# 3D Urban Reconstruction from Wide Area Aerial Surveillance Video

Zhuoliang Kang and Gérard Medioni
University of Southern California
Los Angeles, CA 90089
{zkang|medioni}@usc.edu

## Abstract

*We propose an approach to solve camera pose estimation and dense reconstruction from Wide Area Aerial Surveillance (WAAS) videos captured by an airborne platform hovering around the urban scenes. Our approach solves them in an online fashion: it incrementally updates a sparse 3D model as well as a dense 2.5D Digital Surface Model (DSM) as each new frame arrives; the camera pose of each new frame is estimated using Perspective-n-Point (PnP) method with 2D-3D image-model feature matches. Dense optical flow between successive frames computed after a step of 2-D stabilization is used to guide the feature matching between each new frame and the maintained sparse 3D model.*

*Our approach provides an online solution for camera pose estimation and dense reconstruction, and is significantly faster than the latest batch methods. The camera poses are estimated as accurately as with global Bundle Adjustment without drift along the path. We also produce a highly-detailed full 3D model via volumetric integration. Experiments on both synthetic and real-world datasets validate its performance.*

## 1. Introduction

3D urban reconstruction from imagery is essential for various applications, such as urban planning and virtual city tours. For wide area aerial surveillance (WAAS) specifically, an up-to-date 3D model is helpful to solve occlusion problems for many tasks, such as vehicle tracking [20] and traffic inference [27]. The complete 3D model can be further used to remove redundancy in the video stream and compress the data dramatically, which is very important with the limited power and computation capability on the airborne platform. To achieve these goals, both accuracy and efficiency of the reconstruction process are required.

Two core problems need to be addressed in the process: camera pose estimation, *i.e*. the estimation of $6DOF$ camera poses (position and orientation) for each image; and dense reconstruction, *i.e*. the establishment of a dense 3D model. In state-of-the-art works, most approaches solve these two problems sequentially: feature-based *Structure-from-Motion* (SfM) estimates camera poses based on a sparse set of matched 2D feature points, such as SIFT [15]. After obtaining all camera poses, a multi-view stereo algorithm is used to generate a dense 3D model. In feature-based SfM, camera pose drift is a critical issue due to the errors accumulated along the path. To reduce drift, a global Bundle Adjustment [22] is required. It involves a non-linear optimization over many or all camera poses and feature point positions, which makes it time-consuming when the number of frames is large.

Figure 1. An example of the 3D urban scene and the camera motion path of a WAAS video.

WAAS videos are captured in high-resolution by an airborne platform hovering around the target scenes following a simple camera motion path (Figure 1). They thus come with large overlap between all frames which allows for detailed 3D urban reconstruction that is both accurate and efficient. In this paper, we propose an **online** approach to solve camera pose estimation and dense reconstruction from WAAS videos. A sparse 3D model and a dense 2.5D Digital Surface Model (DSM) are incrementally updated during the process. Our approach estimates the camera pose at each frame while updating these two models as each new frame arrives. Different from traditional SfM solely based on matched 2D points, the camera pose of each new frame is estimated using *Perspective-n-Point* (PnP) method based on 2D-3D Image-Model feature matches. Dense optical flow

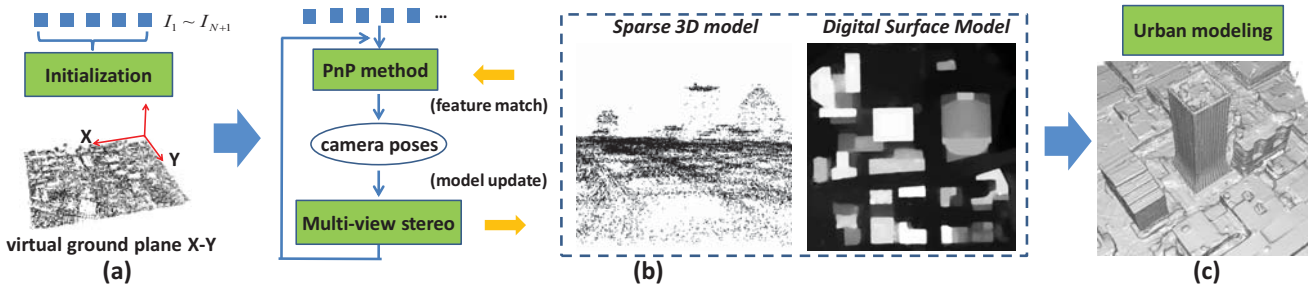



Figure 2. Pipeline overview. (a) initialization. (b) camera pose estimation and model update using multi-view stereo. (c) urban modeling.

