak only localizable in 1 dimension) indicates certain types of scene geometry (and certain degrees of belief about possible motion) and so must also be concisely encoded. Each NCC response is encoded as follows (ENCODENCC), and the concatenation of all these defines our feature vector.

Minimum, maximum and mean values are computed to give an idea of the response distribution, particularly how the peak value compares to the rest of the values. The 2D offset of the peak location is found, and the sizes of the input patches are used to convert it to a normalized offset, so that when the input patches are exactly the same and the peak indicates this, the offset $(0,0)$ will be added to the feature vector. The shape of the response is captured by computing Laplacian Coordinates around the peak point. We apply the Laplacian operator $\frac{1}{4}\begin{bmatrix}1 & -2 & 1\end{bmatrix}$ to 1D "slices" through the 2D surface of N. These slices all coincide with the peak, and are made at 4 different orientations and two different scales (so the points away from the peak which are multiplied by 1 are either 10 or 20 pixels away). This 8D descriptor encodes the shape of the peak - if all the numbers are large then the peak is strongly localizable in all directions. If all the numbers are close to zero this means the peak is very wide, and if (for example) the numbers for horizontal "slices" are low and the numbers for vertical slices are large, the peak is a horizontal "ridge". A normalized histogram with 5 equally spaced bins between -1 and 1 is also stored.

The final step, which we only carry out for the 1×1 and 2×2 grid resolutions, is to run a Gabor filter bank $\mathcal{G}$ over the image. The filters $G \in \mathcal{G}$ vary in orientation, scale and frequency in order to try to capture the multimodal NCC response produced by images with repeating structure. The min, max, mean and median of each Gabor response is stored, thus completing our feature encoding scheme. The parameters used to generate the filter bank can be found in the supplemental material.

3.2. Layout to Global Coordinates

The pairwise estimates provided by the Random Forest use a relative coordinate system, but to navigate images as a Swipe Mosaic we require image locations in a global coordinate system. By limiting motion to a 2D plane and approximating the RRF output as Gaussian, we solve this problem in closed form using linear least squares in a similar spirit to [OLT06]. For each $(j,k) \in \mathcal{P}$, the RRF gives a mean $\mu_{jk} = [\mu^x_{jk} \quad \mu^y_{jk}]^T$ and standard deviation $\sigma_{jk} = [\sigma^x_{jk} \quad \sigma^y_{jk}]^T$. An error function

$$E(x) = \sum_{j,k \in \mathcal{P}} \left(\frac{(x_k - x_j) - \mu^x_{jk}}{\sigma^x_{jk}} \right)^2 + \left(\frac{(y_k - y_j) - \mu^y_{jk}}{\sigma^y_{jk}} \right)^2 \qquad (1)$$

is defined on the vector of all camera locations

$$x = \begin{bmatrix} x_1 & y_1 & x_2 & y_2 & \dots & x_N & y_N \end{bmatrix} \qquad (2)$$

by summing squared differences between the *actual* pairwise offsets in x and the *predictions*, weighted by the prediction uncertainty. $E(x)$ can be written as $e(x)^T e(x)$ where

$$e(x) = \begin{bmatrix} \frac{(x_{k_1} - x_{j_1}) - \mu^x_{j_1 k_1}}{\sigma^x_{j_1 k_1}} & \frac{(y_{k_1} - y_{j_1}) - \mu^y_{j_1 k_1}}{\sigma^y_{j_1 k_1}} & \dots \end{bmatrix}^T. \qquad (3)$$

$e(x)$ can be written as $Ax - b$ where each row of A contains zeros in all but two entries, at locations to match the variables in x, containing positive and negative inverse standard deviation, and the corresponding element of b is the mean prediction from the forest divided by the standard deviation. Two more rows are added to A and b to overdetermine the system by forcing (x_1, y_1) to be $(0,0)$. The unique solution for x is determined by solving the least squares equation, $Ax = b$.

