Efficient generation of multi-perspective panoramas

Enliang Zheng Rahul Raguram Pierre Fite-Georgel
Jan-Michael Frahm
University of North Carolina at Chapel Hill
Chapel Hill, USA

Figure 1: A short segment of a long multi-perspective panorama generated by our system from a video sequence of a street scene.

Abstract

In this paper, we present an efficient technique for generating multi-perspective panoramic images of long scenes. The input to our system is a video sequence captured by a moving camera navigating through a long scene, and our goal is to efficiently generate a panoramic summary of the scene. This problem has received considerable attention in recent years, leading to the development of a number of systems capable of generating high-quality panoramas. However, a significant limitation of current systems is their computational complexity: most current techniques employ computationally expensive algorithms (such as structure-from-motion and dense stereo), or require some degree of manual interaction. In turn, this limits the scalability of the algorithms as well as their ease of implementation. In contrast, the technique we present is simple, efficient, easy to implement, and produces results of comparable quality to state of the art techniques, while doing so at a fraction of the computational cost. Our system operates entirely in the 2D image domain, performing robust image alignment and optical flow based mosaicing, in lieu of more expensive 3D pose/structure computation. We demonstrate the effectiveness of our system on a number of challenging image sequences.

1. Introduction

Recent years have seen a growing interest in the mapping and visualization of the world's cities and sights. For instance, systems such as Google Street View and Bing Maps enable users to browse street level imagery by presenting a panorama-based visualization of video captured at street level from a moving vehicle. In addition, a number of recent efforts have been directed towards enabling the visualization of large internet-based photo collections by generating 3D models of landmarks and cities [1, 6]. With the rapid growth of digital image content, enabling the visualization of this content in efficient and effective ways is an interesting research problem.

In this work, we consider the problem of summarizing video sequences captured by a camera translating horizontally through a long scene. One of the most compelling means of visualizing data of this form is via multi-perspective panoramas [15, 2]. In their most simple form ("strip panoramas"), this amounts to extracting narrow vertical strips from each frame of a video sequence and then aligning them to generate the final panorama. Since each strip of the final panorama is captured from a slightly different viewpoint, these panormas simulate orthographic projection along the horizontal axis. In addition, since each individual strip is obtained from a single perspective image, the projection along the vertical axis of the panorama is perspective in nature. While this often leads to reasonable results, the simple technique outlined here has a number of practical shortcomings, and considerable effort has been made to improve upon this basic technique. Exciting results have been achieved in recent years, with the development of more sophisticated techniques for generating high-quality multi-viewpoint panoramic images [2, 13, 8].

A significant limitation of these state of the art tech-

niques, however, is their computational complexity. In order to achieve high quality results, these techniques often require the use of expensive operations (such as structure from motion and dense stereo), which implies that the scalability of these techniques is limited. In addition, this also poses a challenge in terms of implementation, since building these systems is a non-trivial task. Furthermore, deploying these techniques on low cost mobile computing platforms, such as cellphones, poses significant difficulties.

In this paper, we aim to overcome some of these limitations, by presenting a very simple technique for generating multi-viewpoint panoramas, which produces results that are comparable in quality to the state of the art. While most current techniques rely on 3D information obtained either by performing rigorous structure from motion, or via manual interaction, we demonstrate that using a combination of robust alignment techniques and simple 2D optical flow information is often sufficient to obtain good results in a number of scenarios of practical interest. Our main contributions in this work are: (a) a novel method for aligning frames of an image sequence that is robust to scale and rotation drift and (b) a new cost function based on optical flow, that allows the selection of regions from each image while also minimizing visual artifacts. The modules of our system are simple, and even an unoptimized implementation yields an order of magnitude improvement in speed compared to current techniques.

The remainder of this paper is organized as follows. Section 2 briefly discusses related work in multi-perspective panorama generation. In section 3, we describe in detail our proposed technique and outline the various modules of our system. In Section 4, we present results on a number of challenging image sequences. Section 5 concludes the paper with a discussion of possible future extensions.

