Mosaic of Near Ground UAV Videos Under Parallax Effects

Mohamed A. Helala, Luis A. Zarrabeitia, Faisal Z. Qureshi
Faculty of Science, University of Ontario Institute of Technology, Oshawa, ON, Canada
{Mohamed.Helala, Luis.Zarrabeitia, Faisal.Qureshi}@uoit.ca

***Abstract*—This paper explores the exciting possibility of using Google Earth as a software laboratory for studying wide-area scene analysis using near-ground aerial imagery. To this end we present a new image mosaicing algorithm capable of generating large mosaics from imagery captured by a near-ground aerial vehicle. Our algorithm eschews camera calibration and can handle strong parallax effects visible in the captured imagery. The imagery is generated by simulating an aerial vehicle flying over the New York city within the Google Earth environment. We also evaluate the proposed approach on a real dataset captured by a physical aerial vehicle, demonstrating that the algorithm that was initially developing using synthetic imagery does indeed work on real data.**

I. Introduction

The ability to curate and analyze imagery at the global scale has been pivotal in designing visually rich Geographic Information Systems (GISs), such as the Google Earth, that contain 3D models of major metropolitan areas of the world. These models are painstakingly constructed using, among other sources of information, imaging data captured through aerial vehicles. It turns out that visually rich GISs are also a valuable source of data when studying large scale image analysis systems. With this in mind this paper uses synthetic imagery generated using Google Earth and develops a new method for mosaic construction from near ground aerial imagery. The paper also demonstrates the proposed method on a real dataset captured by a physical UAV flying over a city.

Unmanned Aerial Vehicles (UAVs) have gained a lot of attention lately. Their ability to survey and observe large areas, including regions which are not easily traversable, make them ideally suited for applications, such as search and rescue, forest fire monitoring, forest biomass estimation, agricultural information gathering, land-use changes monitoring, etc [1], [2]. Images collected by UAVs need to be registered with each other to get a coherent picture of the area under observation. This is akin to generating an image mosaic given a set of images taken at different viewpoints. For UAVs flying at low altitude over an uneven terrain that exhibit strong variation in depth, say a city block, parallax effects must be accounted for when constructing a mosaic from the captured images.

Several techniques have been proposed for constructing image mosaics [3], [4], [5]. Typically these either assume negligible parallax or a highly constrained camera motion model or both. Recently [5], [6] proposed an image mosaicing technique that compensates for weak parallax by assuming a dominant translation camera motion model. These techniques can only handle translational camera motion and small variations in camera-scene distances. Both of these assumptions do not hold in our situation.

The work presented in this paper relaxes these assumptions and is able to handle arbitrary camera motions and strong parallax effects. The proposed technique begins by estimating camera motion and calibration parameters from a sequence of images. Next this information is used to project all the images into a common viewpoint, accounting for depth and occlusion properties of every pixel in every captured image. Specifically we employ a plane sweep algorithm to project different images at different depths. An energy minimization step processes the images projected at different depth levels and constructs the final mosaic by assigning the best depth value to each pixel given the selected viewpoint.

The work presented here makes the following contributions: 1) it develops a different formulation for the plane sweep algorithm that can handle arbitrary models in contrast to [5], [6], which assume a translational motion model; 2) unlike [5], [6], the proposed method can construct mosaics from low framerate video sequences, as long as there is some overlap between frames; and lastly, we propose using Google Earth GIS as a software laboratory for studying aerial image analysis algorithms.

The rest of the paper is organized as follows: Section II discusses the related work. The next section describes our methodology. Results are provided in Section IV and we conclude the paper with Section V.

II. Related Work

Steedly *et al.* present a taxonomy of video mosaics and their applications [7]. They divide video mosaics into four classes: 1) static mosaics, 2) dynamic mosaics, 3) temporal mosaic pyramids, and 4) multi-resolution mosaics. The work presented in this paper is closest to the first category (static mosaics). A static mosaic divides a video sequence into a set of shots. Each shot consists of overlapping images. A mosaic is created for each shot. These mosaics are then aligned towards a reference shot.

Static video mosaic construction typically consists of three steps: frame registration, frame integration, and illumination and parallax compensation. For a good introduction to image registration algorithms, we point the kind reader to [8], [9].