3.3. Post-processing

Translational Loop Closure

Errors, however small, will accumulate over long sequences, so that loops in the camera path may not line up correctly. To mitigate this we automatically find "loop points" – pairs of images which are temporally distant but close in the 2D coordinate space. We compute each image's 5 nearest spatial neighbors, and any neighbor whose timestamp differs by more than 25 frames becomes a loop point. The number of pairs chosen by this technique is typically small. For each loop point image pair we compute a feature vector and corresponding prediction from the translational RRF. A new layout is computed from the combined set of temporal and loop point predictions.

Rotational Correction

Video recorded freehand will include rotation around the optical axis. When transitioning between frames in the viewer (§3.4) the differences will be small if moving in order, but at a loop point the angular change can be large. Even a few degrees is noticeable, so rotation is corrected for. A second "rotational" RRF was built using the same features as before, but trained to predict optical axis rotation between two images. Synthetic training data was again used, this time with random rotations around the optical axis. As with the translational loop closure, we predict rotations for image pairs which are found at "loop points". Note that the estimated translation is applied before calculating the rotation, as the RRF is trained on rotation without translation. Relative rotation estimates are combined using an analogous technique to the least squares layout algorithm (§3.2).

3.4. Swipe Mosaic Interface

Versions of the viewer interface exist to run both natively and via web app, with the web app running on a mobile phone. In both cases the interface shows the current image in sharp focus, with the surrounding images optionally shown blurred. The user navigates by clicking anywhere on the image and dragging ("swiping"). The swipe determines where in the 2D map the program looks for a new image to transition to. This behavior is similar to PDF viewers and online mapping services. A "minimap" conveys the overall layout of the scene, and highlights the location of the currently displayed image. As an alternative navigational aid, users can enter "Picasso view" which zooms out and displays all of the images. These images will not line up perfectly as in a panoramic mosaic, but provide a sense of the scene as a whole. Smooth transitions are used throughout, with the intention that if the user clicks on a recognizable feature that feature will remain under the cursor as they swipe. Please see supplementary video to see it in action.

Sequence	Characteristics
FENCE∗	Easy sequence, abundant texture
MINI	Abundant texture
LOBBY∗	Abundant texture, demonstrates loop closure
FACADE	Abundant texture
GRATING	Partially textureless
RAILS	Large scale repeated structure
SKATER	Dynamic and deforming foreground object
FLOWERS†	Dynamic objects, non planar motion, [GF12] failure case
SCULPTURE	Little texture, motion blur
LEAVES	Dynamic and deforming scene
OBELISK	Non planar camera path
HANDBAG	Dynamic specularities
WALL	Ambiguous motion due to repeated structure
VINYL	Motion blur, textureless occluding object
ISS∗	Demonstrates Loop closure
DINO	Moving background elements
PRISM	Automatic gain control, CCD overload
AQUATIC†	Scene from movie, contains dynamic objects + parallax
FREIBURG2†	6D Ground truth available

Table 1: Test sequences along with their defining characteristics.
∗ – only appears in supplemental material.
† – not captured for purpose of building a Swipe Mosaic.

4. Experiments

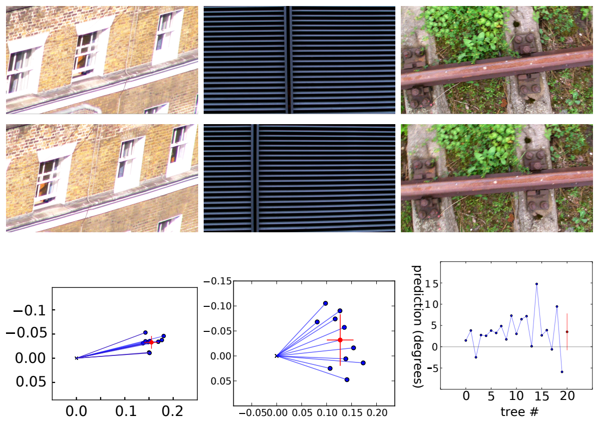
Experiments were performed on videos filmed by our users using mobile phones, camcorders and SLRs. Videos recorded for other purposes were also used. We examine the performance of the pairwise regressor, and the overall Swipe Mosaic rendering and visualization system quality. Qualitative and quantitative comparisons are performed on a variety of scenes, listed in Table 1 with short descriptions.