2. Related Work

Generating panoramic images is a topic that has been well studied over the years, and a wide body of work exists in this area [17, 11, 16, 2, 15, 3, 13, 8]. While it is beyond the scope of this work to fully summarize the many contributions made over the years, in this section we briefly survey some of the most related work.

Over the years, a number of techniques have been proposed to generate multi-perspective panoramas. Some of these techniques include pushbroom panoramas [16], x-slit images [18] and manifold mosaics [11], all of which describe techniques for creating strip panoramas of long scenes. However, these techniques all suffer from characteristic distortion effects for scenes with large depth variations. In particular, distant objects typically become wider and closer objects become narrower. Much effort, therefore, has been spent on reducing these distortion effects. For instance, [14] defines a cost function based on aspect ratio distortion, which is then minimized by appropriate choice of scene segments. The work of [2] moves away from the selection of simple strips, instead using a Markov Random Field (MRF) optimization approach to select arbitrarily shaped regions of the source images that possess various desirable properties. This technique, however, requires some degree of manual interaction to "clean up" visual distortions created by the algorithm. In addition, the technique is computationally intensive, and is thus limited to working with a set of photographs (as opposed to a video sequence). Most recently, the work of [8] presents a complete navigational system that attempts to combine the strengths of single- and multi-perspective panoramas, with smooth transitions between the two modes.

Our work is perhaps most inspired by the recent minimal-aspect distortion (MAD) panorama work from [13]. In this work, it is observed that for scenes with large depth variations only perspective projection results in an undistorted image, while for constant-depth scenes, virtually any projection will result in an undistorted mosaic. The process of mosaicing and strip selection is thus cast as a problem of cost minimization, where the objective function incorporates knowledge of camera motion and scene depth. Our proposed approach assumes a similar framework, but uses a much simpler cost function, based on 2D optical flow information. The intuition behind this approach is that for a properly aligned sequence, optical flow can be used as a proxy for disparity information. Our modified cost function is thus much simpler, and far more efficient.

Finally, we note that a number of techniques address the case of single viewpoint panoramas, typically obtained by rotating the camera about a common optical center. Such techniques (for instance, [3]) have attained an impressive level of maturity, with many consumer cameras today including a panoramic mode that produces rotational panoramas "on-the-fly". However, for the case of multi-viewpoint panoramas, much work remains to be done before this level of widespread adoption can be attained. It is our view that using simple techniques for panorama generation encourages the implementation of our method on low-cost mobile computing platforms, such as cellphones.

3. The method

In this section, we describe in detail the steps of our proposed approach. Our approach to panorama generation is inspired by that of [13], where the process of strip selection is formulated in terms of finding a cut in the space-time volume of a video sequence. However, while [13] requires a full 3D reconstruction pipeline, including dense stereo computation, we choose to make use of simpler 2D constraints and flow information. An overview of our proposed approach is shown in Figure 2.

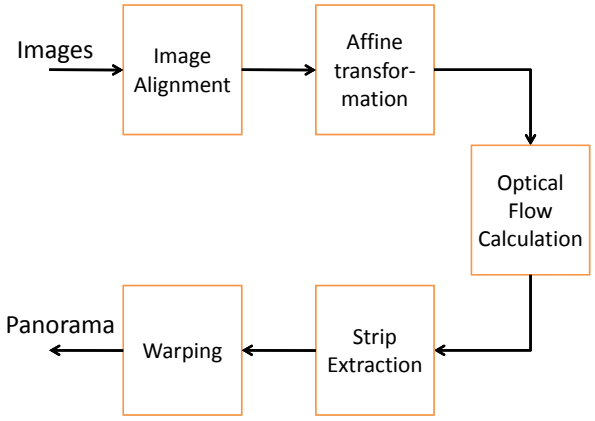


Figure 2: Overview of the proposed system, showing the various modules.

