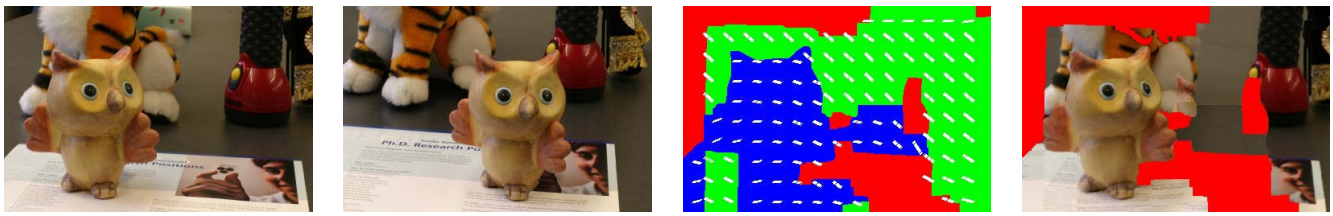
Piecewise Image Registration in the Presence of Multiple Large Motions

Pravin Bhat[1], Ke Colin Zheng[1], Noah Snavely[1], Aseem Agarwala[1],
Maneesh Agrawala[2], Michael F. Cohen[3], Brian Curless[1]
[1]University of Washington, Seattle, WA
[2]University of California, Berkeley, CA
[3]Microsoft Research, Redmond, WA



(a) Left image (b) Right image (c) Labeled flow field (d) Reconstructed image

Figure 1. Dense pixel correspondence between two images of a scene with moving rigid objects. Images (a) and (b) are the input, and (c) is the flow field from (a) to (b) with color labelings. Red indicates occluded areas, and blue and green indicate two motions modeled by fundamental matrices for the bird and the rest of the background(note that the flow vectors are uniformly scaled to a shorter length for visual clarity). Image (d) is the reconstruction of (a) based on the flow; red areas are occluded.

Abstract

We present a technique for computing a dense pixel correspondence between two images of a scene containing multiple large, rigid motions. We model each motion with either a homography (for planar objects) or a fundamental matrix. The various motions in the scene are first extracted by clustering an initial sparse set of correspondences between feature points; we then perform a multi-label graph cut optimization which assigns each pixel to an independent motion and computes its disparity with respect to that motion. We demonstrate our technique on several example scenes and compare our results with previous approaches.

1. Introduction

Many real-world scenes contain multiple objects that undergo independent motions. Two photographs of such a scene taken at different moments in time and different viewpoints will contain motions much larger than most optical flow techniques can handle. In this paper, we consider the problem of estimating a dense correspondence between two such perspective views of a scene containing multiple, independent objects that undergo large, rigid motions.

Our algorithm computes this correspondence in a two-stage process. First, each independent motion is discovered by randomly sampling a set of sparse feature matches. Then, each pixel in the reference view is assigned to a motion by optimizing a Markov random field formulation using graph cuts [5].

This two-stage process is very similar to the work of Wills *et al*. [19]; while they model motions using *homographies*, we model motions with both homographies and *fundamental matrices*. The advantage of using fundamental matrices is that the motion of an entire, rigid 3D object can be modeled with a single fundamental matrix. Homographies, on the other hand, can only model the motion of 3D planes and thus multiple homographies are required to describe the motion of a typical 3D object. In addition, our approach can recover information about the 3D shape of independently moving objects.

The disadvantage of using fundamental matrices is that, unlike other motion models (such as affine or planar perspective transformations) which explicitly provide a point-to-point correspondence, a fundamental matrix only describes the motion of each pixel up to an epipolar line. Thus, to compute a dense pixel correspondence using fundamental matrices as a motion model, we need to both assign each pixel to a motion *and* choose a translation along that pixel's epipolar line to obtain a complete point-to-point correspon-

dance. We use the term *disparity* to refer to this translation because our work bears a strong resemblance to estimating disparities for two-view stereo[13]. However, we handle multiple motions by allowing each pixel to choose a disparity along one of multiple epipolar lines. To handle planar objects, we also allow motions to be modeled with homographies and use a model selection heuristic to choose the appropriate motion model.