between successive frames computed after a step of 2-D stabilization is used to guide the feature matching between each new frame and the maintained sparse 3D model. We also produce a highly-detailed full 3D model via volumetric integration. The pipeline of our approach is illustrated in Figure 2. Experiments on both synthetic and real-world datasets validate the performance of our approach. In summary, we offer the following contributions:

- **online solution:** camera pose estimation and dense reconstruction are solved within an online pipeline, which estimates camera poses at each frame while updating the models during the process.
- **efficient framework:** without global Bundle Adjustment, our online approach is significantly faster than the latest batch methods.
- **accurate camera pose estimation and detailed urban modeling:** camera poses are estimated as accurately as with global Bundle Adjustment without drift along the path. We also produce a highly-detailed full 3D urban model.

## 2. Related work

There are mainly two groups of related works: camera pose estimation and urban reconstruction.

**Camera pose estimation**: In feature-based SfM, the camera poses as well as the 3D positions of a sparse set of feature points are unknown and estimated. Relative pose methods, such as 5-point method [17], are used to generate initial camera poses. A global Bundle Adjustment involving all images is then used to refine the results. The state-of-the-art work is proposed by Furukawa and Ponce [7], which iteratively refines the camera poses and processes patch-based reconstruction. Another interesting SfM work is proposed by Hongdong Li [12], where geometric structure is estimated based on graph rigidity theory without explicit estimation of camera motion. However, this algorithm cannot scale to problems with a large number of feature points.

Several camera tracking frameworks have been proposed that can determine the camera pose of each new frame in real-time. Klein and Murray [10] proposed Parallel Tracking And Mapping (PTAM), which is a real-time camera tracking system based on key frames using sparse feature points and reasonable implementation of Bundle Adjustment. Newcome *et al*. presented Dense Tracking And Mapping (DTAM) [16], a system for real-time camera tracking and dense reconstruction. These works are designed specifically for indoor virtual reality applications. A complete dense 3D model is out of their scope. Wendal *et al*. [23] proposed a work for live dense reconstruction which is designed specifically for micro aerial vehicles.

**Urban reconstruction**: Volumetric methods split the space into voxels and estimate a binary or probability occupancy status for each voxel. In [18], each voxel is associated with an occupancy probability and an appearance model. In [4], probabilistic volumetric modeling is combined with smooth signed distance surface estimation to generate surfaces for urban scenes. Point-based methods generate a dense point cloud by estimating 3D position for pixels in the input images. Structural properties of urban scenes are often used to constrain the problem. In [6], scenes are assumed to be composed of piecewise-planar surfaces with dominant orientations. There are also works [13, 19] that constrain the problem with piecewise planar prior without dominant orientations. Graph cuts [6] and dynamic programming [13] are usually used to find the stereo correspondences. Patch/mesh-based methods have also been proposed with capability to do urban reconstruction. In [8], a patch-based stereo algorithm is proposed to generate dense models via image patch matching and expansion. Hiep *et al*. [9] proposed a pipeline for large-scale and detailed reconstruction, in which a photometric consistent mesh is generated from the initial dense point cloud with variational optimization.

## 3. Description of our approach

### 3.1. Overview

Our approach uses a WAAS video sequence as input. The pipeline is composed of 4 parts: **1. Initialization** (Figure 2 (a)): the camera poses of the first $N + 1$ frames as

well as a virtual ground plane are estimated as initialization. **2. Multi-view stereo and model update** (Figure 2 (b)): two 3D models are maintained and incrementally updated during the process: a sparse 3D model composed of SIFT feature points, which is used for camera pose estimation, and a dense 2.5D Digital Surface Model (DSM) which is used for occlusion handling. When new frames arrive, our approach estimates the depth maps using multi-view stereo and updates the sparse 3D model as well as the DSM. **3. Camera pose estimation** (Figure 2 (b)): the camera pose is estimated at each frame using Perspective-n-Point (PnP) method with 2D-3D Image-Model feature matching. **4. Urban modeling** (Figure 2 (c)): a full 3D model is generated using volumetric integration method. In the rest of this section, we explain each part in detail.