3.1. Image alignment

The first step in our processing pipeline is the pairwise alignment of the frames in the video sequence. In this step, the goal is to align all frames into a common coordinate system. Once this alignment has been performed, the process of panorama generation amounts to finding the best regions from each frame to be mosaiced into the final panorama. Performing an alignment of this form is reasonable when the camera motion is mainly translational with a rotation around the optical axis. While this covers a number of cases of practical interest, we note that in cases of more complex motion, this might lead to visual artifacts.

It is worth noting that there is no single "correct" multi-viewpoint panorama, in the sense that no fixed camera can capture a multi-viewpoint image at a single point in time that is perspectively correct from every viewpoint. This observation is also made in [8], where the goal is to generate an "as-perspective-as-possible" panorama. We adopt the same view, though we use much simpler techniques to try and achieve similar results.

Given a video sequence, we assume that the motion between frames is small, implying that the transformation between adjacent frames can be modeled as a 2D affine transformation [10]. It has been observed that using this type of rigid alignment has some limitations. For instance, using a pairwise alignment method to incrementally register frames into a single reference coordinate system (usually defined by the first frame of the sequence) leads to a drift problem. While the pairwise transformations are locally correct, the error in this transformation accumulates over a long sequence, leading to curved panoramas [19]. In addition, when the camera motion is non-orthogonal to the dominant scene structure, this can sometimes cause changes in scale, with the panorama either shrinking or expanding in the vertical direction. While current techniques solve these problems by relying on camera pose and 3D structure information obtained via structure from motion combined with computationally expensive bundle adjustment, we attempt to use simpler and less computationally expensive techniques. In particular, we look to combat these effects by using a simple *constrained* affine transformation model, making the assumption that the motion of the camera is primarily translational or rotational about a fixed camera center (note that a similar assumption is made in [13]).

Using a feature-based alignment method, assume that we have a set of N feature correspondences between successive image frames. We denote this by $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$, $i = 1, 2...N$. Given a set of tentative correspondences, we use RANSAC [5] to estimate a set of I inliers, and then attempt to estimate a transformation $\mathbf{H}_A$, which is of the form

$$\begin{pmatrix} \mathbf{A} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{pmatrix}$$

where $\mathbf{A}$ is a 2×2 non-singular matrix and $\mathbf{t}$ is a 2-vector consisting of $[t_x, t_y]^T$, denoting horizontal and vertical translations, respectively. This transformation is estimated by carrying out a minimization over the set of I inliers, using the following formulation:

$$minimize : \|\mathbf{H}_A\mathbf{x}'_i - \mathbf{x}_i\| + \lambda|t_y| \quad (1a)$$

$$subject\ to : \|\mathbf{Au}\| <= 1 - \beta \quad (1b)$$

$$\|\mathbf{Av}\| <= 1 - \beta \quad (1c)$$

$$1 - \alpha <= \mathbf{u}'\mathbf{Au} <= 1 - \alpha \quad (1d)$$

$$1 - \alpha <= \mathbf{v}'\mathbf{Av} <= 1 - \alpha \quad (1e)$$

Where $\mathbf{u}$ and $\mathbf{v}$ are the unit vectors $[1, 0]$ and $[0, 1]$ in the x and y directions respectively, t_y is the translation in the y-direction, λ is a weighting factor, and α and β are constants close to zero. Intuitively, the goal of these constraints is the following: we would like the panorama to be relatively straight, and to have a consistent scale. The effect of the matrix $\mathbf{A}$ is a rotation and a non-isotropic scaling [7]. While we would like our estimated transform to be more flexible than a strict similarity transformation, we would also like to ensure that the rotation is not too large, and that the scale does not drift. To do so, we impose constraints on the effect of the tranformation $\mathbf{H}_A$ on the unit vectors $\mathbf{u}$ and $\mathbf{v}$. The first two constraints in the above equations (1b and 1c) ensure that the scale does not drift, while the last two (1d and 1e) ensure that the vectors are not rotated by too large an angle. The parameters α and β are set to small values ($\approx 10^{-4}$). Finally, the parameter λ is used to balance the effects of vertical translation and alignment error. The problem formulated above is a typical inequality constrained convex optimization problem, and can be solved

by the interior-point algorithm [9]. Note that estimating the transformation using the constraints discussed in this section can lead to slight local misalignments, while simultaneously improving the global alignment and reducing drift. However, we have observed that in practice, this decrease in local alignment quality is acceptable.