Several techniques have been proposed for constructing static video mosaics. [3] divides video frames into *key frames*

or *intermediate frames* depending upon the amount of overlap between two successive frames. The overlap between the successive frames is determined through keypoint (feature) matching between the two frames. The frames are registered to construct the final mosaic by relying upon the orientation homographies that are estimated using a subset of the keypoints.

The approach described in [4] addresses the problem of global mosaic registration and super-resolution. This technique constructs a graph which has registered frames as *vertices* and initial homographies between successive frames as *edges*. The initial homography associated with a node is determined by chaining the homographies along the least-cost path to a common reference frame. The constructed mosaic is enhanced by applying a super-resolution step that replaces each pixel in the mosaic by the nearest color pixel from the corresponding frame.

Previous techniques assume motion parallax to be negligible (i.e, the scene is nearly planar). Motion Parallax [6] is defined as the shifting of scene objects with respect to the background due to camera motion. Parallax can introduce incorrect alignment of the overlapping images, object repetition or ghost effects (see [6]), thereby reducing the quality of the generated mosaic.

Recent advances try to solve the parallax problems in several ways. Hsu and Tsan [10] perform a global motion estimation to stitch images. There technique starts by segmenting the image into a number of blocks and detecting a motion vector for each block. A general motion vector is determined and the blocks that have significant deviation are identified as foreground objects and subsequently eliminated from the final mosaic. Keypoints are then extracted from the background regions and used to refine the global motion vector and to construct the final mosaic. Results show that the global motion estimation can accumulate stitching errors which degrade the quality of the final mosaic.

Zhi and Cooperstock [5] use virtual dense sampling to enhance the mosaic by addressing problems that arise due to motion parallax. The technique starts by having several stationary cameras looking at a scene. These cameras form a long baseline multi-camera system. The source cameras are initially calibrated with respect to the view. A virtual camera location is defined by interpolating the physical source cameras and a virtual mosaic image is constructed from the input by estimating depth information using a plane sweep algorithm.

As stated somewhere else, aerial imagery captured by UAVs is useful, and indeed essential, for many applications. Many of these applications rely on the ability to combine images captured by one or more UAVs into a single high-def, wide-FOV image. Consequently, image mosaicing algorithms provide a natural starting point to analyze aerial imagery. [1] and [2], for example, construct a mosaic from images taken from multiple UAV vehicles and incrementally generate an overview image. Local mosaics constructed aboard each UAV are transmitted to the ground station, which is responsible for constructing the global mosaic. This technique was demonstrated on an aerial camera network monitoring a fire drill. The images captured through the cameras were combined to construct a single mosaic that provided an overall picture of the scene.

[11] investigates the use of miniature UAVs (MAVs) for aerial monitoring and surveillance. It operates upon synchronized videos and telemetry data collected from these MAVs. A video mosaic is generated by extracting and tracking image corners and using RANSAC to handle outliers. The generated mosaic is geo-referenced by geo-locating multiple ground points based on MAVs' pose estimates. Motion parallax was also investigated in [12] for building stereoscopic mosaics for far ground scenes by seamless registration of images. This technique uses an airborne camera and assumes a predetermined translational dominant camera trajectory.

In summary the existing techniques either assume a known camera motion model (typically translational motion model) or ignore motion parallax effects that are dominant when the scene exhibit large depth variations. Our work aims to address these two issues.

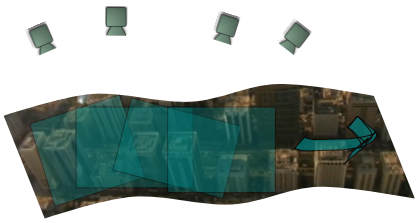
III. Methodology

Our technique begins by projecting the set of captured images onto a common image plane, assumed to be defined by a virtual camera (see Figure 1). The technique computes a depth value for each pixel and the pixels are projected onto the common plane taking into account their respective depth values, thereby avoiding the artifacts due to motion parallax. In order to compute the depth value of a pixel, the proposed approach uses information from the set of images containing that pixel and assuming 1) a reasonably smooth depth variations in local regions and 2) color invariance. A similar scheme has been previously explored in [10].