The main contribution of our work is to show that dense correspondence can be computed for scenes containing multiple large, rigid motions by assigning both a fundamental matrix and disparity to each pixel using Markov random field optimization. Since this optimization space is larger than both the work of Wills *et al*. [19] and graph cuts-based stereo algorithms [8], we also show that a solution can be computed efficiently by limiting the range of possible disparities using a good initialization technique and hierarchical estimation. To demonstrate the success of our approach we show our dense correspondence results and compare them to results computed using only homographies as a motion model. Finally, we demonstrate that we can reconstruct the 3D shape of individual objects in the scene by using the per-pixel disparities recovered by our approach.

2. Related Work

Techniques for computing dense pixel correspondence between two views can be categorized by the type of motion. In the case of a static scene, the epipolar constraint reduces the problem to two-view stereo[13]. Recent techniques for Markov random field optimization such as graph cuts [5] and belief propagation [15, 14] yield impressive solutions to the stereo problem, but cannot be directly applied to dynamic scenes, containing moving objects.

Techniques for dynamic scenes vary with the magnitude of motion. If motions are small, differential techniques based on the optical flow constraint are effective [10, 7]; extensions include discontinuous flow fields [4] and layer-based motion segmentation [18, 1]. Hierarchical coarse-to-fine estimation can increase the range of motion these techniques can handle [2], but only up to a point; very large motions require a different approach.

Techniques for computing dense, long-range correspondence in the presence of large motions are less common, and mostly use a feature-based registration approach. Torr [16] explores methods for selecting different motion models, including fundamental matrices and homographies, and proposes a robust metric for model selection. As already mentioned, Wills *et al*. [19] follow a similar approach to ours for computing dense correspondence between images with large inter-frame motion; they have also extended their approach to handle non-rigid motion[20] using splines to fit residual motion. In both cases, a planar projection motion model is used to estimate an initial set of candidate motions; we extend their approach by also using fundamental matrices with per-pixel disparities.

Multi-body structure-from-motion techniques (such as Vidal *et al*. [17]) estimate and segment rigid-body motions from the projections of 3D points in two images using a factorization approach. Our work uses similar inputs and motion models; however, our output and technical approach are very different. In particular, we produce a dense 2D motion field, while their work estimates 3D motions for a sparse set of feature points.

The rest of the paper proceeds as follows. In Section 3 we describe the piecewise object registration algorithm in detail, and in Section 4 we present the performance of our algorithm on a variety of examples. In Section 5 we conclude and suggest areas for future work.

3. Piecewise Object Registration

Given two $m \times n$ images, I_{left} and I_{right}, our goal is to match every pixel in I_{left} (where a pixel is an (x, y, r, g, b) tuple) to at most one pixel in I_{right}. We formulate this task as a labeling problem: each pixel $p \in I_{\text{left}}$ has a set of candidate labels $C(p)$ (corresponding to pixels in I_{right}), and a labeling L assigns a label, $L(p) \in C(p)$ to each pixel $p \in I_{\text{left}}$. An objective function, $E(L)$ measures the "quality" of a labeling. Typically, $E(L)$ includes a *data* term measuring the similarity of corresponding pixels and a *smoothness* term for encouraging nearby pixels to have similar labels (e.g., the corresponding pixels should also be nearby). The goal of the labeling problem is to find the minimum cost labeling.

In general, the complexity of solving a labeling problem depends on the number of pixels being matched, the number of candidate labels for each pixel, and the form of the cost function. For some cost functions, techniques such as belief propagation and graph cuts can efficiently produce approximate solutions. In our pixel labeling problem, if every pixel in I_{right} is a candidate label for each pixel in I_{left}, the number of labels is mn. Finding even an approximate solution with so many candidate labels can be prohibitively slow for all but the smallest images.

We assume that objects in the scene are piecewise rigid and therefore the motion associated with each rigid piece can be explained by a single 2D or a 3D motion model; namely a homography (for planar objects) or a fundamental matrix plus disparity.

Because we do not know *a priori* how many rigid objects are in the scene or how they are moving, we must first discover the dominant motions. To do so, we generate a sparse correspondence between the two images using automatic feature matching, then extract motions from the sparse correspondence using RANSAC. Each extracted motion generates one or more candidate labels for pixels in I_{left}—e.g., a fundamental matrix generates labels corresponding to the

pixels along an epipolar line. Finally, each pixel in I_{left} is assigned a single label using a graph cut optimization. We will now elaborate on the components of our registration algorithm.