Many baseline algorithms that produce camera paths on simple, textured, static, planar scenes are compared to. When they work they also prepare a sequence for use as a Swipe Mosaic. We demonstrate that our approaches degrades gracefully with footage that is less simple, in a variety of ways.

4.1. Regressing Motion Between Image Pairs

We start by inspecting what the regressor has learned about the relationship between the computed features and estimating translations. When images contain unambiguous texture (Fig. 5a) our regressor is confident in both x and y. For scenes with repeated texture but a unique vertical structure (Fig. 5b), the regressor outputs a small σ^x and large σ^y, reflecting the certainty in x and uncertainty in y (aperture problem). Fig. 5c show the result of using the same type of regressor and features, but training to estimate the in-plane rotation between two images. We expect the rotational RRF to perform better with a customized feature vector, but the vector designed for translation allowed rotational correction within $\pm 5^\circ$.

The RRF training process selects some elements of the feature vector more frequently than others for estimating the transform parameters at test time. Fig. 6 shows spatial histograms for each scale of the NCC grid, indicating the frequency with which a feature computed from that NCC sub-window was chosen by the forest training process. Interestingly, the most used level is the 4×4 grid, and there is a strong peak within that histogram for the most central 4 of the 16 possible NCC sub-windows. While using a single



(a) FACADE (b) WALL (c) RAILS

Figure 5: In each column the top two images were inputs resulting in the RRF output at the bottom. The graphs in a) and b) show 2D results from the Translational RRF, and c) shows 1D results from the Rotational RRF. Blue dots show individual tree outputs, red dots and bars show the mean and variance of the fitted Gaussian.

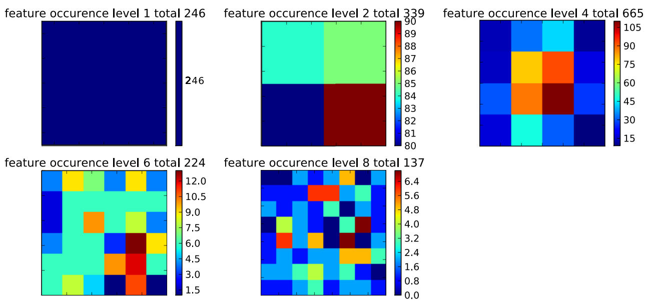


Figure 6: Spatial histograms showing the quantity and arrangement of features chosen most frequently during training from each level of the NCC grid, for our translational RRF.

NCC calculation to compare whole images is a common approach for image alignment, these graphs show that the finer grained NCC sub-windows are providing important extra information to get the right offset. The top level of features (representing a single global NCC comparison) are chosen by the forest training process 246 times, whilst the 4×4 resolution features are chosen 665 times. We know that this is due to these features being more informative, rather than simply more numerous, because the 8×8 level, which contains a larger number features than all the other levels put together, is only chosen 121 times.

4.2. Swipe Mosaic Visualization

A key property of Swipe Mosaic visualization is being able to grab elements and navigate spatially between temporally distant frames. As shown in the supplementary video, it is possible to maintain a sense of position while navigating within the wider scene. Possible applications for the system include recording the damage of a car accident for later scrutiny, or examining products when shopping online.

4.3. Suitable Image Sequences

Swipe mosaics are best observed in motion, so we present qualitative results in the video. Our system performs best when the camera motion and scene geometry are parallel and both approximately planar (as with the training data), but we are robust to deviations from this setup. OBELISK shows that if the object of interest fills much of the screen, we infer a 2D version of the motion as the camera *rotates around* the object. DINO contains people in the background moving in various directions, but the transforms computed allow navigation along the main item of interest. The level 4 histogram in Fig. 6 helps explain this; the RRF learns that the center of the image is usually more informative, and so treats the motion implied by central pixels as more informative than that implied by edge pixels. View dependent effects such as the specularities in the HANDBAG sequence do not lead to incorrect motion estimates. SKATER, however, features non rigid movement in the *centre* of the frame and only the *outskirts* imply the (correct) sideways motion.

