An empirical comparison of methods for image-based motion estimation

Henele Adams, Sanjiv Singh, and Dennis Strelow
Carnegie Mellon University
hia@andrew.cmu.edu,{ssingh,dstrelow}@cs.cmu.edu

Abstract

This paper presents a comparison between methods that estimate motion of a camera from a sequence of video images. We have implemented two methods– a homography based method that assumes planar environments, and shape-from-motion, a general method that can deal with a fully three dimensional world. Both methods have been formulated in an iterative, online form to produce estimates of camera motion. We discuss a trade-off in accuracy and run time efficiency based on experimental results for these two general methods in relation to ground truth. We show how a variation of the homography method can produce accurate results in some cases when the environment is non-planar, with low computational cost.

1. Introduction

Estimation of camera motion from a sequence of images is attractive because it provides a potentially simple method using a solid state sensor. Motion estimation based on imagery can be used effectively for robot navigation especially when it is combined with other modalities such as odometery and inertial measurement. Since cameras are projective devices that map the three dimensional world onto an image plane, it is intuitive that estimation of motion using a sequence of images should consider the three dimensional nature of the environment. However, the shape of the environment is generally unknown and hence we would like a method that proceeds without such explicit *a priori* knowledge.

One well known class of techniques for visually based camera motion estimation utilizes *homographies*. A homography is a 3×3 matrix capable of describing the projective transformation of points viewed on a plane between images. Here the visual features are assumed to lie in a plane or a set of planes that can be determined heuristically by a separate step [4]. The general idea is that features tracked from one frame to the next are used to compute the displacement between two camera locations. These displacements can be integrated to estimate camera motion. In some cases, it is not possible or feasible to separately determine planar patches in the scene and we would like to estimate differential motion directly without prior knowledge of the scene. Another class of techniques is called shape-from-motion(SFM) which can deal with a three dimensional structure in the world by simultaneously estimating camera motion and scene structure [20]. Generally speaking, the former set of methods have low computational complexity because they benefit from constraints on the operating environment. Conversely, the latter methods have higher complexity because of a more general representation of the environment.

The purpose of this paper is to compare these methods in terms of accuracy and computational complexity in various environments that vary from completely planar to those with unknown high curvature. In addition, we would like to consider environments that are planar but have sparse, although significant excursion from the plane. A motivating scenario for this work is estimation of motion from small autonomous aircraft which must by necessity be computationally efficient.

We have implemented online versions of the two methods mentioned. That is, estimates of motion are available as images are captured. The SFM method uses an extended Kalman filter while the homography method operates in a memoryless frame to frame fashion. In addition, we have implemented a variation to the homography based method that uses the best of a number of random trials to estimate camera motion at every frame. We discuss trade-offs using simulation and image sequences taken from a "flying" camera for which we have accurate ground truth. In addition to comparing the iterative methods with the ground truth we also show how the results compare to a batch implementation of SFM.

2. Related Work

In the past few years homography-based methods have become widely used for the estimation of autonomous vehicle motion. Recent work by Garcia, et al. [6] and Lots, et al. [10], for example, utilizes homographies to estimate

the motion of underwater vehicles for mosaicking and station keeping. The estimation of homographies between two images is discussed by Hartley and Zisserman[7], who consider linear (algebraic error), "gold standard" (geometric error), and robust techniques for the estimation, as well as methods for recovering metric properties of the motion and imaged scene from the estimated homography. Our own algorithms for estimating the homography, and for converting the estimated homography to a camera motion, are similar to those described by Faugeras and Lustman[4].