We use a fast GPU implementation of SIFT feature detection and matching [1] to detect tentative feature correspondences, and then use an efficient RANSAC implementation [12] to find inlier matches. The speed of this stage of the pipeline is on the order of 12 Hz on a PC equipped with an Intel Xeon 2GHz processor and an Nvidia GTX285 graphics card.

3.2. Transformation

Once the initial alignment has been performed, the estimated pairwise transformations are used to derotate and vertically align the images. More specifically, an inverse rotation is applied to each image, followed by a vertical translation by the computed t_y. Thus, the only remaining component of the image motion is a translation in the horizontal direction. We exploit this remaining component in order to perform the selection of regions from each image.

3.3. Optical flow computation

While planar scenes are well suited for multi-viewpoint panoramas, scenes with large variation in depth pose significant problems. Due to the effect of parallax, using a feature-based alignment method (as described in Sec 3.1) to align images and select strips from each image results in visual artifacts, with foreground objects being truncated and background objects being duplicated. While the content within each image strip exhibits perspective projection, the artifacts occur near the border of two image strips, due to inconsistent strip boundary selection that does not account for local motion. The boundary of the strips selected from each image must thus be adapted to the depth of the scene. While current techniques such as [13] use dense stereo to estimate the scene depth at every pixel, we use optical flow information as a proxy for pixel disparities. In particular, our technique is based on the intuition that if we are able to select an image strip where the optical flow at the strip border corresponds to the overall horizontal translation, then we have obtained a cut that does not distort the dominant aspect of the scene. This process of panorama generation is further illustrated in Figure 3. As seen from the figure, we extract an image patch from each image and warp it to a rectangular strip before placing it in the panorama. Given the $(t-1)^{th}$ and t^{th} images, $I(t-1)$ and $I(t)$, let us denote the position of the right boundary of the strip in image $I(t)$ by $C_R(t)$, and the left boundary by $C_L(t)$. Assume that we know $C_R(t-1)$ (the selection of C_R is introduced

[1] Available online: http://cs.unc.edu/~ccwu/siftgpu/

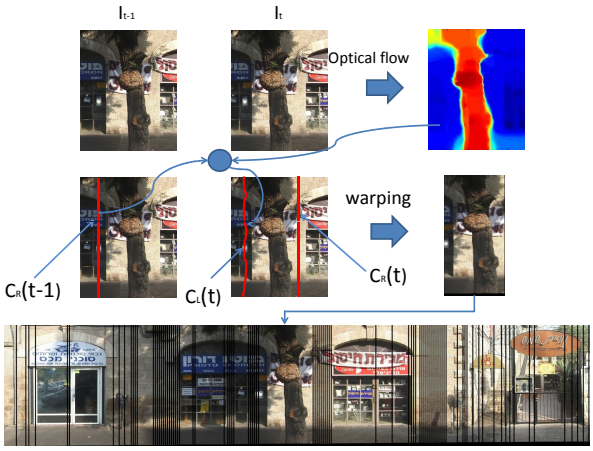


Figure 3: Illustration of the optical flow based strip selection method used in our approach.

in Section 3.4), then $C_L(t)$ is predicted using the computed optical flow between $I(t-1)$ and $I(t)$, as:

$$C_L(t) = C_R(t-1) + O_{C_R(t-1)} \tag{2}$$

where $O_{C_R(t-1)}$ is the optical flow for each pixel in $C_R(t-1)$. Some points are worth noting here. Firstly, as $C_R(t-1)$ is straight, the shape of $C_L(t)$ is determined by $O_{C_R(t-1)}$. If this boundary deviates significantly from a line, there will be more distortion when warping the image patch into a rectangular strip (we discuss warping in more detail in Section 3.5). In other words, strips that have a border with lower variance in their optical flow imply that less distortion will occur during the compositing phase. This is similar to the conclusion reached in [13], where the variance of pixel disparities (computed from dense stereo) is used for strip selection. In our case, we use optical flow as a proxy for disparity, which provides similar results.