To demonstrate the utility of our system on existing sequences filmed by others, we processed a scene from the movie "The Life Aquatic with Steve Zissou". AQUATIC features the camera panning over a cutaway version of a boat, traveling between rooms and showing different characters. The camera trajectory roughly matches the assumptions made in our training data, but the scene contains large amounts of parallax due to depth variation, as well as dynamic characters. We put 600 frames into our system and built a swipe mosaic which allows intuitive navigation between seven distinct areas. Sample frames from transitioning "through" a wall are shown in Fig. 7. Please see the supplemental video to view this scene in motion.

Sequences such as HANDBAG can be processed successfully despite containing out-of-plane camera translation and strong specularities. Note that if a loop point featured images with differing scale, this would pose a problem for our system both in terms of the loop closure algorithm and in terms of viewer artifacts. To test the limits of our system, we ran it on a challenging part of the FLOWERS scene, a failure case from [GF12], which the authors describe as troublesome due to the pedestrians occluding geometry and cutting feature trajectories. The camera is moving forwards as well as sideways so that the contents of the flower stall appear to be moving roughly diagonally in image space. Our system copes with this motion, and we can browse the scene by swiping elements on the flower stall. This challenging video also demonstrates a failure mode of our system; obstructing bystanders in the image center, combined with motion that differs significantly from that of our training data, makes for a very difficult scene.

4.4. Qualitative Evaluation against Baseline Algorithms

We evaluate the performance of our odometry regressor by comparing to simple NCC based alignment [Sze06], VisualSfM by Wu *et al*. [Wu07, WACS11], Viewfinder Alignment by Adams *et al*. [AGP08], Real-time image-based tracking of planes using Efficient Second-order Minimization (ESM) by Benhimane & Malis [BM04], and "microSfM" or "μSfM", a new system which uses the methodology of Fundamental Matrix computation but produces a 2D translation. While these techniques can be applied to a wide range of sequences, we confirmed known circumstances

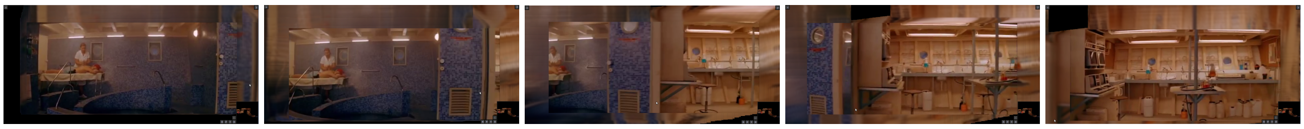


Figure 7: A series of screenshots taken during a single swipe, navigating AQUATIC, a scene generated from the movie "The Life Aquatic with Steve Zissou"

under which each of the baselines failed, and our technique succeeded. Please see the Appendix and supplementary video for details.

4.5. Quantitative Evaluation Against NCC

While graceful degradation is easy to illustrate qualitatively, it is reasonable to check if regression using our NCC-based feature vectors is actually different than just using NCC directly, at least for best-case in-plane motion and static scenes with negligible motion blur. To quantitatively evaluate the odometry of our system, we compare on the FREIBURG2 sequence from the TUM [SEE*12] dataset. This dataset consists of Kinect video sequences alongside 6D ground truth camera positions/rotations, generated using 100Hz active motion capture. Unlike the rest of the dataset, the camera path in FREIBURG2's first 950 frames contains almost exclusively vertical and horizontal translation. This is the kind of motion that both NCC and our system should be able to handle. The appearance of the indoor office scene is unremarkable in terms of texture or dynamic elements.