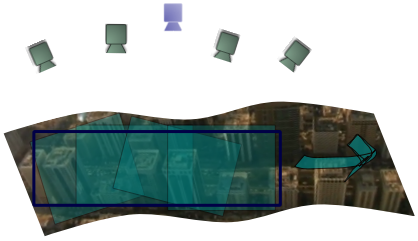
A. Camera Calibration

Our strategy requires sufficiently precise estimates for the internal and external camera parameters for each view. If the source images come from, say a UAV, it can be expected that the calibration data can be obtained alongside the image data, measured by the onboard sensors such as the GPS and gyroscopes. In general, however, we can assume that high-quality calibration data is *not* available alongside the captured image stream. It is, therefore, necessary to estimate (from scratch) or improve the camera calibration.

Automatic camera calibration has been an active area of research for the last two decades. [13] presents an effective method to simultaneously estimate, up to a scale-ambiguity, 1) the intrinsic and extrinsic camera parameters, 2) the lens distortion coefficients and 3) the 3D coordinates of the features, given point correspondences between every pair of frames. We follow a similar approach. The main difference being the assumption that we deal with a single smooth video, which defines an ordering between the frames, and that the intrinsic camera parameters remain constant. In both cases, the point correspondences serve to set up a structure-from-motion



(a) Real cameras



(b) Virtual camera

Fig. 1: Virtual view reconstruction: (a) A moving camera, or a set of cameras, capture different, overlapping regions of the scene. (b) A virtual camera is placed near the center of the captured scene and a virtual view is computed as if taken by the virtual camera. The virtual view will be wider than the individual images. The plane sweep algorithm allows us to resolve the ambiguities introduced by the motion parallax in the overlapping regions. The blue rectangle represents the virtual camera view.

problem, which can be solved with a Levenberg-Marquardt solver [14].

In the following discussion, we will assume that a good estimation for the calibration parameters is given. Furthermore, we will assume that there is no lens distortion. This is a fair assumption, as the distortion coefficients estimated by the bundle adjustment process described in the previous paragraph can be used to undo the lens distortion.

B. Virtual View Generation

Given camera positions and the intrinsic camera parameters, we generate a scene depth map using a technique similar to [15], which uses plane sweep algorithm to obtain a depth map with respect to an intermediate virtual viewpoint. Unlike [6], however, our approach does not require the images to be rectified, enables it to ignore the coplanarity assumption when considering more than two images simultaneously.[1]

Let $\{I_i\}$ be the set of images, $\{P_i = K_i\,(R_i|\mathbf{t}_i)\}$ be the corresponding set of known cameras, $P_v = K_v\,(R_v|\mathbf{t}_v)$ the virtual camera, and I_v the image we want to obtain. For each (signed) distance d, there is exactly one plane $\mathbf{n}_d$ parallel to the image plane of P_v at distance d from the camera center $\mathbf{c}_v = -R_v^T\mathbf{t}_v$. If we assume that the scene captured by P_i lies on the plane $\mathbf{n}_d$, then the points $\mathbf{p}_v$ of I_v are related to the points $\mathbf{p}_i$ of I_i via the plane transfer homography

$$\mathbf{p}_i = H_i(d)\mathbf{p}_v, \tag{1}$$

where

$$H_i(d) = K_i\left(R_i' - \frac{\mathbf{t}_i'\mathbf{n}^T}{d}\right)K_v^{-1}. \tag{2}$$

Here $\mathbf{n}^T = (0,0,-1)$, $\mathbf{n} \perp \mathbf{n}_d$ and R_i', $\mathbf{t}_i'$ are the camera parameters R_i, $\mathbf{t}_i$ in the reference frame of P_v given by

$$R_i' = R_iR_v^T \text{ and } \mathbf{t}_i' = -R_iR_v^T\mathbf{t}_v + \mathbf{t}_i. \tag{3}$$

Evidently the assumption that all the scene objects lie in the plane $\mathbf{n}_d$ does not hold except for the simplest scenarios, such as when the objects are so far from the cameras that the parallax can be ignored. In the general case, the only pixels in I_v that satisfy (1) are those corresponding to objects that actually lie on $\mathbf{n}_d$ and are not occluded from $\mathbf{c}_v$. Thus, if we apply the homography $H_i(d)^{-1}$ to the image I_i, the resulting image will contain at most an unknown and possibly empty set of pixels with the correct color. As d changes, the set of correct pixels will also change.