To perform the process of strip selection, we formulate a cost function which is then minimized over the set of images. Assume that each patch has an associated cost. According to the analysis above, the cost function should be defined as the variance of $O_{C_R(t-1)}$. However, the computed optical flow is not perfect, and sometimes has errors, particularly in the case of large motion. In this case, the computed border $C_L(t)$ will be inconsistent, and hence leads to visual artifacts. In our technique, we define the following cost function, which is more robust to errors in the optical flow:

$$cost(C_R(t-1)) = \|O_{C_R(t-1)} - t_x\mathbf{1}\|_1 \tag{3}$$

where $O_{C_R(t-1)}$ is the optical flow at $C_R(t-1)$ and $\mathbf{1}$ represents a vector with each component equal to 1. As we have

derotated and vertically aligned the images, ideally the vertical component of the optical flow should be zero. t_x is the translation that is estimated using the dominant scene structure, and we thus attempt to select strips that align this part well. In other words, if we find a cut C_{min} that minimizes Eqn. (3), there are two implications: (a) the optical flow calculated on $C_R(t-1)$ is reliable and (b) the variance of the optical flow for the pixels in $C_R(t-1)$ is small. When the cost function equals zero, the optical flow is exactly the same as the horizontal translation between the image pair and we obtain a well aligned panorama. For scenes with large depth variation where no "perfect" cut exists, the intuition is to find the minimum cost cut, which makes the optical flow at the seam match the dominant scene motion and minimizes any visual distortion effects. The exact process of minimizing this cost function over the complete set of images is discussed in the next section.

In our system, we make use of a fast GPU-based optical flow algorithm [2], which operates at speeds of ≈ 5 Hz for 720x480 images.

3.4. Strip extraction

In the previous section, we introduced a cost function that seeks to minimize the seam effect between a pair of images. To generate the final panorama, we need to find all strips across the entire image sequence and composite them into a single image. To do so, we adopt a similar strategy as in [13].

In order to make a panorama with the minimum overall distortion, the total cost is defined as the sum of the cost for each successive image in the sequence:

$$S = \sum_t cost(C_R(t)) \tag{4}$$

Minimizing the total cost given in equation (4) is equivalent to finding the best strip from each image. Solving the minimization problem can be formulated as finding the minimum path in the graph, which is constructed as shown in Figure 4. In Figure 4, the x-axis represents image columns, while the y-axis represents different image frames. Each node thus corresponds to a column in a given image. The edges connecting the source node S and the nodes of the first row have $cost(C_R(1))$. Each node for image $I(t)$ is connected to all nodes in image $I(t+1)$, with the weight of the edge being defined by the cost function $cost(C_R(t+1))$. For efficiency, we exclude those edges that make $C_R(t+1) < C_L(t+1)$, which prevents the strips from going spatially backwards. To find the panorama with least distortion, we compute the shortest path from the node S to the last row (i.e., the last frame in the image sequence).

[2] Available online: http://www.inf.ethz.ch/personal/chzach/opensource.html

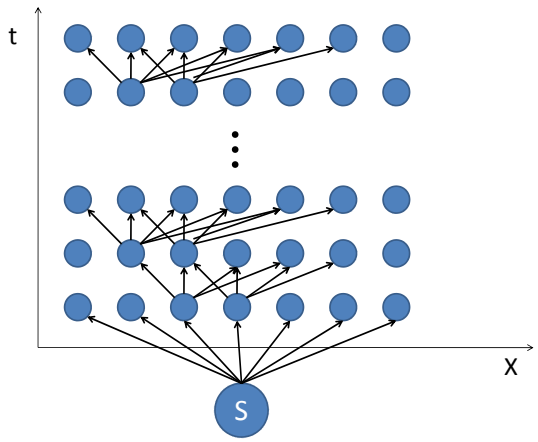


Figure 4: Graph used to perform the minimization of Eqn. 4. The x-axis represents image columns, while the y-axis represents different image frames. Each node thus corresponds to a column in a given image, and there exists a path from the source (S) to the last frame in the sequence. The goal is to find the path with minimum cost.