Both NCC and our system were used to generate 2D camera locations for the sequence. Every pair within a 4 frame temporal window was used to generate relative offset predictions. For both systems, the offsets were fed to our least squares layout algorithm. To compare the ground truth with the 2D solution the 6D positions were projected to 2D – see appendix for details.

NCC gave a final Mean Squared Error of 14.2 cm^2 against our system's 12.9 cm^2, an improvement of 9.4%. The alignments resulting from both systems are shown in Fig. 8. It is interesting to note that both algorithms make similar mistakes, owing perhaps to our system being built on top of NCC based features, but our system produces errors of smaller magnitude, especially towards the center of the horizontal axis, because it incorporates more information than just the top level, entire-image NCC comparison. As well as comparing with this fitted plane, we tried aligning to each of the planes defined by one individual camera's orientation, by projecting all other cameras onto the local "right" and "up" vectors. The results of this are in the appendix. Unsurprisingly, whichever one of the 950 cameras we choose, we see an improvement with our system, as shown by the green line always being underneath the blue line. This evaluation shows that even for a texture-rich real world sequence, not captured for Swipe Mosaics, our system produces a measurable improvement over the alignment produced by NCC.

Our evaluations show our robustness to visual phenomena found difficult by other systems. In the presence of dynamic objects, lack of texture, or repeated structure, we are able to compute 2D locations and browse the scene, degrading gracefully in the presence of inconclusive visual information. Many of our sequences were captured by users unfamiliar with the workings of the system, and our success here demonstrates our robustness to input data straying outside the assumptions of the training data.

5. Conclusions

Swipe Mosaics can be used to intuitively navigate video sequences, including those where existing approaches struggle or fail. As seen in the supplementary video, the Swipe Mosaic interaction provides a good sense of a scene's layout. Traditional panoramic image mosaics undoubtedly have a cleaner overall appearance than our Picasso-view, but we have obtained robustness in the presence of view dependent effects, such as parallax and specularities, and preserve the veracity of the image. Our RRFs, trained on synthetic data, estimate visually-acceptable translations and rotations both in textured scenes, where existing methods also work well, but also in difficult scenes, where other methods become brittle or fail.

Motions omitted from the training set may perform badly, though not always. For instance, forward motion is not in the training set, yet the approach remains robust to it. However, as shown by the FLOWERS example, occluders in the image center can cause failure. A valuable improvement to the interface would be to give live feedback at capture time regarding which parts of the scene require more detailed recording, *e.g.* as in [DLD12]. Swipe Mosaics will hopefully encourage content-creators to document and share the world around them.

References

[AAC*06] AGARWALA A., AGRAWALA M., COHEN M., SALESIN D., SZELISKI R.: Photographing long scenes with multi-viewpoint panoramas. In *TOG* (2006). 3

[AGP08] ADAMS A., GELFAND N., PULLI K.: Viewfinder alignment. *Eurographics* (2008). 6

[ATP*10] ADAMS A., TALVALA E.-V., PARK S. H., JACOBS D. E., AJDIN B., GELFAND N., DOLSON J., VAQUERO D., BAEK J., TICO M., LENSCH H. P. A., MATUSIK W., PULLI K., HOROWITZ M., LEVOY M.: The frankencamera: an experimental platform for computational photography. *TOG* (2010). 2

[AZP*05] AGARWALA A., ZHENG K., PAL C., AGRAWALA M., COHEN M., CURLESS B., SALESIN D., SZELISKI R.: Panoramic video textures. In *TOG* (2005). 3

[BL03] BROWN M., LOWE D. G.: Recognising panoramas. *ICCV* (2003). 2

[BM04] BENHIMANE S., MALIS E.: Real-time image-based tracking of planes using efficient second-order minimization. In *Intelligent Robots and Systems* (2004), vol. 1, pp. 943–948. 6

[Bre01] BREIMAN L.: Random forests. *Machine Learning* (2001). 3

[Cam04] CAMPBELL J.: Techniques for evaluating optical flow for visual odometry in extreme terrain. *IROS* (2004). 2

