Accurate Efficient Mosaicking for Wide Area Aerial Surveillance

Jan Prokaj and Gérard Medioni
University of Southern California
Los Angeles, CA 90089
{prokaj|medioni}@usc.edu

Abstract

Wide Area Aerial Surveillance (WAAS) imagery is captured by an array of smaller sensors sharing an optical center, instead of one large sensor. It is desirable to generate a single image (mosaic) from the sensor array, since it simplifies higher level vision tasks. It is important that the mosaic be of high quality, without noticeable seams, and be estimated efficiently for every frame of the video.

We propose a piecewise affine model to handle image distortions not captured by a homography. This model has more degrees of freedom than a standard lens distortion model used in previous work, yet can be estimated just as efficiently by carefully selecting a small number of constraints in the optimization of model parameters. We have evaluated our algorithm on publicly available WAAS imagery and the results show our proposed model produces more accurate mosaics than the widely used lens distortion model, thus simplifying further stages of image analysis.

1. Introduction

The development of unmanned aerial vehicles (UAV), advances in optics and digital photography, and increased computational power has led to the growth of a particular kind of imagery, called Wide Area Aerial Surveillance imagery (WAAS) [2, 3, 4]. This imagery is characterized by its generation on aerial platforms, low temporal sampling rate, spatial resolution of 0.5 m/pixel or better, and large format (tens of megapixels). Furthermore, physical constraints require that the imagery is captured by an array of smaller sensors sharing an optical center, rather than one large sensor, as is illustrated in Figure 1. All of these characteristics have non-trivial implications for computer vision algorithms, such as stabilization, target tracking, and 3D reconstruction.

In this paper we consider the mosaicking problem, that is the generation of a single image from the sensor array, as if it were captured by a single (virtual) camera. Having access to such image has at least two benefits: it makes it easy to see the extent of the geographic area being captured and thus simplifies UAV control, it avoids the problem of tracking moving objects across several cameras, which adds algorithmic complexity. High quality estimation is a primary requirement. If it is inaccurate (has visible seams), errors will result in algorithms that use this image is input. For example, image stabilization will not be possible and there will be many false moving object detections around the seams.

The estimation of the mosaic must also be efficient. Even though the cameras in the sensor array are fixed, there are slight, but noticeable, changes in the sensor geometry at every frame. These may be caused by vibration or imperfections in mechanical mounting. In any case, it means that we cannot rely on a one-time, offline, estimate of the mosaic, but must re-estimate it online for every frame.

The primary contribution of this work is a mosaicking approach that satisfies both of these requirements. It is able to generate high-quality mosaics by allowing a non-rigid deformation of the image, and it is able to estimate this deformation efficiently by carefully selecting a small number of optimization constraints. The results show that our piecewise affine model has better accuracy compared to the classic lens distortion model used in previous work [22, 20, 19]. Furthermore, we show that we can estimate our model in a similar amount of time as a model with a smaller number of degrees of freedom. We have performed our evaluation on one of the publicly available WAAS datasets [2].

Before describing our approach in detail in section 3, we review related work in the next section. Evaluation of our method is presented in section 5, while section 6 concludes and suggests remaining challenges for the future.

2. Related Work

The interest in mosaic computation has a rich history. Some of the early applications include whiteboard or document scanning [23], video enhancement [18], video compression [11], and video indexing [10]. In all these applications, what is being mosaicked is a sequence of images taken from the same camera, which is in contrast to the na-

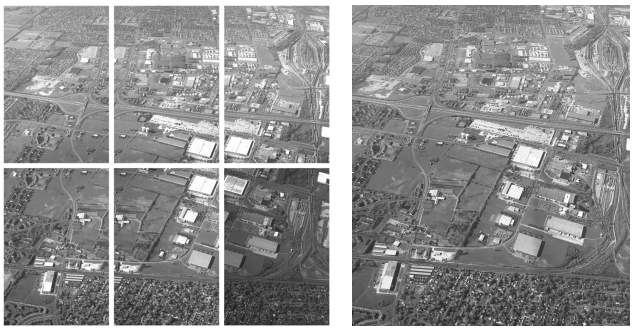


Figure 1. One frame of the imagery is captured by an array of cameras (left), while it is desirable to work with only one image per frame, as if it were captured by a virtual camera (right).

ture of mosaicking done here, where the images to be mosaicked come from different sensors. This makes the mosaic estimation problem more difficult since different sensors potentially have significantly different camera calibrations and image response functions.

One of the major problems in mosaicking is global alignment of a set of images. Since geometry (usually a homography) is estimated between pairs of images, to generate a mosaic from more than two images, several homographies may need to be concatenated (multiplied) to estimate a correct transformation from one image to the reference image. Each time two homographies are multiplied, a small error is introduced. After a few multiplications, the accumulated error is large enough to cause misalignment. This problem is addressed by [12, 21, 19]. The common solution is to optimize all homographies of the mosaic at once, with a prior topology calculation step if necessary [24]. In our application, the number and arrangement of images in the mosaic is known, therefore topology determination is not necessary. However, a joint optimization of the geometry is still needed.