3.1. Feature Detection and Matching

To extract motions between I_{left} and I_{right} we detect feature points in the two images using SIFT [9], Harris-Affine [12] and MSER [11] feature detectors. The local image information around the detected features is stored using the SIFT feature descriptor. We obtained implementations of these detectors and descriptors from the websites of the respective authors. Next, we match feature points in I_{left} to feature points in I_{right} using the ratio test proposed by Lowe [9], limiting matches to be between features found by the same detector. The ratio test has been found to reduce the number of false matches. For each feature point f in I_{left}, we find the two feature points, f_1 and f_2, of I_{right} whose descriptors have the smallest L_2 distance (d_1 and d_2, where $d_1 \leq d_2$) to f's descriptor. If the ratio $\frac{d_1}{d_2}$ is less than a user-defined threshold, the two feature points are added to the set of matches. For many real world images, the ratio test provides a robust measure for the quality of a feature match. The results in this paper were generated using a ratio threshold of 0.4. The result of this step is a set of matches M between feature points of I_{left} and I_{right}.

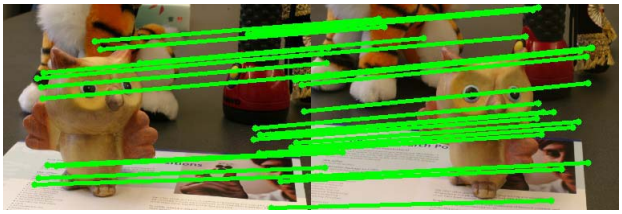
3.2. Fitting Motion Models

The next step is to discover the motions between the two images by clustering feature matches in M that can be explained by a single motion model. We use a fundamental matrix to model the 3D motion of objects in the scene. In the absence of 3D cues we fall back on homographies, since the motion of planar objects or the plane at infinity is better modeled using a single homography. Our motion-fitting algorithm is as follows:

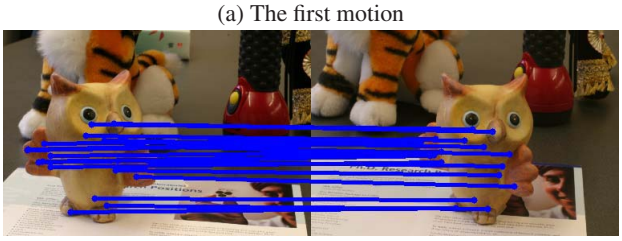
motions = empty-list
M = the set of all feature matches

while M not empty

1. Extract the dominant 3D motion by finding a fundamental matrix, F, in M using RANSAC [6]
2. Compute a 2D motion model by fitting a homography, H, to the matches explained by F
3. If H explains most of the matches explained by F then add H to *motions*, else add F
4. Delete the matches explained by F (or H) from M



(a) The first motion



(b) The second motion

Figure 2. Fitting motion models using feature matches. In this scene, the camera is translating and the bird figure moves. Our system detects both motions by fitting two fundamental matrices to the feature matches, one for the background (green), the other for the bird (blue).

In Step 3, we replace F with a homography if 95% of the matches explained by F can also be accurately explained by a homography. In Step 4, deleting the matches that have been explained by an extracted motion moves the algorithm towards termination.

An example scene where two motions are detected with this algorithm is shown in Figure 2.

3.3. Multi-Label Optimization

After finding a set of motions, the final step is to compute a dense correspondence between the pixels in I_{left} and I_{right}. We formulate this step as a multi-label optimization problem and solve for a local minimum using graph cuts.

Each motion, m_i, generates a set of candidate labels for each pixel, p, in I_{left}, where each label is a pair *(motion, disparity)*. The *motion* component of the pair identifies a motion model in the list of motions generated in Section 3.2. The *disparity* component of the pair specifies an offset from the disparity window center for that motion model. For the homography motion model the disparity is always 0. Thus, each motion classified as a homography generates a single candidate label for each pixel in I_{left}.