[COSH11] CRANDALL D., OWENS A., SNAVELY N., HUTTENLOCHER D.: Discrete-continuous optimization for large-scale structure from motion. In *CVPR* (2011). 2

[CSK11] CRIMINISI A., SHOTTON J., KONUKOGLU E.: Decision forests: A unified framework for classification, regression, density estimation, manifold learning and semi-supervised learning. *Found. Trends. Comput. Graph. Vis.* (2011). 3

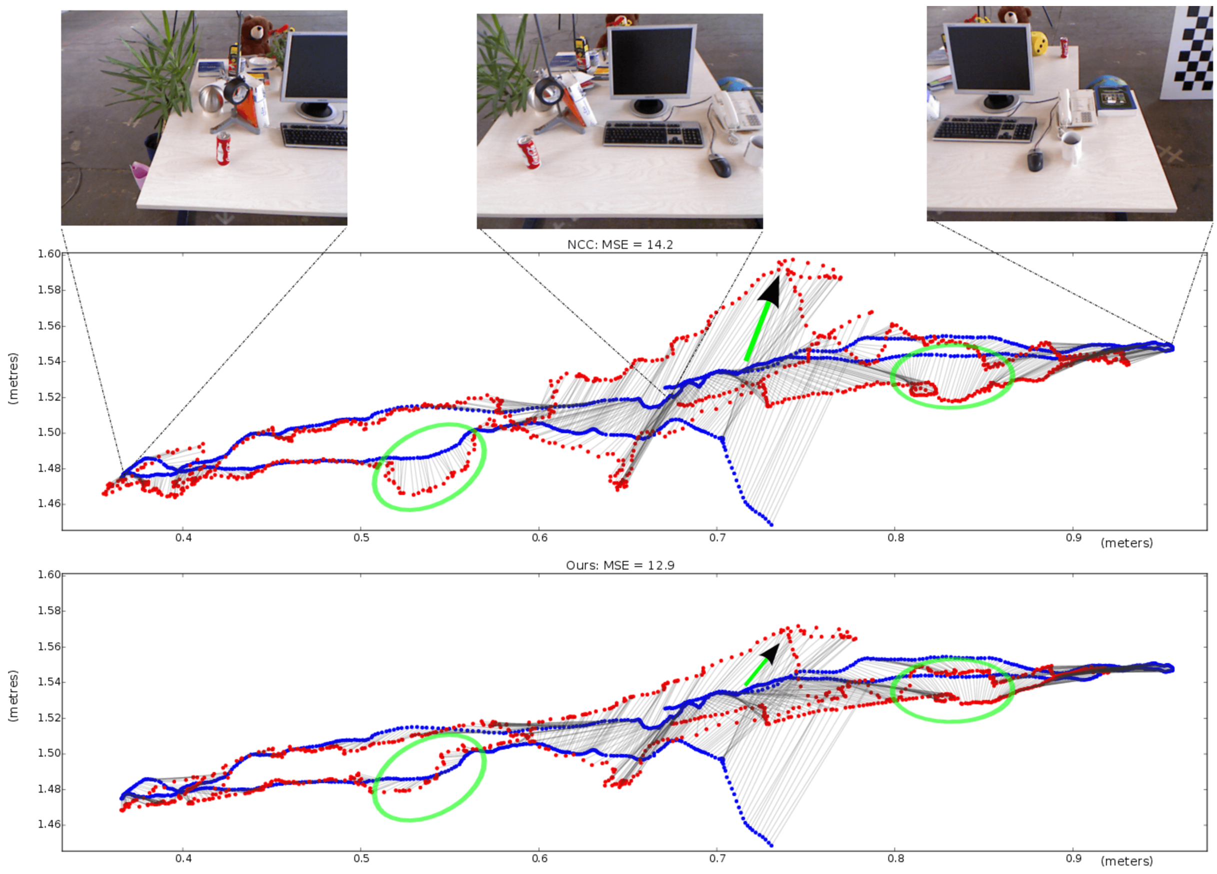


Figure 8: Alignment result for NCC (top) and our system (bottom) for the FREIBURG2 sequence [SEE*12]. Each graph shows the camera locations from the prospective solution (Red) after being aligned using Procrustes to the ground truth locations computed with a "best fit" plane (Blue). Corresponding camera locations are joined by grey lines. Note the highlighted regions (green ellipses / arrows) in which our system provides a solution substantially closer to ground truth. Navigating these areas of the NCC solution as a Swipe Mosaic would require users to follow a more distorted trajectory than would seem natural. Corresponding frames are shown above to give an idea of the scene makeup. Onscreen viewing recommended.

[CW93] CHEN S. E., WILLIAMS L.: View interpolation for image synthesis. *SIGGRAPH* (1993). 2

[Dav98] DAVIS J.: Mosaics of scenes with moving objects. *CVPR* (1998). 2

[DLD12] DAVIS A., LEVOY M., DURAND F.: Unstructured light fields. In *Computer Graphics Forum* (2012), vol. 31. 3, 7

[DRB*08] DRAGICEVIC P., RAMOS G., BIBLIOWITCZ J., NOWROUZEZAHRAI D., BALAKRISHNAN R., SINGH K.: Video browsing by direct manipulation. In *CHI* (April 2008). 1

[DRMS07] DAVISON A., REID I., MOLTON N., STASSE O.: Monoslam: Real-time single camera slam. *PAMI* (2007). 2

[FB81] FISCHLER M. A., BOLLES R. C.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM* (1981). 2