Deviations from the pinhole camera model in the context of mosaic construction are discussed in [22, 20, 21]. In [22, 20], a lens distortion model is adopted to handle all residual error. Shum and Szeliski [21] propose a local alignment step based on optical flow that handles arbitrary camera model. However, this introduces too many degrees of freedom and the estimation of this model is more likely to get stuck in local minima.

A similar problem to mosaicking is video stabilization. The goal there is to take an existing jittery image sequence (usually from a hand-held camera) and use it to synthesize a new sequence without any jittering. Warping the input images to their stabilized configuration without any artifacts requires a non-rigid deformation of the input. Recent methods in this area [9, 15] have used a similar image deformation model as in this work, called as-rigid-as-possible deformation. However, estimating such deformation requires precise, uniformly distributed, correspondences between images. In our experience, the current state-of-art feature detectors and descriptors do not produce such correspondences.

Another approach to mosaicking was recently proposed by Lin *et al*. [14]. There, an affine stitching field is estimated, similar to our piecewise affine model. Instead of using point correspondences directly to estimate the stitching field, which causes problems as mentioned above, the authors formulate the problem as finding a smooth stitching field that minimizes the difference between the SIFT descriptors of the resulting (implied) correspondences. While the stitching results are state-of-art, the flexibility of the algorithm to fit generalized motion comes at a computational cost. A MATLAB implementation is able to stitch a pair of 500x500 images in 8 minutes. In our domain, we are interested in stitching 6 images, each 4008x2672 in resolution, which is not tractable by their approach.

3. Approach

As stated earlier, our goal is to generate a high quality mosaic that is suitable for higher-level computer vision tasks, such as stabilization, tracking, and 3D reconstruction. To facilitate the generation of this image, the sensors in the array are arranged so that they all share an optical center (or center of projection). Given the long distance from the scene, a small translation between the centers of projection can be discounted. In other words, the 3D transformation from the pose of one sensor in the array to another is a rotation. This helps, because it is well known [8] that if there is no translation between two cameras, the image transformation between them is fully described by a homography

$$H = K^{'} R K^{-1} \,, \tag{1}$$

where R is a 3D rotation matrix, K' is the calibration matrix of the destination camera, and K is the calibration matrix of the source camera. By selecting one of the images in the array as a reference, and estimating homographies between it and the rest of the images in the array, all images can be "registered", or aligned to a common coordinate frame.

However, this model is only valid for a true pinhole cameras, which is not physically practical. Additional factors such as lens distortion corrupt this model. Therefore, a model with more degrees of freedom is necessary to accurately register the images. A common solution to this is to adopt a global radial distortion model [8]. We have observed that while this model improves the registration accuracy, it does not accommodate all of the deviations from the pinhole model and an even more flexible, more local model is needed. We propose a piecewise affine model to solve this problem.

3.1. Piecewise Affine Model (PAM)

The model which registers one image onto another needs to have enough degrees of freedom to handle the image deformation. For a true pinhole camera, this model is a homography, which has 8 degrees of freedom. In practice, this model does not hold and must be augmented with additional transformations. A point $\mathbf{p}$ in one image is thus transformed to point $\mathbf{p}'$ in another image as:

$$\mathbf{p}^{'} = H \cdot L\left(\mathbf{p}; \theta\right) \tag{2}$$

where H is a homography and L is the additional transformation parameterized by θ. L is non-linear in general and can be a flow field, or a lens distortion correction. Here we propose to define L as a piecewise affine model:

$$L(\mathbf{q}) = \sum_{k=1}^{K} \delta(m(\mathbf{q}) - k) A_k \mathbf{q} \tag{3}$$

where K is the number of pieces, $m(\mathbf{q})$ is a function that determines the corresponding piece for point $\mathbf{q}$, and A_k is an affine transformation. Each piece covers a unique region of R^2. In other words, the pieces define a tessellation of R^2.

We have defined the tessellation to be triangular as follows. The image is first divided into a $2N \times 2N$ grid of equal sized cells, with $N \geq 1$. Each cell is then divided into 2 triangles, such that a radial pattern is created. In other words, cells in the northwest and southeast quadrants are divided with an up-diagonal and cells in the northeast and southwest quadrants are divided with a down-diagonal. The goal of this construction is to facilitate the representation of radial distortions. The tessellation is illustrated in Figure 2 for $N = 2$.