## 3.2. Initialization

We estimate the camera poses of the first $N+1$ frames using standard Structure-from-Motion (SfM) algorithm as initialization. $N$ is the number of neighboring frames used in multi-view stereo algorithm, which is described in section 3.3.1. The choice of a proper $N$ relies on the baseline between successive frames. In all experiments presented in this paper, we use $N = 20$. Based on the sparse point cloud generated from the initial SfM algorithm, we also aim to find a virtual ground plane in terms of which the elevation values in the DSM are defined. Our approach estimates a virtual ground plane $G$ using Principal Component Analysis (PCA) on the obtained sparse point cloud (Figure 2 (a)). The normal direction is determined as the direction with minimum variance. For WAAS videos of urban scenes, a reasonable plane $G$ can be found using the above strategy since the height range is very small compared with the range spanned in the ground plane. Our approach still works well when the plane is virtual, *e.g.*, in mountainous terrain case, as shown later in section 4.2.

## 3.3. Multi-view stereo and model update

As shown in Figure 2 (b), our approach maintains two models and updates them when new frames arrive: a sparse 3D model $M_S$ composed of SIFT feature points and a 2.5D Digital Surface Model $M_D$. When a new frame and its neighboring frames arrive, we compute its depth map using multi-view stereo as described in section 3.3.1. Then, we update $M_S$ and $M_D$ as described in section 3.3.2. The camera pose of each new frame is estimated as described in section 3.4, except for the first $N+1$ frames whose camera poses are estimated in the initialization step.

It is not necessary to update the models at each frame due to the small variance between successive frames in video sequences. In practice, we update the models on key frames only. A new frame is labeled as key frame when the feature matches between this frame and the maintained sparse 3D model $M_S$ is lower than a threshold. Details of the feature matching is described in section 3.4.1. In any case, we also add a key frame every 50 frames to keep the models up-to-date.

### 3.3.1 Multi-view stereo

Given camera poses of a key frame $I_k$ and its neighboring frames $I_i \in \mathcal{N}(k)$, we compute the height map $H_k$ for $I_k$ using multi-view stereo. Finding stereo correspondences for urban imagery is a difficult problem due to the homogeneous textures and massive occlusions. In our approach, we use the multi-view stereo algorithm proposed in DTAM [16]. Photometric evidence from multiple views as well as epipolar geometry are used to constrain the problem and reduce ambiguities. We minimize an energy function including a non-convex photometric error term and a convex regularization term:

$$\mathbf{E_h} = \sum_{\Omega} \lambda C(\mathbf{x}, \mathbf{h}(\mathbf{x}))d\mathbf{x} + \sum_{\Omega} \|\nabla \mathbf{h}(\mathbf{x})\|_{\epsilon} d\mathbf{x} \quad (1)$$

where $\mathbf{h}(\mathbf{x})$ represents the height of pixel $\mathbf{x}$ above the virtual ground plane $G$. $\Omega$ is the 2D image domain of $I_k$. $C(\mathbf{x}, \mathbf{h}(\mathbf{x}))$ represents the photometric error term measuring the average intensity error across its neighboring views:

$$C(\mathbf{x}, h) = \frac{1}{N} \sum_{i \in \mathcal{N}(k)} \left| I_k(\mathbf{x}) - I_i(\pi_i(\pi_k^{-1}(\mathbf{x}, h, G))) \right| \quad (2)$$

where $\pi_k^{-1}(\mathbf{x}, h, G)$ is the operator to compute the position of the 3D point projected from pixel $\mathbf{x}$ on $I_k$ when assigned to height $h$ above the virtual ground plane $G$, and $\pi_i(\pi_k^{-1}(\mathbf{x}, h, G))$ is the operator to compute the pixel location on $I_i$ back-projected from this 3D point. $\|\nabla \mathbf{h}(\mathbf{x})\|_{\epsilon}$ is a Huber norm regularization term used to smooth the generated height map while reserving boundary discontinuities at the same time. Different from a pure total-variation regularization term, the Huber norm allows smooth variance in small-scale to avoid the stair-casing effect.