Homography-based motion estimation is most applicable when the imaged scene is at least approximately planar. Algorithms for shape-from-motion estimate both camera motion and general three-dimensional scene structure from a video sequence. So, these algorithms are naturally robust to deviations from, but do not exploit a priori knowledge of, planar scene scene structure. Shape-from-motion algorithms can be categorized as either online or batch methods. Online methods, such as Broida, et al.[3] and Azarbayejani and Pentland[1], use a Kalman filter to refine the motion and shape estimates as each image arrives. Batch methods, such as Szeliski and Kang[20], typically apply general nonlinear minimization methods to all observations simultaneously, producing optimal estimates for the shape and motion. The time complexity of online methods is linear in the number of images, whereas batch methods are typically cubic in the number of images. We have implemented both online and a batch shape-from-motion algorithms, which we describe in the subsequent sections.

To provide some perspective, we also mention a few other paradigms for motion estimation that we have excluded from our comparison. It is possible to compute six-degree-of-freedom egomotion up to a scale in translation from the dense, two-dimensional optical flow between two images [8][15]. Multilinear techniques such as fundamental matrix and trilinear tensor estimation make it possible to compute motion from sparse features without making any assumptions about or simultaneously computing structure [9][17]. However, both dense flow and multilinear techniques are limited to only a few images and are unlikely to result in accurate motion estimates when integrating over the long sequences we expect to encounter in autonomous vehicle navigation. Factorization methods [21][14] for shape-from-motion are highly efficient and deal with multiple images, but require that every feature appear in every image, which limits their usefulness for autonomous vehicles.

3. Method

We use a two-stage approach with our SFM and homography based methods. In the first stage (common to both methods) sparse, two-dimensional point features are extracted and tracked though the image sequence. In the second stage image feature locations are processed to provide estimates of camera motion.

In our experiments we have used the Lucas-Kanade tracker[11][2][12], which has shown good performance in comparative studies. Lucas-Kanade is an iterative method that tracks to subpixel resolution. Like all trackers, the method has difficulties when tracked points become occluded or change appearance. Promising features are extracted by enumerating the goodness of all features in the image and picking the best *n* features. Since camera motion causes features to move outside the field of view, it is necessary to find new features continually to keep the distribution of tracked features uniform throughout the image. This is done by dividing the image into a 3 by 3 grid of equally sized regions and extracting new features whenever the number of features within a region falls below a predefined minimum. Newly extracted features within a thresholded image distance of existing features are discarded.

3.1 Shape-from-motion overview

For the second stage of the SFM solution, the resulting feature tracks are used to recover the camera's motion and the three-dimensional locations of the tracked points by minimizing the error between the tracked point locations and the image locations predicted by the shape and motion estimates. Because we recover a three-parameter rotation and a three-dimensional translation for each image, and a three-dimensional position for each point, the total number of estimated parameters is $6f + 3p$, where f is the number of images and p is the number of points.

SFM recovers shape and motion estimates only up to a scaled rigid transformation. That is, applying the same scaling and rigid transformation to all of the camera and point estimates produces new estimates that explain the observed data as well as the originals. In many applications it is not necessary to remove this ambiguity, but for autonomous navigation, this ambiguity must be removed by augmenting shape-from-motion with appropriate additional sensors. We will address this issue in future work.

The estimation process is done either as a batch method or as an online iterative implementation.

Batch shape-from-motion. Our batch shape-from-motion algorithm, based on previous work [18], uses Levenberg-Marquardt to minimize the error between observed projections (i.e., feature tracks) and predicted projections with respect to the camera and point estimates. We refer the reader to [16] for the details of Levenberg-Marquardt, which is widely used.

To guarantee convergence, the initial estimate provided to the method must be sufficiently close to the solution. We currently use the estimate provided by the online method, described in the next section, as the initial estimate to the batch method.

Online shape-from-motion. Our online shape-from-motion method uses an extended Kalman filter to estimate shape and motion and their covariances. Our method is based on previous work [18] but uses a conventional camera. The method is similar to that described in [3]. See [13] for details on the Kalman filter in general, or [3] and [1] for more detailed information on Kalman filtering for conventional shape-from-motion.