The piecewise affine model is parameterized by the coordinates of each grid point after the deformation. Let V_i be the known coordinates of grid point i before the deformation, and let V_i' be the unknown coordinates of the same grid point after the deformation. Given the three correspondences of a triangle, $V_i \leftrightarrow V_i'$, $V_j \leftrightarrow V_j'$, $V_k \leftrightarrow V_k'$, an affine transformation for that triangle is estimated by solving a small linear system. Therefore,

$$\theta = [V_1' V_2' \cdots V_M'] \tag{4}$$

where $M = (2N+1) \times (2N+1)$. For example, in Figure 2 there are 25 grid points, giving 50 parameters.

3.2. Estimating PAM

The piecewise affine model, as other parametric motion models, may be estimated using a direct method where all pixels are used in optimization [6] or a feature-based method. An advantage of a feature-based method is that it is usually faster than a direct method. However, as mentioned earlier, the disadvantage is that it requires correspondences to be uniformly distributed across the overlapping region, and the correspondences must be precise (low inlier noise) to guarantee a high quality mosaic. On the other hand, in a direct method, no correspondences are necessary. What is being optimized there is the intensity difference (or variance) of overlapping pixels. The disadvantage of this method is that it uses all overlapping pixels in the optimization, which can make the parameter estimation quite slow.

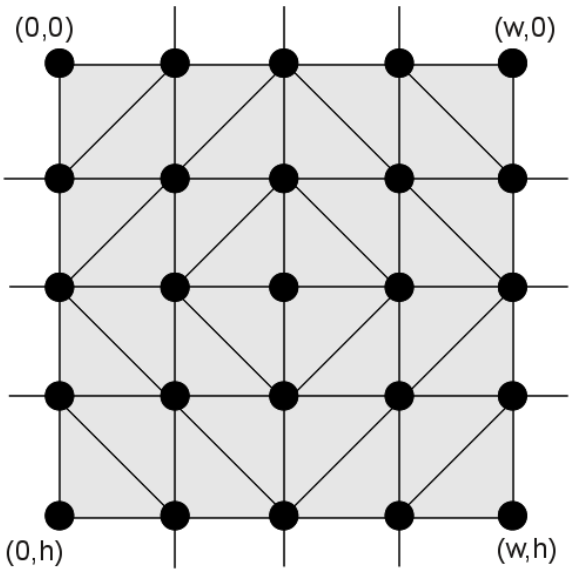


Figure 2. Tessellation of R^2 with respect to an image of width w and height h.

We initially used a feature-based approach, but the results were not satisfactory. After switching to a direct method, we noticed immediate improvement in mosaic quality. Unfortunately, as expected, the computational burden was indeed a problem. In our domain, the number of overlapping pixels in a 6-camera sensor array is over 35 million! This number of pixels does not lend itself to real-time parameter estimation.

Baker *et al.* [5] made the important observation that not all pixels are necessary in this optimization. In fact, pixels with zero, or very small gradients have no effect on convergence. By removing such pixels altogether, great computational savings are made. We have used a similar idea in our approach, but ensuring the pixels selected for optimization are uniformly distributed across the mosaic.

The mosaicking surface is first divided into a grid of $P \times P$ equal sized cells. We select one pixel from each cell, provided there is at least one pixel in the cell that overlaps two or more images. The pixel that gets selected is the one that has the highest Harris corner measure in one of the overlapping images. In other words, we select pixel p as

$$p = \arg\max_i \det(C_i) - \kappa \operatorname{tr}(C_i)^2 \tag{5}$$

where C_i is the second-moment matrix of pixel i. By selecting pixels in this manner, we ensure that all parts of the

mosaic are aligned well and that each selected pixel makes as much contribution to the optimization of parameters as possible.

When using a direct method, an initial solution of the parameters is required. The parameters of L are initialized so that PAM is an identity map. The parameters of H are initialized by finding corresponding points in the images, using SIFT [17] for example, and using a robust RANSAC estimator. When mosaicking more than two images, the homographies estimated are homographies to the reference image. For images that do not overlap with the reference image, a homography is estimated by concatenating appropriate neighboring homographies. The resulting homographies to the reference image are then jointly optimized to minimize the distance between corresponding points on the mosaicking surface, as expressed in this error function:

$$E(H_1, \cdots, H_I) = \sum_{\mathbf{p}_i, \mathbf{p}_j} ||H_j \mathbf{p}_j - H_i \mathbf{p}_i||_2^2 \qquad (6)$$

where I is the number of images, H_i is a homography from image i to the reference image and $\mathbf{p}_i, \mathbf{p}_j$ are a pair of corresponding points from images i and j.