Each motion described by a fundamental matrix restricts correspondences for a pixel in I_{left} to lie along an epipolar line in I_{right}. The maximum number of pixels along the epipolar line is $\sqrt{m^2 + n^2}$ (the length of the diagonal of I_{right}). In order to efficiently find a good labeling, we restrict the number of candidates along this line to a disparity window of size k. To find the center of the disparity window for a given pixel p and fundamental matrix, we compute an initial estimate of the corresponding pixel in I_{right}, given

the fundamental matrix, as described in Section 3.3.1. We also compute the disparity window centers hierarchically, as described in Section 3.3.2, which allows us to increase the "virtual" size of the disparity window without introducing additional labels.

We also add a special label $\emptyset$ to the set of candidate labels for each pixel. The $\emptyset$ label will be assigned to pixels in I_{left} that we deem to be occluded in I_{right}, and which therefore correspond to no pixel.

Finally, we compute a labeling L, which assigns each pixel in I_{left} one of its candidate labels. We want our labeling to match pixels in I_{left} to pixels in I_{right} with similar color. At the same time, we would like the *motion* component of the labels to be piecewise-constant, and the *disparity* component to be piecewise-smooth. This reflects our assumptions that nearby pixels are likely to be projections of the same object and that most objects have a smooth shape.

We use the following cost function to express these desired properties of a labeling:

$$E(L) = E_{\text{data}}(L) + \lambda \cdot E_{\text{smooth}}(L) \tag{1}$$

The data term, E_{data}, measures how well the color of each pixel, p, in I_{left} matches the color of its corresponding pixel, p', in I_{right}:

$$E_{\text{data}}(L) = \sum_{p \in I_{\text{left}}} D(p, p'(p, L(p))) \tag{2}$$

where $p'(p, L(p))$ is the mapping of p to the right image given label $L(p)$ (we abbreviate this as simply p'), and $D(p, p')$ measures the distance between the RGB colors of p and p', unless p' is deemed to be occluded or falls outside the boundary of I_{right}. We define D as:

$$D(p, p') = \begin{cases} \gamma & \text{if } L(p) = \emptyset \\ d(I_{\text{left}}(p), I_{\text{right}}(p')) & \text{if } L(p) \neq \emptyset \end{cases} \tag{3}$$

where γ is a user-defined constant, and d is the Birchfield measure of color distance [3], which is more robust to pixel sampling effects than the L_2 norm.

E_{smooth} measures how well the labeling agrees with our smoothness assumption. E_{smooth} has the form:

$$E_{\text{smooth}} = \sum_{(p,q) \in N} V(L(p), L(q)) \tag{4}$$

where N is the set of 4-connected neighbors in I_{left}. The full form of V is defined as follows:

$$V(L(p), L(q)) = \begin{cases} \alpha & \text{if } L_m(p) \neq L_m(q) \\ \min(|L_d(p) - L_d(q)|, \beta) & \text{if } L_m(p) = L_m(q) \end{cases} \tag{5}$$

where $L_m(p)$ is the *motion* component of p's label, $L_d(p)$ is the *disparity* component of p's label, and α and β are user-defined constants. To ensure that our smoothness term is a metric we require $\alpha \geq \frac{\beta}{2}$. We define the constants as $\lambda = 0.1$, $\gamma = 0.4$, $\alpha = 10$, and $\beta = 10$ for all of our examples. We use the graph cut alpha-expansion algorithm of Boykov *et al.* [5] to find a labeling whose energy is within a constant factor of the global minimum.

3.3.1 Estimating the Window Center

Each fundamental matrix restricts the candidate matches of a pixel in I_{left} to pixels on an epipolar line in I_{right}. Ideally, we would treat every pixel on this line as a label in our optimization. However, to reduce the size of the problem, we limit the candidate correspondences generated by a fundamental matrix to a window centered around an initial guess for each pixel's match in I_{right}. To find the initial estimate, we first fit a similarity transformation to the feature matches used to generate the fundamental matrix. For each pixel $p \in I_{\text{left}}$, this transformation is applied to p's location, and the transformed location is projected onto p's epipolar line given the fundamental matrix. The location of the pixel in I_{right} closest to this projected location is used as the center of the disparity window for that pixel and motion model.