This graph can be easily and efficiently solved by dynamic programming [4].

3.5. Warping

To composite the final panorama, the right border of the image patch, which is a perpendicular line, is found through calculating the minimum path of the graph. The left curved border of the image patch is calculated using (2). By warping each line of the image patch to the size of the average width of the patch, we obtain a rectangular image strip, which is then inserted into the final panorama (Figure 3).

4. Results

In this section, we present the results of our panorama generation algorithm on a number of challenging image sequences. We use the same parameter values for all experiments, setting $\alpha = 10^{-4}$ and $\beta = 10^{-5}$ (for Equation (1)). The results of our algorithm are not very sensitive to the λ weighting factor, and we obtained acceptable results for values ranging from $0.1 - 10$. To demonstrate the effectiveness of our technique, we compare our results against the work of [13], which is one of the state of the art techniques for multi-perspective panorama generation. We use the same video sequences as in [13] to generate multi-viewpoint panoramas.

The **Hillel** sequence is a long street-level video sequence, consisting of 2200 frames with resolution 720x480. The final panorama has a resolution of 49057x720 [3]. Figures 5(a)

[3] Sample results available online: http://www.cs.unc.edu/∼ezheng/panorama/panorama.html

and 5(b) show short excerpts from the final panorama, constructed from 350 frames of the input video. Figure 5(a) is the result of our proposed method, while Figure 5(b) is the result obtained using the MAD algorithm [13]. It can be seen that the overall quality is comparable across the two methods. While the result produced by our system shows more local distortion effects, note that the MAD method is prone to drift (visible towards the right edge of the Figure 5(b)). In addition, our technique was able to process frames at a rate of approximately 1 Hz (using an unoptimized MATLAB implementation), whereas [13] report processing times of 10-30 seconds per frame, even when using downsampled 320x240 images (half the resolution at which our system operates). This is a considerable improvement in efficiency, while achieving results of similar quality. Figure 6 shows a second comparison of our technique with the MAD algorithm, using 450 from the same **Hillel** sequence. It can be seen that our system provides results of good quality, due to the robust alignment and optical flow-based strip selection methods.

The **Train** and **Boat** sequences contain 543 and 449 frames of video, respectively. The resolution of the **Train** sequence is 640x360, while that of the **Boat** sequence is 345x466. The resulting panoramas are shown in Figure 7 and 8. For brevity, we do not present the results from [13], which are all available online at: http://www.vision.huji.ac.il/mad/. We note that the panoramas we generate are very similar in overall quality to those obtained using the MAD technique.

5. Conclusion

In this paper, we have presented a simple, yet effective method for multi-viewpoint panorama generation. Our system provides results that are similar in quality to state of the art techniques, while achieving computational speed-ups of up to an order of magnitude. Our system leverages simple and robust 2D alignment and flow techniques in order to effectively generate panoramas for long video sequences. The simplicity of our technique is encouraging, particularly with a view towards implementation on low-end mobile computing devices such as cellphones, which are slowly beginning to incorporate more computing power, including on-board GPU units. In future work, we aim to transition our system to these mobile devices, so as to produce multi-viewpoint panoramas "on the fly".

6. Acknowledgement

We would like to acknowledge NSF grant IIS-0916829, DOE grant DE-FG52-08NA28778 for supporting parts of this research.

References

[1] S. Agarwal, N. Snavely, I. Simon, S. M. Seitz, and R. Szeliski. Building Rome in a day. In *ICCV*, 2009.