Given an initial solution, the parameter estimation proceeds in a coarse-to-fine fashion. Since our initial solution is often quite accurate, we build an image pyramid with only 2 levels. In each level, we first minimize the intensity variance of the selected pixels over the homography parameters, keeping the PAM parameters fixed. This helps to ensure that the final warp is as rigid as possible, and the non-rigid deformation only accounts for the residuals not handled by the homography transformation. Then we minimize the intensity variance over the PAM parameters, keeping the homography parameters fixed. Formally, we minimize the following objective:

$$E(H_1, \cdots, H_I, \theta_1, \cdots, \theta_I) = \sum_{\mathbf{p}} \frac{1}{O(\mathbf{p})} \sum_{o}^{O(\mathbf{p})} \left(I_o \left(L_o^{-1} \left(H_o^{-1} \mathbf{p} \right) \right) - \mu(\mathbf{p}) \right)^2 \qquad (7)$$

where $\mathbf{p}$ is one of the selected pixels on the mosaicking surface, $O(\mathbf{p})$ is the number of overlapping images at pixel $\mathbf{p}$, $I_o(\mathbf{x})$ is the intensity of image o at location $\mathbf{x}$, and $\mu(\mathbf{p})$ is the mean intensity of the overlapping pixels. Note that taking the inverse of PAM (L^{-1}) is more expensive than taking the inverse of a simple homography. In practice, most of the image deformation is handled by the homography, so that it is sufficient to use small grid sizes in PAM (such as 4x4), which does not make the inverse computation a significant problem. We minimize the objective using Levenberg-Marquardt algorithm [16].

3.3. Intensity correction

Geometrical alignment is not sufficient to produce a seamless mosaic. Notice in Figure 1 that there is large variation in intensity across different cameras in the sensor array. Therefore, intensity alignment is also needed to generate a seamless mosaic. There are various approaches to do this in the literature, such as [13]. Here we use a simple and fast approach that works well in practice to remove gross intensity differences.

We parameterize the differences in intensity as differences in camera gain (scaling factor). In other words, the intensity corrected image $I^{'}$ is a scaled version of the input image I:

$$I^{'} = gI \qquad (8)$$

where g is an unknown gain correction factor. Therefore, we wish to estimate gain correction factors for each camera, such that the differences in intensity in the overlapping regions are minimized. It is easy to compute a correction factor for image A that optimally minimizes the intensity difference from image B. However, such an estimate is likely to conflict with a correction factor for the same image that minimizes the intensity difference from image C. It is clear the correction factors must be estimated jointly.

The pairwise correction factors provide a useful set of constraints for joint optimization. Consider the set of equations that must hold for optimal gain correction factors. Each pair of overlapping images A and B generates an equation such as this

$$g_A I_A = g_B I_B \qquad (9)$$

$$\frac{g_A}{g_B} I_A = I_B \qquad (10)$$

where g_A and g_B are unknown. For the same pair of images, we have

$$g_{AB} I_A = I_B \qquad (11)$$

where g_{AB} is the known optimal pairwise correction factor. This implies that we have

$$g_A - g_{AB} g_B = 0 \qquad (12)$$

for each pair of images. Stacking these equations together creates a linear system, whose non-trivial solution is the optimal set of gain correction factors, up to scale. We find the appropriate scale by assuming that the average of the corrections factors should equal 1. This minimizes the amount of intensity correction from input. It is necessary that the number of overlapping pairs of images be at least the number of images, but this is always satisfied for sensor arrays in our domain.

This intensity correction is performed before geometric alignment. To determine which pixels are in the overlapping regions, we use the initial solution of mosaic geometry from

SIFT correspondences. Knowing the optimal solution is not necessary here. After geometric alignment, the images are blended using [7] to remove any remaining seams resulting from residual photometric differences or alignment errors.

4. Complexity and Implementation

Parameters of PAM are estimated using non-linear optimization. The runtime depends on the number of iterations, and the complexity of each iteration. In each iteration, every sample is transformed to all the overlapping images and variance of the intensity at that location is estimated. Therefore, the complexity of each iteration is $O(SK)$, where S is the number of samples and K is the number of overlapping images. Note that $S \in O(P^2)$. When using analytical Jacobian, the number of iterations in each round of optimization is 19 on average.

For intensity correction, we first estimate the pairwise gain correction terms and then solve a small linear system using SVD. Estimating the pairwise gain correction terms is $O(MWH)$ where M is the number of overlapping pairs of images and W, H is the width and height of each image. In common sensor array configurations, $M < 2K$, where K is the number of images. The size of the linear system is $M \times K$, which can be solved in $O(K^3)$ time.

We have implemented the algorithm just presented in C++. Our unoptimized implementation can estimate a mosaick of 6 WAAS images in 50 seconds for a model with $N = 2$. The implementation is available on the author's website [1].

5. Results

The proposed algorithm was evaluated on the CLIF2006 dataset [2], which was captured by an array of 6 cameras. Each of the six images is grayscale and 4008x2672 in size. The performance of mosaicking was quantified using (7), except that all overlapping pixels were used in the evaluation. In the following discussion and plots, this number is divided by the number of pixels to give an average variance of intensity.