3.2 Homography-based Motion Estimation

When implementing the second stage as a homography solution, the tracked points are used to generate frame to frame homographies from which a frame to frame motion estimate may be derived. Each frame to frame camera motion estimate is integrated to derive the current position estimate. The homography matrix is calculated using singular value decomposition(SVD) from equations derived from the point correspondences as in [4]. A camera motion estimate is then derived from the homography as per [19]. Our current implementation assumes there is only one dominant plane in the scene.

A homography is a 3×3 matrix capable of describing the projective transformation of points viewed on a plane between images. To compute the homography a minimum of four point (no three of which may be colinear) correspondences on the viewed plane are required. In general, more are used to make the homography generation system more robust to tracking errors.

For additional robustness against local planar excursions and tracking errors a random sampling and consensus method(RANSAC)[5] may be applied to the homography calculation. In this case the homography calculation is attempted multiple times. Each trial starts with a homography calculation using four randomly selected points from the set of available feature track points. As every tracked feature has an initial coordinate in the old image and a current coordinate in the current image, the homography may be checked by applying the homography to the initial coordinates and noting the difference between the transformed coordinate and the current coordinate. For every transformed point in agreement (to within an error threshold) the homography gets a vote. If the number of votes exceeds another threshold, which may be proportional or absolute, the homography is accepted and all the point correspondences in agreement are used to calculate the final resultant homography using SVD. If the voting threshold is not exceeded another trial occurs with a new randomly selected set of point correspondences, repeating the process. To limit execution time an upper limit on the number of trials is used. The thresholds discussed above may be calculated according to [5].

4. Results

Here we present results from simulation and from lab experiments. We have simulated two types of terrain with increasing deviations from the plane as discussed above. Simulated motion in this terrain is used to generate synthetic features and three algorithms are used to recover the motion. We also report on laboratory experiments in which a camera mounted on a crane was moved along a specified path above a terrain. Motion recovered by the three online algorithms and a batch method are compared to the ground truth.

4.1 Synthetic

Synthetic image features were generated to experimentally test the robustness of the homography based methods. Synthetic runs assume a downward pointing camera viewing a single plane. Tracking is unnecessary as the relation between all points in the plane and the projected image are known mathematically. In the two domains of interest the 3D model of the plane under view is gradually degraded in two different ways: globally and locally.

The globally degraded domain distorts the viewed ideal plane into an increasingly less planar object over its entire surface. The globally deformed 3D surface is described by:

$$z = k * sin(x/\pi) * sin(y/\pi)$$

Local deformations are modeled as spikes jutting out of the viewed plane. More specifically, $z = 0$ when $\sin(x/\pi) * \sin(y/\pi) < 0.5$ or $k * \sin(x/\pi) * \sin(y/\pi)$ otherwise.

This description yields a mostly planar viewing object, with 2D real estate taken up by any distortion being independent of the level of severity.

For both domains the severity, k, of the introduced distortion is calculated as a proportion of the camera to plane viewing distance. Experiments are done using these proportions: 0.00, 0.05, 0.10, 0.15, 0.20, and 0.25.

For each frame, tracked features are calculated by projecting the viewable points in the point field (generated using one of the two surface models described) onto the viewing plane of a virtual camera. Field of view of the camera is 90 degree, has no distortion, is located at 10 distance units from the plane with an optical axis parallel with the Z axis and points at the viewed plane. Excess points from

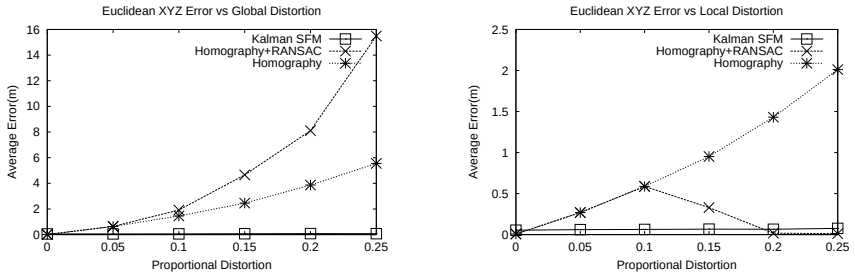

Figure 1. Average error under two deformation models. (left) global variation, (right) local variation

the 3D point field outside the bounded image projection were clipped and not recorded as tracking information for the frame. The camera undergoes translations in the following frames in a spiral pattern while keeping Z constant. For each sequence 967 frames of tracking information were generated.