3.3.2 Hierarchical Estimation of Disparity Window Center

For very large motions in high-resolution images, our estimate of the window center may still be considerably far from the correct match, necessitating the use of large disparity windows. To increase the "virtual" extent of the disparity windows without adding more labels, we use a Gaussian image pyramid to hierarchically estimate the center of the disparity window. We run the graph cut optimization on each level of the pyramid, from coarsest to finest. The disparity window center for the coarsest level is chosen as in Section 3.3.1. For the rest of the levels we use the dense correspondence computed in the coarser levels to initialize the disparity window centers in the finer levels.

4. Results

In this section, we demonstrate our technique on four examples with varying amounts of motion. The elephant scene (Figure 3) is captured with a static camera and includes a single moving object. The bird scene (Figure 1) and the face scene (Figure 4) are captured with a moving camera and contain one moving object. Finally, the desktop scene (Figure 5) contains multiple moving objects and is shot with a moving camera. These scenes are challenging because they exhibit large motions, resulting in large

occluded areas, and contain textureless regions, such as the table surface, and specular objects, such the metal can.

In all our examples, we compute a pixel-to-pixel correspondence from the left image to the right image. For a 320x240 image pair the feature detection and matching takes 2 minutes to compute. The graph cut optimization runs for 10 to 20 minutes depending on the number of motion models in the scene, using two pyramid levels and a search window size of 40 pixels. To test our algorithm, we synthesize the left image using the dense correspondence field computed by our algorithm. We visualize the correspondences in a gridded flow field, where the background color indicates the motion labels (red pixels indicated occluded regions). For our scenes the flow vectors are very large, and thus hard to visualize clearly. To make our flow visualizations more comprehensible, we shorten the flow vectors using a uniform scale.

In Figure 3, we capture two frames of a moving elephant from the same view point. Our motion fitting algorithm automatically finds a homography to explain the background and a fundamental matrix to explain the motion of the elephant. In Figure 4, correspondences for the upper body of the person and the background regions are correctly computed; however, some regions are incorrectly classified as occluded, mostly due to regions of low texture information and the use of a small disparity window. Figure 5 demonstrates the ability of our algorithm to find correct correspondences even when the displacement is very large (in this case, over 30% of the image width).

Next we show an application of our algorithm. We recover the motions of independently moving objects; however, if we consider a single independent moving object, we can equivalently think of the object as static and the camera as moving. Thus, we have two views of each independent object, and can use the recovered disparity maps to infer the shape of the visible portions of such objects. Given the disparity map and the fundamental matrix for an object, we can create a projective reconstruction for that object. If the intrinsic parameters of the camera used to image the scene are known, we can upgrade the reconstruction to a Euclidean reconstruction. In Figure 6 we show a reconstruction of the ceramic bird created using the output of our algorithm and estimated intrinsic camera parameters.

Finally, we show two results computed using only homographies as a motion model, which is similar to the approach of Wills *et al*. [19] (Figure 7). Notice that multiple homographies are required to describe the motion of a single 3D object, while our results can describe an object with a single fundamental matrix and provide depth information.

5. Conclusion and future work

In this paper we describe a framework for computing dense, pixel-to-pixel correspondence between two images containing multiple objects that move independently. Unlike previous techniques, our approach can handle large motions of both 3D objects and 3D planes. In addition, we can recover depth information for every independently moving object in the scene. We are able to achieve high quality results for many scenes because we use fundamental matrices as well as homographies to model motion. However, we found that for some scenes, such as those containing mostly planar objects, using homographies alone as a motion model can achieve better results, because they model planar scenes with fewer degrees of freedom.

There are many avenues for future work. Our framework relies heavily on feature detectors such as SIFT [9] to establish a sparse correspondence between two images. The density of this sparse correspondence is crucial for extracting every independent motion in the scene; thus, our algorithm fails to produce good results for low resolution images. We also cannot handle non-rigid objects or piecewise-rigid motion. In future work, we would like to make our framework robust to these phenomena.

Large occluded or unoccluded regions are also difficult for our algorithm to handle. Our framework can fail to accurately identify such regions, especially when the occluded/exposed regions are visually similar to other parts of the scene. In such cases the matching is inherently ambiguous, because exposed regions in the left image could be explained by a phantom motion of the exposed regions in the right image, and vice versa.

Acknowledgements

This work was supported by the University of Washington Animation Research Labs, the Washington Research Foundation, Adobe, a National Science Foundation Graduate Research Fellowship, and an industrial gift from Microsoft Research.