Because this set of pixels, albeit unknown, is the same regardless of the source image I_i, if we project all the images I_i into the camera P_v using the homographies (1) for the same distance d, the objects lying in the plane $\mathbf{n}_d$ will appear to be focused in the combined image, while the rest of the objects will appear blurred. This can be seen with two source images in Figure 2. At depth $d = 30$, the green bottle appears to be focused, while the checkerboard in the background appears blurred. At depth $d = 60$, the checkerboard is on focus, but the rest of the scene looks blurred.

The color of each pixel for the final image I_v will be selected from the depth d at which the combined image is less blurred around that pixel, trying to avoid sudden changes of depth between neighbouring pixels. The depth assignment strategy will be discussed in Section III-D. Also note that the homography (2) maps from the virtual to the real image planes.

It is generally unnecessary to use the inverse homography, and even the intermediate depth images shown in figure 4 (c)-(f) do not need to be explicitly computed. We still need to select the parameters R_v and $\mathbf{t}_v$. While the previous reasoning applies regardless of the selection of R_v and $\mathbf{t}_v$, for the mosaicing objective it makes sense to maximize the area of I_v that is colored. One suitable option is to choose the average of the translations and rotations:

$$\mathbf{t}_v = \frac{1}{n_s}\sum_{i=1}^{n_s}\mathbf{t}_i \text{ and } \mathbf{r}_v = \frac{1}{n_s}\sum_{i=1}^{n_s}\mathbf{r}_i. \tag{4}$$

For the dataset used in Figure 4, we selected one of the images as ground truth for comparison purposes: the image was excluded from the computations and its corresponding camera parameters were used as the virtual camera.

[1]Coplanarity assumption means that the image planes of two cameras are coplanar.

(a) Depth =30 (b) depth =60

Fig. 2: Two different intermediate virtual planes that show different focus for scene objects.

C. Depth Range Estimation

Previous techniques did not provide an automatic way to estimate the depth range $[d_{min}, d_{max}]$. For example the method proposed in [10], [15] use equal spacing between planes in the sweep direction, which limits its applicability. This can result in undesired artifacts as the sweep plane may never be at the exact focus depth of some pixels. In order to handle the depth range problem, we leverage the bundle adjustment techniques mentioned in Section III-A to determine an estimated depth range from the set $\mathbf{X} = \{\mathbf{X_i} = (x_i, y_i, z_i, 1)^T\}$ of the 3D points. Without the loss of generality, we can assume that the points in $\mathbf{X}$ are expressed in the coordinate frame of the camera P_v, so the coordinates z_i encode depths. Then we can choose the range $[d_{min}, d_{max}]$ as

$$d_{\min} = \min_{\mathbf{X}_i \in \mathbf{X}} z_i \text{ and } d_{\max} = \max_{\mathbf{X}_i \in \mathbf{X}} z_i. \tag{5}$$

Instead of defining equal depth spacing for plane sweep algorithm, we dynamically construct the depth values by iteratively subdividing the depth range $[d_{\min}, d_{\max}]$. This process stops when the homographies induced by new subdivisions do not produce significant displacement in the virtual plane.

D. Generating Mosaic Depth Map

In order to generate the depth map for the virtual camera image we assign to each pixel of the output image a depth label corresponding to the plane in which it has the correct focus. We view depth map construction as an assignment problem, where a single depth label from a set of depths L is assigned to each pixel in the set of Π of output pixels. Consequently we define a depth label assignment $f : \Pi \rightarrow L$ for each output pixel. We use multi-label Graphcut optimization [16], where the labeling energy is defined using two terms: the data cost energy E_d and the smoothness cost energy E_s. The total energy of label assignment is defined as

$$E(f) = E_d(f) + E_s(f), \tag{6}$$

where

$$E_d(f) = \sum_{p \in I_v} A(p, f(p)) \text{ and} \tag{7}$$

$$E_s(f) = \sum_{\substack{p \in I_v \\ q \in N_p \\ d(p) \neq d(q)}} B(p, q, f(p), f(q)). \tag{8}$$

Here $A(p, f(p))$ is a function that penalizes the depth assignment $f(p)$ of p if the set of colors corresponding to the projections of p into the source image, according to the homography (1), has a large variance, and $B(p, q, f(p), f(q))$ favors that the neighbouring pixels $p, q \in I_v$ are assigned similar depth values $f(p), f(q)$.