To determine the appropriate number of samples, needed to robustly estimate the mosaic, we took an empirical approach. Note that the number of samples is proportional to the sampling grid size P. Usually, the number is $< P^2$, since many of the cells do not contain any pixels with overlapping images. In the dataset used for evaluation, a value of $P = 100$ corresponds to approximately 2762 samples, $P = 300$ corresponds to 23568, and $P = 450$ to is 52316. Five images were randomly selected from the dataset, and mosaicking performance was measured for different values of P. This plot can be seen in Figure 3 for two different values of N, an indicator of model complexity. Both plots in the figure show that in general, more samples lead to better accuracy. However, it is also interesting to see that the model with higher complexity ($N = 2$) is more susceptible to falling into local minima, as is evidenced by several sharp peaks for relatively high values of P. From these plots, we determined that an appropriate value of P for $N = 1$ is 350 and for $N = 2$ is 420. For both of these models, the number of samples is a small fraction of the total number of overlapping pixels, which is in the millions. As discussed below, we can generate a better quality mosaic using this smaller number of pixels than previous work which used all overlapping pixels. This shows the power of our sampling strategy.

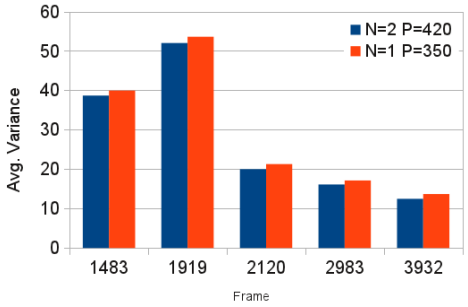
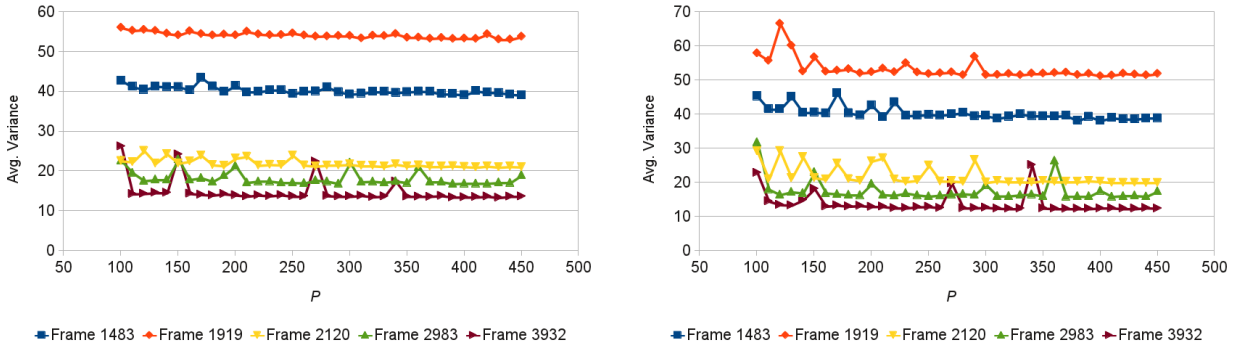


Figure 4. Effect of PAM complexity on mosaicking accuracy.

Similarly, to determine the appropriate model complexity (value of N) needed to robustly estimate the mosaic, we also took an empirical approach. For the same five images selected earlier, we measured mosaicking performance for different values of N. This plot can be seen in Figure 3, but a more informative plot is in Figure 4, where the models are compared with the empirically selected values of P. As expected, the model with higher complexity is able to generate a more accurate mosaic, although the difference between the two models is not large. Since our primary goal is the most accurate mosaic possible, we prefer model with $N = 2$. However, if the efficiency is a more important concern in some applications, model with $N = 1$ may be a good alternative.

We have also compared our approach with competing approaches [22, 20, 19] that use the lens distortion model (LDM) for non-rigid deformations. Our implementation models 2 radial distortion coefficients and 2 tangential distortion coefficients. In many of these approaches, the objective function is minimized using all overlapping pixels. In the dataset we used for evaluation, there are tens of millions of such pixels, which makes using all of them intractable. Therefore, in our implementation, we uniformly sampled 200,000 overlapping pixels for minimizing the objective. Similarly, for the baseline comparison with no non-rigid deformation (homography only), we have also used "only" 200,000 pixels in the minimization. Quantitative compar-



(a) N=1 (b) N=2

Figure 3. Effect of the sampling grid size P on mosaicking accuracy. In general, the more samples are used, the better the accuracy. However, a model with higher complexity $N = 2$ is more likely to fall into local minima during parameter estimation.

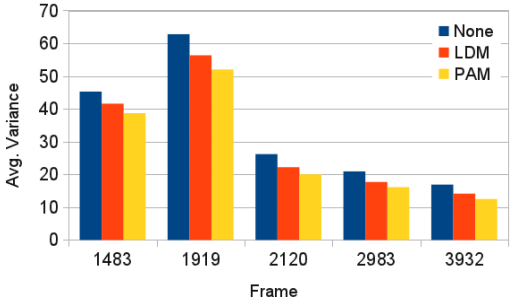


Figure 5. Comparison of different deformation models.

ison with the lens distortion model as well as the baseline is illustrated in Figure 5. The lens distortion deformation model generates better mosaics than no deformation at all, however, the proposed piecewise affine model is able to achieve futher improvement, outperforming the LDM. Note that it takes more than 3 minutes to compute an LDM mosaic with that many pixels in our implementation.