[GAF*10] GOESELE M., ACKERMANN J., FUHRMANN S., HAUBOLD C., KLOWSKY R.: Ambient point clouds for view interpolation. *TOG* (2010). 1, 3

[GF12] GOLDSTEIN A., FATTAL R.: Video stabilization using epipolar geometry. *ACM Trans. Graph.* (2012). 5, 6

[GGC*08] GOLDMAN D. B., GONTERMAN C., CURLESS B., SALESIN D., SEITZ S. M.: Video object annotation, navigation, and composition. In *UIST* (2008). 1

[GGSC96] GORTLER S., GRZESZCZUK R., SZELISKI R., COHEN M.: The lumigraph. *SIGGRAPH* (1996). 3

[GS12] GARG R., SEITZ S. M.: Dynamic mosaics. *3DIMPVT* (2012). 2

[HKM09] HOLMES S. A., KLEIN G., MURRAY D. W.: An o (n^2) square root unscented kalman filter for visual simultaneous localization and mapping. *PAMI 31*, 7 (2009), 1251–1263. 3

[Ho95] HO T. K.: Random decision forests. In *Document Analysis and Recognition* (1995). 3

[HZ06] HARTLEY A., ZISSERMAN A.: *Multiple View Geometry in Computer Vision (2. ed.)*. CUP, 2006. 2

[IAH95] IRANI M., ANANDAN P., HSU S.: Mosaic based representations of video sequences and their applications. In *ICCV* (1995). 1

[KCSC10] KOPF J., CHEN B., SZELISKI R., COHEN M.: Street slide: Browsing street level imagery. *SIGGRAPH* (2010). 3

[KD04] KLEIN G., DRUMMOND T.: Tightly integrated sensor fusion for robust visual tracking. *Image and Vision Computing* (2004). 2

[KHS10] KALOGERAKIS E., HERTZMANN A., SINGH K.: Learning 3d mesh segmentation and labeling. *TOG* (2010). 3

[KM07] KLEIN G., MURRAY D.: Parallel tracking and mapping for small AR workspaces. In *ISMAR* (2007). 3

[KUDC07] KOPF J., UYTTENDAELE M., DEUSSEN O., COHEN M. F.: Capturing and viewing gigapixel images. *SIGGRAPH* (2007). 2

[KWLB08] Karrer T., Weiss M., Lee E., Borchers J.: Dragon: A direct manipulation interface for frame-accurate in-scene video navigation. In *CHI* (2008). 1