The results shown in Figure 4 use the data penalty function

$$A(p, d) = \alpha \sum_i (\bar{\rho}_{p,d} - \rho(C_{i,d}(p)))^2, \tag{9}$$

where $C_{i,d}(p)$ is the color of the pixel in I_i corresponding to p through the homography (1), ρ is a summarization function, $\bar{\rho}_{p,d}$ is the average of color values, and the summations are only considered over the set of images I_i for which the projection of p lie on its interior. This function computes the centroid of the colors of the pixels corresponding to p at depth d and penalizes quadratically the colors that are away from the centroid. The summarization function ρ maps from the space of colors to $\mathbb{R}$. Both converting the pixel to gray-scale and the norm of the *RGB* value produced consistently bad results in our experiments. In the results of figure 4, we used the hue value from the *HSV* color space, which led to considerably better results.

For the smoothness cost function, we use

$$B(p, q, d_p, d_q) = \beta \sum_i B_i(p, q, d_p, d_q) \tag{10}$$

$$B_i(p, q, d_p, d_q) = \begin{cases} 0 & \text{if } p \text{ or } q \text{ back-project to an edge of } I_i \\ (d_p - d_q)^2 & \text{otherwise.} \end{cases} \tag{11}$$

This function penalizes a depth difference between neighboring pixels in I_v, unless one of the pixels corresponds to an edge in the source images. As before, the summation is only considered over the source images that contain the projection of p. The parameters α and β can be tweaked to favor minimizing the smoothness over the data energy, or vice-versa.

After performing graphcut optimization and obtaining the optimal labelling, each pixel p in the output will have the color of the corresponding pixel in the intermediate image plane at depth label $d(p)$.

If some output pixels have a large matching energy, they are indicated as occluded regions and left as holes in the output. In order to fill these holes, the output virtual image is scanned row by row. When encountering a hole region, the depths of the leftmost and rightmost pixel surrounding the hole are considered. The hole is filled from pixels of the nearest of these two values. This corresponds to the assumption that the hole is occluded by closer objects.

IV. Experimental Results

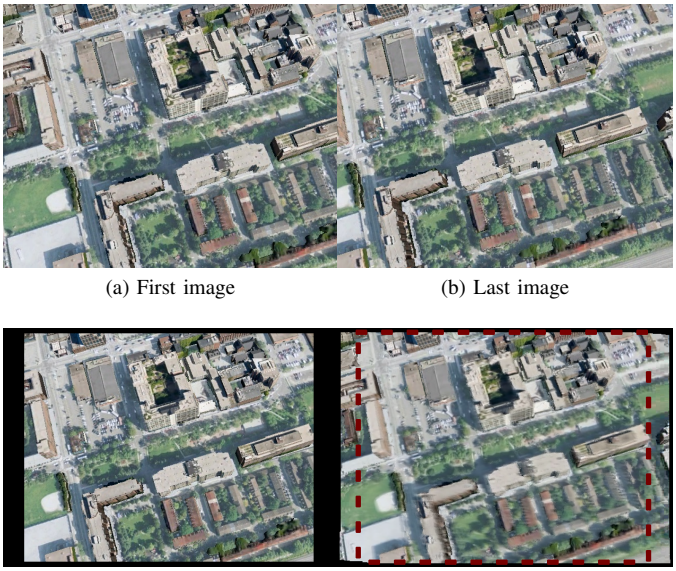
We tested our algorithm on both synthetic (generated using Google Earth) and real datasets. Figure 3 shows results on

a synthetic dataset, while Figure 4 shows results on a real dataset.