Qualitative comparison of the piecewise affine model and the standard lens distortion model is illustrated in Figure 6. The piecewise affine model has practically perfect registration, while the lens distortion model has some remaining error. More qualitative results are shown in Figure 8 and in the supplemental material.

We evaluated the proposed intensity correction algorithm qualitatively. As Figure 7 shows, our approach removes most of the intensity differences between the different cameras. Any remaining differences will disappear after blending the images [7].

Our mosaicking algorithm is quite efficient. With $N = 2$ and $P = 420$, a mosaick of 6 WAAS images is estimated in 50 seconds on a 3.0 GHz Intel Xeon X5450 CPU. We are in the process of porting our approach to the GPU. Preliminary analysis shows potential for real-time performance on WAAS imagery.

6. Conclusions

Mosaicking a sensor array from WAAS is the first step in the semantics extraction process. This step is critical, because all the following vision tasks depend on it being reliable. We have proposed an approach that generates high-quality mosaics in an efficient manner, suitable for such tasks. A comparison with the standard lens distortion model used in previous work reveals that the proposed model is superior.

In the future, we plan to investigate more approaches for efficient joint optimization of parametric motion models, such as sampling strategies, objective criteria, and numerical methods. At the same time, we plan to complete a GPU implementation of our approach with real-time performance.

7. Acknowledgments

This work was supported in part by grant DE-FG52-08NA28775 from the U.S. Department of Energy and by funding from Lawrence Livermore National Laboratory.

References

[1] http://iris.usc.edu/people/prokaj/. 277
[2] CLIF 2006. https://www.sdms.afrl.af.mil/index.php?collection=clif2006. 273, 277
[3] CLIF 2007. https://www.sdms.afrl.af.mil/index.php?collection=clif2007. 273
[4] Greene 2007. https://www.sdms.afrl.af.mil/index.php?collection=greene07. 273
[5] S. Baker, R. Gross, I. Matthews, and T. Ishikawa. Lucas-kanade 20 years on: A unifying framework: Part 2. Technical Report CMU-RI-TR-03-01, Robotics Institute, 2003. 275