[2] A. Agarwala, M. Agrawala, M. Cohen, D. Salesin, and R. Szeliski. Photographing long scenes with multi-viewpoint panoramas. In *ACM SIGGRAPH*, pages 853–861, 2006.

[3] M. Brown and D. G. Lowe. Automatic panoramic image stitching using invariant features. *Int. J. Comput. Vision*, 74:59–73, August 2007.

[4] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. The MIT Press, 2nd revised edition edition, September 2001.

[5] 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.

[6] J.-M. Frahm, P. Fite-Georgel, D. Gallup, T. Johnson, R. Raguram, C. Wu, Y.-H. Jen, E. Dunn, B. Clipp, S. Lazebnik, and M. Pollefeys. Building Rome on a Cloudless Day. In *ECCV*, volume 6314, pages 368–381, 2010.

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

[8] J. Kopf, B. Chen, R. Szeliski, and M. Cohen. Street slide: Browsing street level imagery. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2010)*, 29(4):96:1 – 96:8, 2010.

[9] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, August 2000.

[10] S. Peleg and J. Herman. Panoramic mosaics by manifold projection. In *Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition (CVPR '97)*, CVPR '97, pages 338–, 1997.

[11] S. Peleg, B. Rousso, A. Rav-Acha, and A. Zomet. Mosaicing on adaptive manifolds. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22:1144–1154, October 2000.

[12] R. Raguram, J.-M. Frahm, and M. Pollefeys. A comparative analysis of RANSAC techniques leading to adaptive real-time random sample consensus. In *European Conference on Computer Vision*, pages II: 500–513, 2008.

[13] A. Rav-Acha, G. Engel, and S. Peleg. Minimal aspect distortion (mad) mosaicing of long scenes. *Int. J. Comput. Vision*, 78:187–206, July 2008.

[14] A. Roman. *Multiperspective Imaging for Automated Urban Imaging*. PhD thesis, Stanford University, USA, 2006.

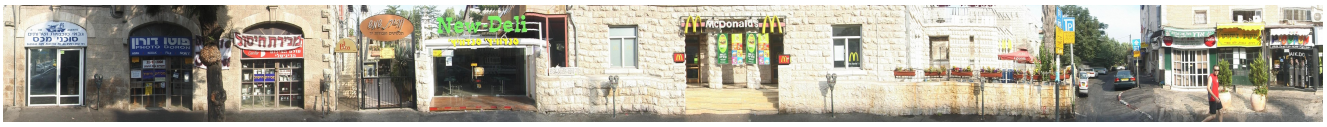
[15] A. Roman and H. P. A. Lensch. Automatic multiperspective images. In *Eurographics Symposium on Rendering*, pages 83–92, 2006.

[16] S. M. Seitz and J. Kim. Multiperspective imaging. *IEEE Comput. Graph. Appl.*, 23, November 2003.

[17] R. Szeliski and H.-Y. Shum. Creating full view panoramic image mosaics and environment maps. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 251–258, 1997.

[18] A. Zomet, D. Feldman, S. Peleg, and D. Weinshall. Mosaicing new views: The crossed-slits projection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25:741–754, 2003.

[19] A. Zornet, S. Peleg, and C. Arora. Rectified mosaicing: mosaics without the curl. In *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, volume 2, pages 459 –465 vol.2, 2000.



(a)

(b)

Figure 5: Comparison of our proposed technique (top) with the MAD algorithm (bottom) [13]. Our system produces results of comparable quality to the state of the art, while obtaining a speedup factor of up to an order of magnitude.

(a)

(b)

Figure 6: A second comparison of our technique (top) against the MAD algorithm (bottom), on a different and more challenging part of the Hillel sequence. Note that the MAD panorama appears curved, while our panorama remains undistorted due to the constrained alignment and robust optical flow estimation. In addition, note that some forground objects, such as the trees, in Figure 6(b) are misaligned and truncated, whereas our system is able to better handle these objects.

Figure 7: Results of our system on the **Train** sequence.

Figure 8: Results of our system on the **Boat** sequence.