The following table lists the parameters used for each dataset. Figures 3a and 4a show the initial images for each sequence. Figures 3b and 4b, on the other hand, show the last images for the two sequences, respectively. In both cases, the middle image of the dataset was selected as ground truth (Figures 3c and 4g) and excluded from the computations. Figures 3d and 4h show the final reconstructed mosaic. Additionally, to illustrate the plane sweep algorithm, Figures 4c-4f show the combined images at some intermediate depth levels.

Seq	#images	d_{min}	d_{max}	# intermediate depths
Simulated	10	45	64	18
Real	15	26	60	36

TABLE I: Parameters values for our tests.



(a) First image (b) Last image

(c) Ground truth (d) Final mosaic

Fig. 3: Mosaic using synthetic imagery from Google Earth. (a) First and (b) last images of the sequence. (c) Ground truth. (d) Final mosaic. For comparison, the dashed line highlights the region of the mosaic corresponding to the ground truth image.

V. Conclusions

In this paper we explore using Google Earth as a software laboratory for designing image analysis systems. Specifically, we present a technique for constructing mosaics from videos captured by near ground UAVs, remembering that such aerial videos exhibit strong parallax effects. The proposed method is evaluated on synthetic imagery generated using Google Earth and on a real UAV video sequence, which also exhibit strong parallax effects. The experimental results presented here appears promising. An important aspect of our approach is that we can work with low frame rate cameras as long as there is enough overlap between successive frames. Moreover the proposed method can work with several motion models as long as the camera parameters can be estimated. One possible line of work to improve the quality of the generated mosaic is to make better use of the edge information from the source images. More research is needed in that direction.

References

[1] D. Wischounig-Strucl, M. Quaritsch, and B. Rinner, "Prioritized data transmission in airborne camera networks for wide area surveillance and image mosaicking," in *in Proc. IEEE CVPR*, vol. 1, Colorado Springs, USA, June 2011, pp. 692–698.

[2] M. Quaritsch, R. Kuschnig, V. Mersheeva, D. Wischounig-Strucl, S. Yahyanejad, E. Yanmaz, G. Friedrich, H. Hellwagner, C. Bettstetter, and B. Rinner, "Collaborative uavs for aerial reconnaissance in rescue scenarios," in *in Proc. Austrian Robotics Workshop*, Innsbruck, Austria, May 2011.

[3] D. Steedly, C. Pal, and S. Szeliski, "Efficiently registering video into panoramic mosaics," in *in Proc. ICCV*, vol. 2, Beijing, China, Oct. 2005, pp. 1300–1307.

[4] R. Marzotto, A. Fusiello, and V. Murino, "High resolution video mosaicing with global alignment," in *in Proc. IEEE CVPR*, vol. 1, Los Alamitos, CA, USA, June 2004, pp. 692–698.

[5] Q. Zhi and J. R. Cooperstock, "Toward dynamic image mosaic generation with robustness to parallax." *IEEE Transactions on Image Processing*, vol. 21, no. 1, pp. 366–378, 2012.

[6] Q. Zhi, "Towards dynamic mosaic generation with robustness to parallax effects," in *PhD thesis*. McGill University, 2009.

[7] M. Irani, P. Anandan, and S. Hsu, "Mosaic based representations of video sequences and their applications," in *in Proc. IEEE ICCV*, Boston, MA, USA, Jun. 1995, pp. 605–611.

[8] R. Szelisk, "Video mosaics for virtual environments," *IEEE Comput. Graph. Appl.*, vol. 16, no. 2, pp. 22–30, Mar. 1996.

[9] H. Wallin, C. Christopoulos, and F. Furesjo, "Robust parametric motion estimation for image mosaicing in the mpeg-7 standard," in *in Proc. ICIP*, vol. 2, Thessaloniki, Greece, Oct. 2001, pp. 961–964.

[10] C. ting Hsu and Y. chun Tsan, "Mosaics of video sequences with moving objects," *Signal Processing: Image Communications*, vol. 19, pp. 81–98, January 2004.

[11] C. Taylor and E. Andersen, "An automatic system for creating geo-referenced mosaics from mav video," in *in Proc. IEEE/RSJ IROS*, Nice, France, Sept. 2008, pp. 22–26.

[12] Z. Zhu, A. Hanson, and E. Riseman, "Generalized parallel-perspective stereo mosaics from airborne video," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 2, pp. 226–237, Jan. 2004.