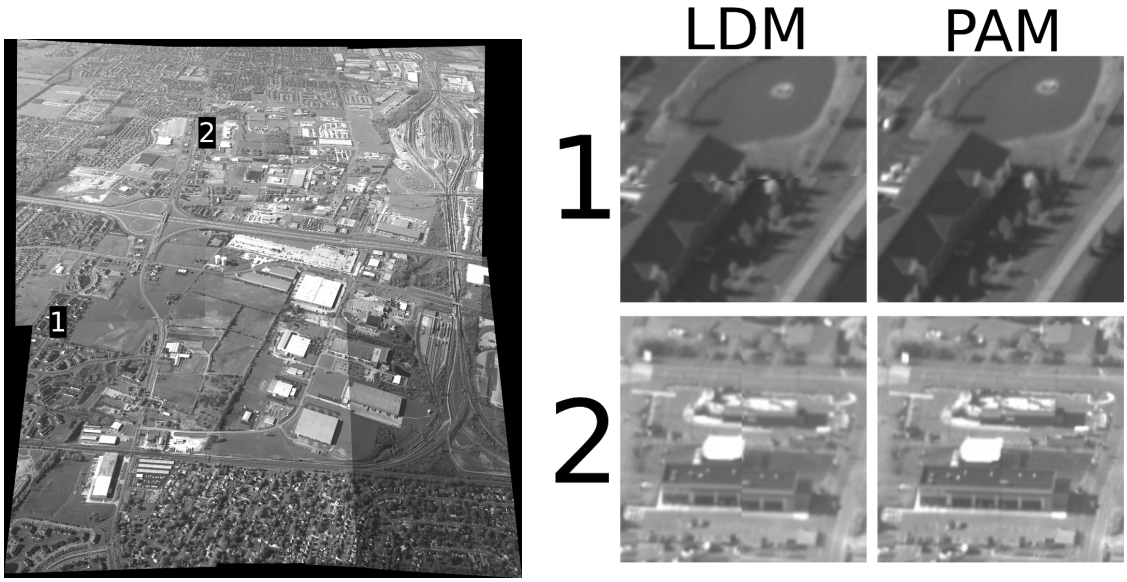
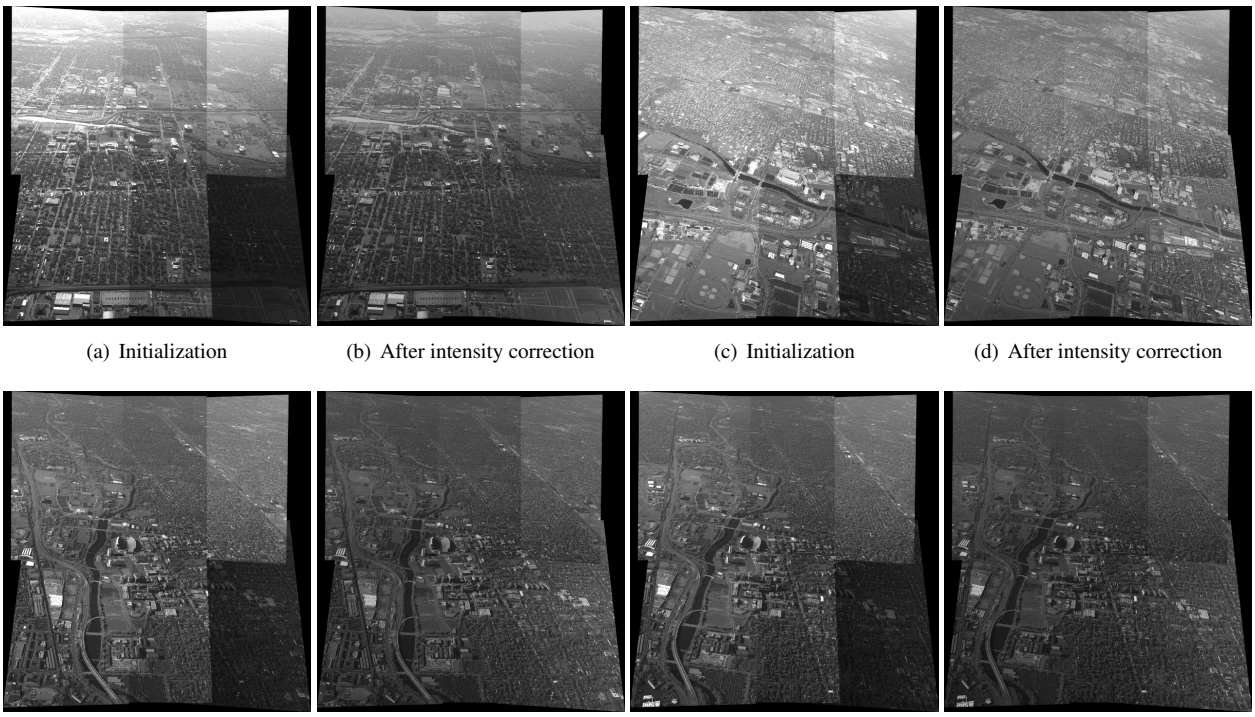


Figure 6. Mosaick comparison between a lens distortion model (LDM) and the proposed piecewise affine model (PAM). Notice a visible seam in the LDM mosaic.



(a) Initialization (b) After intensity correction (c) Initialization (d) After intensity correction

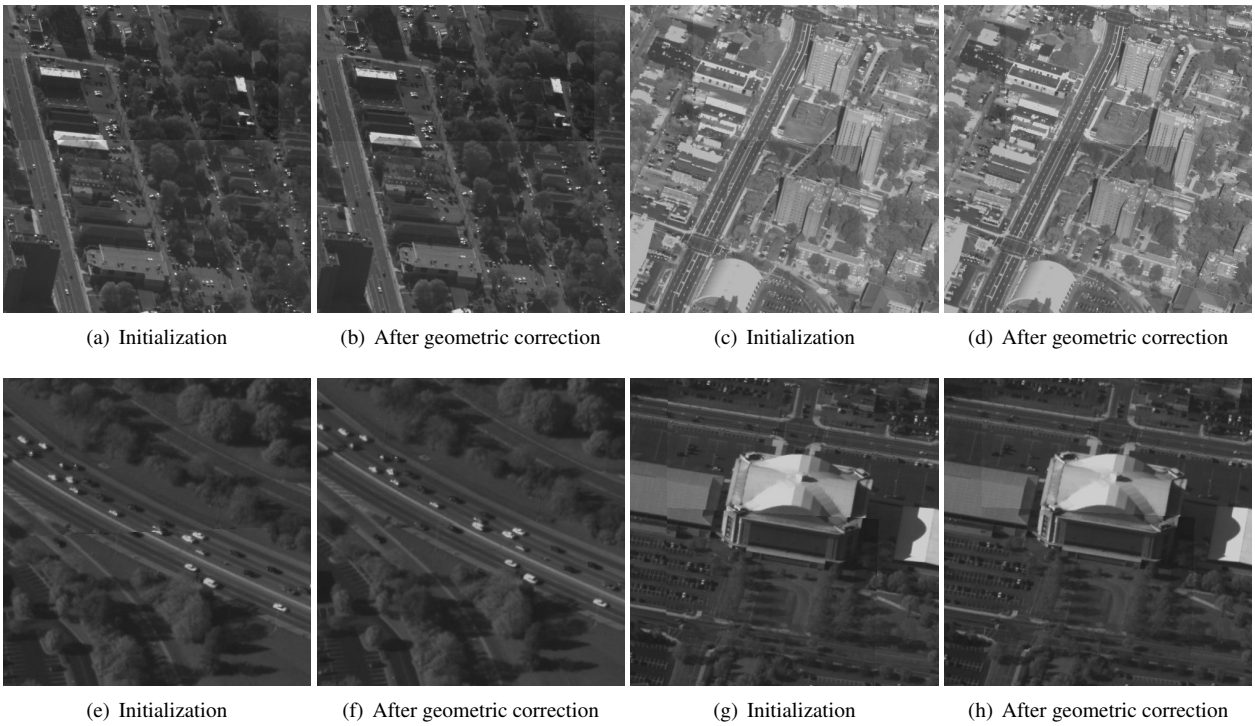
(e) Initialization (f) After intensity correction (g) Initialization (h) After intensity correction