[LGW*11] Liu F., Gleicher M., Wang J., Jin H., Agarwala A.: Subspace video stabilization. *TOG* (2011). 3

[LH96] Levoy M., Hanrahan P.: Light field rendering. *SIGGRAPH* (1996). 3

[Low04] Lowe D. G.: Distinctive image features from scale-invariant keypoints. *IJCV* (2004). 2, 3

[MWJ99] Murphy K., Weiss Y., Jordan M.: Loopy belief propagation for approximate inference: An empirical study. In *Proc. 15th conf. on Uncertainty in Artificial Intelligence* (1999), pp. 467–475. 2

[NLD01] Newcombe R. A., Lovegrove S. J., Davison A. J.: Dtam: Dense tracking and mapping in real-time. *ICCV* (2001). 2

[NNL13] Nguyen C., Niu Y., Liu F.: Direct manipulation video navigation in 3d. In *CHI* (2013). 1

[OLT06] Olson E., Leonard J., Teller S.: Fast iterative alignment of pose graphs with poor estimates. In *ICRA* (2006). 4

[OT06] Oliva A., Torralba A.: Building the gist of a scene: The role of global image features in recognition. *Progress in brain research* (2006). 3

[Pho12] Photosynth:. `www.photosynth.net`, 2012. 1, 2

[RAPLP05] Rav-Acha A., Pritch Y., Lischinski D., Peleg S.: Dynamosaics: Video mosaics with non-chronological time. In *CVPR* (2005), vol. 1, pp. 58–65. 3

[SEE*12] Sturm J., Engelhard N., Endres F., Burgard W., Cremers D.: A benchmark for the evaluation of rgb-d slam systems. In *Proc. Int. Conf. Intelligent Robot Systems* (2012). 7, 8

[SFC*11] Shotton J., Fitzgibbon A., Cook M., Sharp T., Finocchio M., Moore R., Kipman A., Blake A.: Real-Time Human Pose Recognition in Parts from a Single Depth Image. *CVPR* (2011). 3

[SGSS08] Snavely N., Garg R., Seitz S. M., Szeliski R.: Finding paths through the world's photos. *SIGGRAPH* (2008). 3

[SH99] Shum H., He L.: Rendering with concentric mosaics. In *SIGGRAPH* (1999). 3

[SS97] Szeliski R., Shum H.-Y.: Creating full view panoramic image mosaics and environment maps. SIGGRAPH. 2

[SSS06] Snavely N., Seitz S., Szeliski R.: Photo tourism: exploring photo collections in 3d. *SIGGRAPH* (2006). 1, 3

[Sze96] Szeliski R.: Video mosaics for virtual environments. *IEEE Computer Graphics and Applications 16* (1996), 22–30. 2

[Sze06] Szeliski R.: Image alignment and stitching: a tutorial. *Found. Trends. Comput. Graph. Vis.* (2006). 2, 6

[TDSL00] Tenenbaum J., De Silva V., Langford J.: A global geometric framework for nonlinear dimensionality reduction. *Science 290*, 5500 (2000), 2319–2323. 3

[TFBD01] Thrun S., Fox D., Burgard W., Dellaert F.: Robust monte carlo localization for mobile robots. *AI* (2001). 2

[TM07] Tuytelaars T., Mikolajczyk K.: Local invariant feature detectors: A survey. *Found. Trends. Comput. Graph. Vis.* (2007). 2

[WACS11] Wu C., Agarwal S., Curless B., Seitz S.: Multicore bundle adjustment. *CVPR* (2011). 6

[WMLS10] Wagner D., Mulloni A., Langlotz T., Schmalstieg D.: Real-time panoramic mapping and tracking on mobile phones. VR. 2

[Wu07] Wu C.: SiftGPU: A GPU implementation of scale invariant feature transform (SIFT). `cs.unc.edu/~ccwu/siftgpu`, 2007. 6

[YN01] You S., Neumann U.: Fusion of vision and gyro tracking for robust augmented reality registration. In *VR* (2001). 2