Average euclidean errors between the ground truth and the motion estimate for each level of severity, k, and each plane model is presented in figure 1. The local deformation results show the effectiveness of the RANSAC addition; as points violate the planar model with increasing k they are rejected with increasing frequency leading to the better motion estimate results. In the global deformation cases RANSAC becomes a liability. The plane derived from using all of the points in the non-RANSAC version remains consistent from frame to frame while the RANSAC variant subsamples tracked points resulting in a larger subspace of possible planes leading to less constraints on the plane in the next frame and on the resulting motion estimate.

4.2 Robocrane

The Robocrane is a fifteen-foot high robotic crane (more specifically, an inverted Stewart platform) built by the National Institute of Standards and Technology (NIST). The Robocrane features a large triangular platform whose six degree of freedom position within the $10 \times 10 \times 10$ foot workspace can be specified by computer. Because the Robocrane's platform can be moved to known positions in such a large space above the ground, it is ideal for creating simulated air vehicle motions with ground truth.

For this experiment, a downward pointing black and white analog camera (fov = 60°) was mounted on the moving platform within the Robocrane. A tarp was placed on the concrete floor below the camera in a fairly flat manner. Rocks of a variety of sizes (all less than 30 cm tall) were placed atop the prepared surface. Finally, additional texture in the form of a small amount of pebbles and dirt were distributed over some of the area.

Two PCs were used to collect data and control the Robocrane. One PC was dedicated to the acquisition of time stamped images while the second controlled the Robocrane and logged the ground truth of the camera at 5 Hz. The path specified for the robocrane was specified in 10 cm increments and starts with a downward Z motion, not quite along the Z axis, to a height approximately 1 m from the floor then continues in a spiral pattern at constant height.

Feature tracking information was generated using a feature extractor and Lucas-Kanade tracker discussed previously. This one set of tracking data was fed into the four methods up for comparison: homography, homography + RANSAC, Kalman SFM, and batch SFM. All the methods used 498 frames of tracking data except the batch SFM method which was downsampled by three, leaving 164 frames, due to computational limitations.

Resulting paths of each motion estimation method were then fitted to the crane ground truth path using the best scaled rigid transformation minimizing the error between the two. XY results of the four estimators are shown in figure 4 along with a XYZ Euclidean error histogram in figure 2.

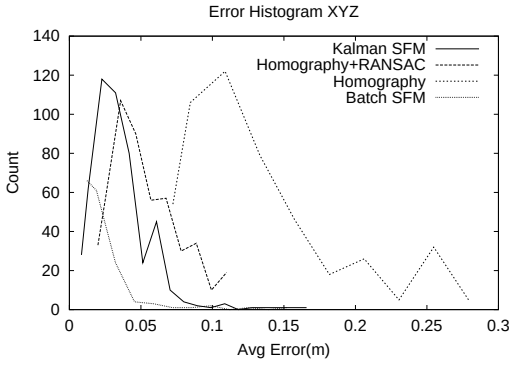

Figure 2. Euclidean error histogram of estimated motion vs. ground truth

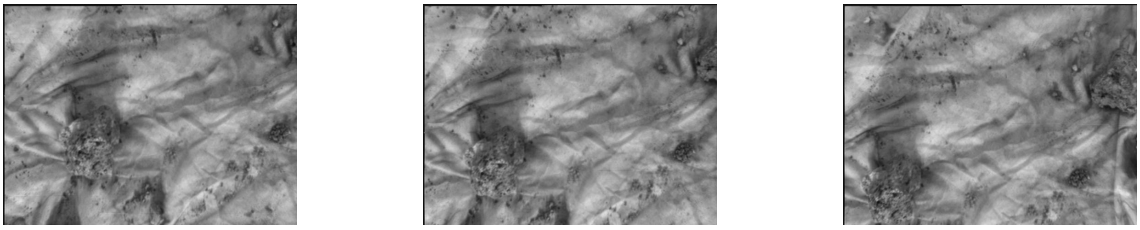

Figure 3. Three Images from the Robocrane sequence.