[13] N. Snavely, S. Seitz, and R. Szeliski, "Modeling the world from internet photo collections," *International Journal of Computer Vision*, vol. 80, pp. 189–210, 2008, 10.1007/s11263-007-0107-3. [Online]. Available: http://dx.doi.org/10.1007/s11263-007-0107-3

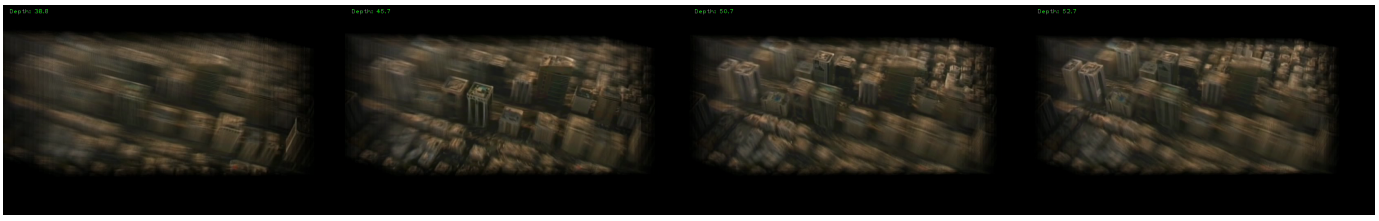
[14] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery, *Numerical Recipes in C*, second ed. ed. Cambridge University Press, 1992, ch. Chapter 15.

[15] I. Geys, T. P. Koninckx, and L. V. Gool, "Fast interpolated cameras by combining a gpu based plane sweep with a max-flow regularisation algorithm," in *in Proc. 3DPVT, 2nd Int. Symp.*, Washington, DC, USA, Sept. 2004, pp. 534–541.

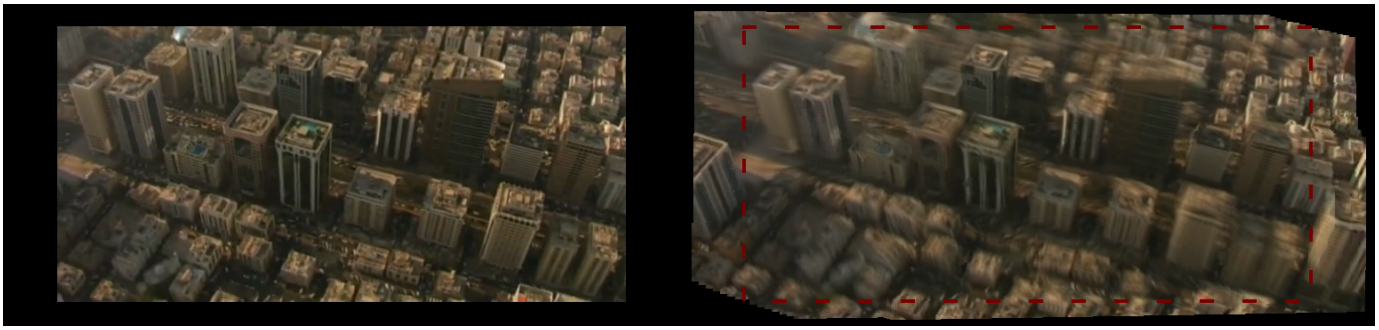
[16] A. Delong, A. Osokin, H. N. Isack, and Y. Boykov, "Fast approximate energy minimization with label costs," *Int. J. Comput. Vision*, vol. 96, no. 1, pp. 1–27, Jan. 2012.

(a) First image

(b) Last image



(c) d = 38.8 (d) d = 45.7 (e) d = 50.7 (f) d = 52.7



(g) Ground truth (h) Final mosaic

Fig. 4: Mosaic using a real dataset. (a) First and (b) last images of the sequence. (c)-(d) Virtual planes at different depths, illustrating the plane sweep algorithm. Note how different objects come into focus at different depth values. (g) Ground truth. (h) Reconstructed mosaic from a virtual view. The virtual camera had the same position and orientation as (g), but (g) was not used for this reconstruction. As in figure 3, the dashed line highlights the region of the mosaic corresponding to the ground truth image.