References

[1] S. Ayer and H. Sawhney. Layered representation of motion video using robust maximum-likelihood estimation of mixture models and mdl encoding. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, pages 777–784, 1995. 2

[2] J. R. Bergen, P. Anandan, K. J. Hanna, and R. Hingorani. Hierarchical model–based motion estimation. In *European Conference on Computer Vision (ECCV)*, pages 237–252, 1992. 2

[3] S. Birchfield and C. Tomasi. A pixel dissimilarity measure that is insensitive to image sampling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(4):401–406, 1998. 4

[4] M. Black and P. Anandan. The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields. In *Computer Vision and Image Understanding (CVIU)*, volume 63, pages 75–104, 1996. 2

[5] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11):1222–1239, 2001. 1, 2, 4

[6] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981. 3

[7] B. Horn and B. Schunck. Determining optical flow. In *Artificial Intelligence*, volume 17, pages 185–204, 1981. 2

[8] V. Kolmogorov and R. Zabih. Multi-camera scene reconstruction via graph cuts. In *Proceedings of the 7th European Conference on Computer Vision (ECCV)*, pages 82–96, 2002. 2

[9] D. Lowe. Distinctive image features from scale invariant keypoints. *International Journal of Computer Vision*, 2004. 3, 5

[10] B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of DARPA Image Understanding Workshop*, pages 121–130, 1981. 2

[11] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In P. L. Rosin and D. Marshall, editors, *Proceedings of the British Machine Vision Conference*, volume 1, pages 384–393, London, UK, September 2002. BMVA. 3

[12] K. Mikolajczyk and C. Schmid. Scale and affine invariant interest point detectors. *International Journal of Computer Vision*, 60(1):63–86, 2004. 3

[13] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision (IJCV)*, 47(1-3):7–42, 2002. 2

[14] J. Sun, Y. Li, and S. B. Kang. Symmetric stereo matching for occlusion handling. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 399–406, 2005. 2

[15] J. Sun, N.-N. Zheng, and H.-Y. Shum. Stereo matching using belief propagation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(7):787–800, 2003. 2

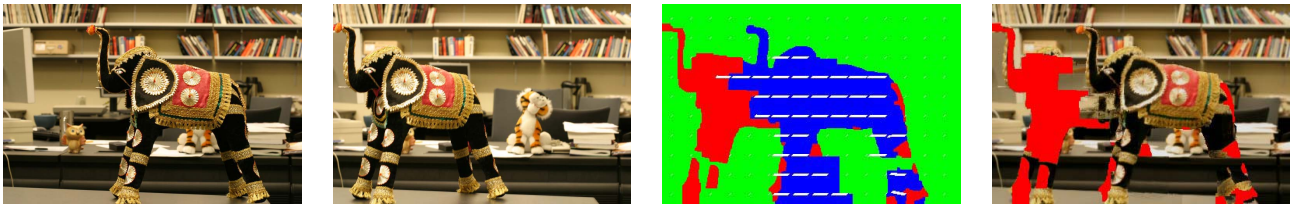
[16] P. Torr. An assessment of information criteria for motion model selection. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 47–52, 1997. 2

[17] R. Vidal, Y. Ma, S. Soatto, and S. Sastry. Two-view multibody structure from motion. *International Journal of Computer Vision (IJCV)*, 2005. 2

[18] J. Wang and E. Adelson. Representing moving images with layers. *IEEE Transactions on Image Processing*, 3(5):625–638, 1994. 2

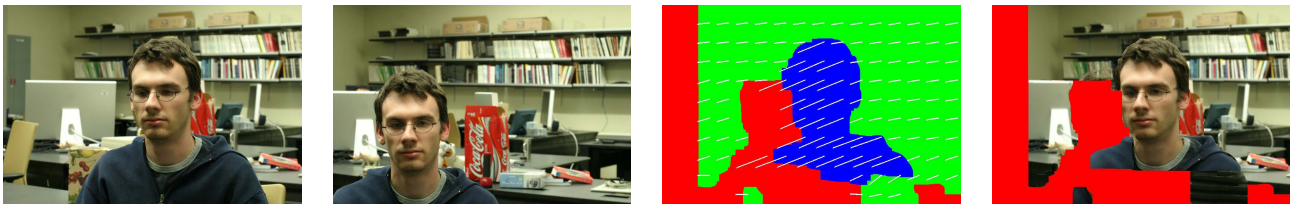
[19] J. Wills, S. Agarwal, and S. Belongie. What went where. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 37–44, 2003. 1, 2, 5