4.3 Computation

The platform used to compute the motion estimates was a Pentium III running at 800 MHz. Total time required to generate each of the motion estimates for the Robocrane experimental sequence are summarized in table 1 and does not include time required for feature tracking. Our feature tracker uses approximately 20 ms per image when tracking 80 features. Two cases are given for the random sampling modification; the worst occurs when the number of trials in the homography calculation is allowed to reach the maximum allotment per frame(25). Typically, consensus may be reached within 5-7 iterations.

Table 1: *Average running time per frame for Robocrane experiment motion estimates.*

Method	msec per Frame
Homography	28
Homography+RANSAC Typical	157
Homography+RANSAC Worst	588
Kalman SFM	1050
Batch SFM	$\sim$44000

5. Conclusions

We have implemented two methods of estimating camera motion from a sequence of video imagery based on vastly different paradigms and have tested these within varying terrain conditions. One method, based on homography, assumes that all feature points lie in a plane while the other is able to deal with a general 3D world. This paper has compared the trade-offs between these methods in terms of accuracy and computational efficiency. The simple homography-based method works fast and accurately when the planar assumption is valid. Accuracy degrades as the excursions from the plane become significant. We find that the other method based on an online SFM implementation is able to deal with both types of variations from the plane and estimates motion with low error. However, the SFM method is relatively computationally expensive. In some cases, when the terrain is mostly planar but has sparse, although non-trivial excursions (as would be posed for example, by randomly placed trees observed by an aircraft) a variation of the homography method that uses RANSAC can provide significantly improved performance. This variation can provide accuracy that is approximately equal to online SFM but with lower computational cost. We also note that the motion estimates produced from the online methods are significantly lower in computational cost to batch SFM but without a proportional degradation in accuracy. In future work we seek to improve the implementations for both online methods to operate at a higher throughput. We will also integrate the estimates from vision based motion estimation with other sensor modalities such as odometry and inertial measurement.

Acknowledgements

The authors acknowledge Jeffrey Mishler for assisting with the experiments on the Robocrane.

References

[1] A. Azarbayejani and A. Pentland. Recursive estimation of motion, structure, and focal length. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(6):562–575, June 1995.

[2] S. Birchfield. `http://vision.stanford.edu/~birch/klt/`.

[3] T. J. Broida, S. Chandrashekhar, and R. Chellappa. Recursive 3-D motion estimation from a monocular image sequence. *IEEE Transactions on Aerospace and Electronic Systems*, 26(4):639–656, July 1990.

[4] O. Faugeras and F. Lustman. Motion and structure from motion in a piecewise planar environment. *Intl. Journal of Pattern Recognition and Artificial Intelligence*, 2(3):485–508, 1988.