Figure 7. Results of intensity correction on frames 1483, 1919, 2120, and 2983.

[6] J. Bergen, P. Anandan, K. Hanna, and R. Hingorani. Hierarchical model-based motion estimation. In *ECCV*, volume 588 of *Lecture Notes in Computer Science*, pages 237–252. Springer, 1992. 275

[7] P. J. Burt and E. H. Adelson. A multiresolution spline with application to image mosaics. *ACM Trans. Graph.*, 2:217–236, 1983. 277, 278

[8] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2nd edition, 2003. 274



(a) Initialization (b) After geometric correction (c) Initialization (d) After geometric correction

(e) Initialization (f) After geometric correction (g) Initialization (h) After geometric correction

Figure 8. Detailed results of an estimated mosaic on frames 1483, 1919, 2120, and 2983.

[9] T. Igarashi, T. Moscovich, and J. F. Hughes. As-rigid-as-possible shape manipulation. *ACM Trans. Graph.*, 24:1134–1141, July 2005. 274

[10] M. Irani, P. Anandan, and S. Hsu. Mosaic based representations of video sequences and their applications. In *International Conference on Computer Vision*, pages 605 –611, 1995. 273

[11] M. Irani, S. Hsu, and P. Anandan. Video compression using mosaic representations. *Signal Processing: Image Communication*, 7(4-6):529 – 552, 1995. Coding Techniques for Very Low Bit-Rate Video. 273

[12] E.-Y. Kang, I. Cohen, and G. Medioni. A graph-based global registration for 2d mosaics. In *International Conference on Pattern Recognition*, volume 1, pages 257 –260, 2000. 274

[13] S. J. Kim and M. Pollefeys. Radiometric alignment of image sequences. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 645 – 651, 2004. 276

[14] W.-Y. Lin, S. Liu, Y. Matsushita, T.-T. Ng, and L.-F. Cheong. Smoothly varying affine stitching. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 345 –352, 2011. 274

[15] F. Liu, M. Gleicher, H. Jin, and A. Agarwala. Content-preserving warps for 3d video stabilization. *ACM Trans. Graph.*, 28:44:1–44:9, July 2009. 274

[16] M. Lourakis. levmar: Levenberg-marquardt nonlinear least squares algorithms in C/C++. http://www.ics.forth.gr/~lourakis/levmar/, 2004. 276

[17] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004. 276

[18] S. Mann and R. Picard. Virtual bellows: constructing high quality stills from video. In *IEEE International Conference on Image Processing*, volume 1, pages 363 –367, 1994. 273

[19] H. Sawhney, S. Hsu, and R. Kumar. Robust video mosaicing through topology inference and local to global alignment. In *ECCV*, volume 1407 of *Lecture Notes in Computer Science*, pages 103–119. Springer, 1998. 273, 274, 277

[20] H. Sawhney and R. Kumar. True multi-image alignment and its application to mosaicing and lens distortion correction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(3):235 –243, mar 1999. 273, 274, 277

[21] H.-Y. Shum and R. Szeliski. Systems and experiment paper: Construction of panoramic image mosaics with global and local alignment. *International Journal of Computer Vision*, 36:101–130, 2000. 274

[22] S. N. Sinha and M. Pollefeys. Pan-tilt-zoom camera calibration and high-resolution mosaic generation. *Computer Vision and Image Understanding*, 103(3):170 – 183, 2006. Special issue on Omnidirectional Vision and Camera Networks. 273, 274, 277

[23] R. Szeliski. Image mosaicing for tele-reality applications. In *IEEE Workshop on Applications of Computer Vision*, pages 44 –53, 1994. 273

[24] R. Szeliski. Image alignment and stitching: A tutorial. Technical Report MSR-TR-2004-92, Microsoft Research, 2006. 274