[20] J. Wills and S. Belongie. A feature-based approach for determining long range optical flow. In *Proceedings of the 8th European Conference on Computer Vision (ECCV)*, Prague, Czech Republic, May 2004. 2



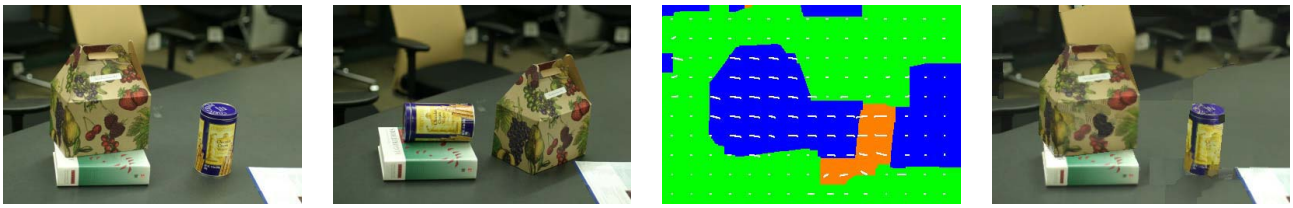
(a) Left image (b) Right image (c) Flow visualization (d) Reconstructed image

Figure 3. Elephant scene and results. The camera is static and the elephant moves.



(a) Left image (b) Right image (c) Flow visualization (d) Reconstructed image

Figure 4. Face scene and results. Both the camera and the person move.



(a) Left image (b) Right image (c) Flow visualization (d) Reconstructed image

Figure 5. Desktop scene and results. The camera, the box and the can move.

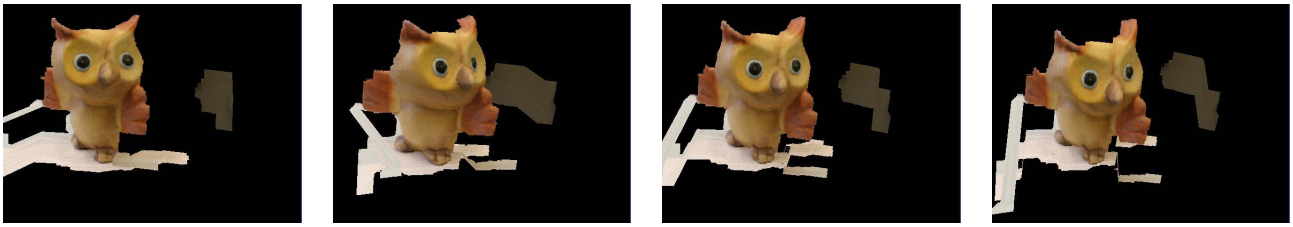


Figure 6. 3D reconstruction of the bird from dense correspondence. The supplementary video shows the object rotating continuously.

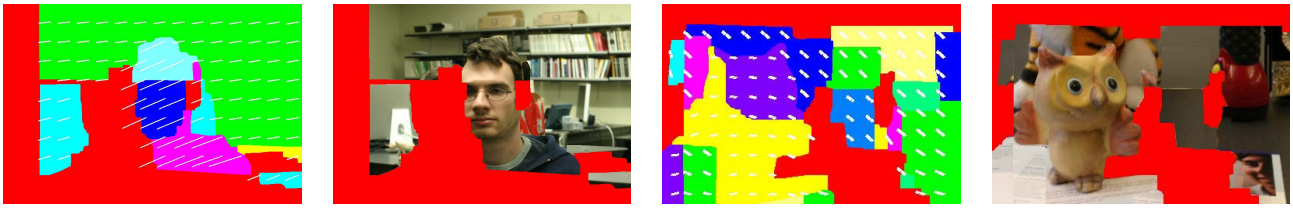


Figure 7. Results for two scenes computed using only homographies as the motion model; compare these results to those in Figures 4 and 1 that are computed using both homographies and fundamental matrices with per-pixel disparities.