[5] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to im-

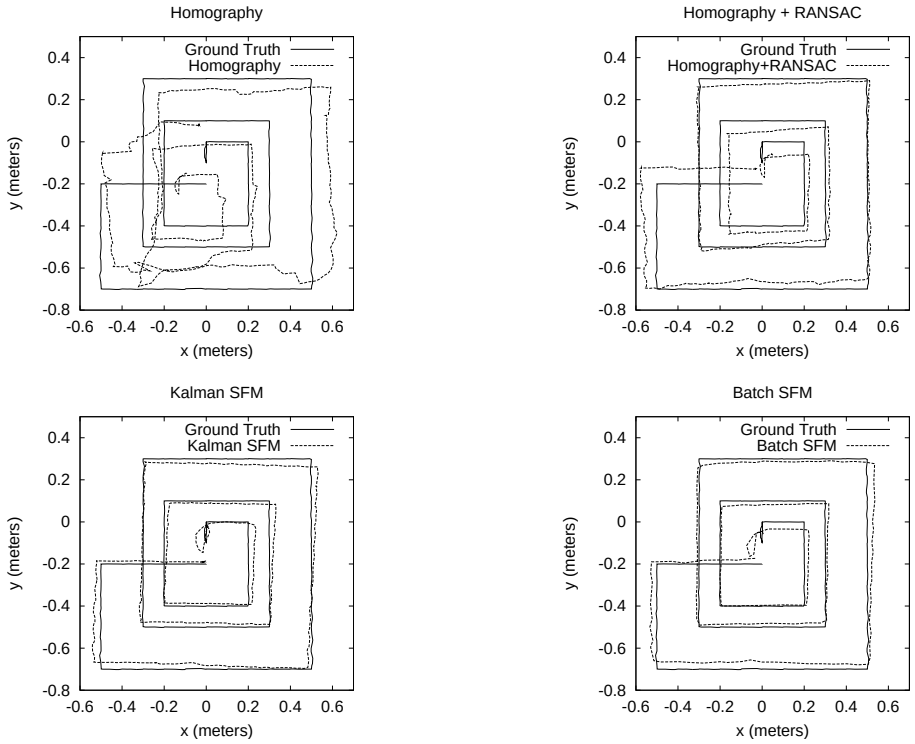

Figure 4. Crane ground truth and estimated motion for the crane experiment. (Top left) homography based method, (Top right) homography + RANSAC based method, (Bottom left) On-line SFM, (Bottom right) batch SFM

age analysis and automated cartography. *Comm. of the ACM*, 24(6):381–395, June 1981.

[6] R. Garcia, J. Batlle, and X. Cufi. Positioning an underwater vehicle through image mosaicking. In *Proceedings of the 2001 IEEE International Conference on Robotics and Automation*, 2001.

[7] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2000.

[8] B. K. Horn and B. G. Schunck. Determining optical flow. *Artificial intelligence*, 17:185–203, 1981.

[9] H. C. Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Nature*, 293:133–135, September 1981.

[10] J.-F. Lots, D. Lane, E. Trucco, and F. Chaumette. A 2-D visual servoing for underwater vehicle station keeping. In *Proceedings of the 2001 IEEE International Conference on Robotics and Automation*, 2001.

[11] B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the Seventh International Joint Conference on Artificial Intelligence*, 1981.

[12] Intel Corporation. Open source computer vision library. `http://www.intel.com/research/mrl/research/cvlib/`.

[13] The Analytic Sciences Corporation. *Applied Optimal Estimation*. MIT Press, Cambridge, Massachusetts, 1974.

[14] C. J. Poelman and T. Kanade. A paraperspective factorization method for shape and motion recovery. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(3), March 1997.

[15] K. Prazdny. Egomotion and relative depth map from optical flow. *Biological cybernetics*, 36:87–102, 1980.

[16] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, Cambridge, 1992.

[17] A. Shashua. Algebraic functions for recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(8):779–789, 1995.

[18] D. Strelow, S. Singh, J. Mishler, and H. Herman. Extending shape-from-motion to noncentral omnidirectional cameras. In *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, 2001.

[19] P. Sturm. Algorithms for plane-based pose estimation. In *IEEE Intl. Conf. on Computer Vision and Pattern Recognition*, 2000.

[20] R. Szeliski and S. B. Kang. Recovering 3D shape and motion from image streams using nonlinear least squares. *Journal of Visual Communication and Image Representation*, 5(1):10–28, March 1994.

[21] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: a factorization method. *International Journal of Computer Vision*, 9(2):137–154, 1992.