To optimize Equation 1, we couple the photometric error term and regularization term with an auxiliary variable $\mathbf{h}'$ and optimize $\mathbf{h}$ and $\mathbf{h}'$ alternatively:

$$\mathbf{E}_{\mathbf{h},\mathbf{h}'} = \sum_{\Omega} \left\{ \lambda C(\mathbf{x}, \mathbf{h}(\mathbf{x})) + \|\nabla \mathbf{h}'(\mathbf{x})\|_{\epsilon} \right\} d\mathbf{x} + \sum_{\Omega} \frac{1}{2\theta} (\mathbf{h}(\mathbf{x}) - \mathbf{h}'(\mathbf{x}))^2 d\mathbf{x} \quad (3)$$

With $\mathbf{h}'$ fixed, we search exhaustively over a finite range of sampled height values to optimize the non-convex photometric error term of variable $\mathbf{h}$. Fixing $\mathbf{h}$, the optimization of $\mathbf{h}'$ can be achieved using a primal-dual approach. More details can be found in [16].

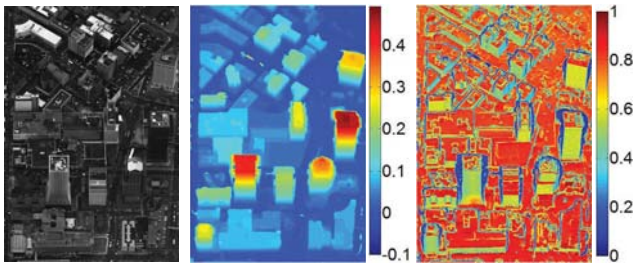



Figure 3. From left to right: key frame $I_k$, height map $H_k$ and confidence map $W_k$

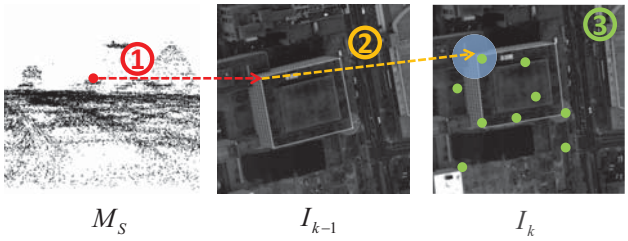



Figure 4. Illustration of feature matching between a new frame $I_k$ and the maintained sparse 3D model $M_s$

Our implementation is slightly different from the one used in DTAM, which optimizes over the inverse depth of each pixel. Instead, we optimize over the height above the virtual ground plane, since depth range varies significantly but height range is limited in urban scenes allowing efficient optimization with a limited number of samples. Furthermore, the height map tends to be a piece-wise linear filed with the planar structures in urban scenes, which fits the Huber norm smoothness term well. The height range as well as the virtual ground plane $G$ are estimated in the initialization step. A confidence map $W_k$ is also computed representing the confidence of the estimated height values in $H_k$:

$$W_k(\mathbf{x}) = \min(\cos(\xi(\mathbf{x})), 1 - C(\mathbf{x}, H_k(\mathbf{x}))) \quad (4)$$

where $\xi(\mathbf{x})$ is the angle between the estimated surface normal and optical ray to avoid grazing ramps. The intensity values of input frames are scaled to $[0, 1]$ such that the photometric error term $C(\mathbf{x}, H_k(\mathbf{x}))$ lies in the same range. An example is illustrated in Figure 3.

#### 3.3.2 Model update

$M_S$ is a sparse 3D point cloud composed of SIFT feature points used for camera pose estimation. Our approach updates $M_S$ after obtaining the height map $H_k$ and confidence map $W_k$ of a key frame $I_k$. For each SIFT feature point $\mathbf{x}$ detected on $I_k$, we compute the corresponding 3D point $X$ according to $H_k(\mathbf{x})$ if its confidence value $W_k(\mathbf{x})$ is larger than a threshold. The 3D positions of $X$ as well as the feature descriptors of $\mathbf{x}$ are then added into $M_S$. The confidence threshold is set as 0.7 in our experiments.

A digital Surface Model (DSM) is a proper representation to store the geometric information of urban scenes with values representing elevations above the virtual ground plane. Our approach also maintains a DSM $M_D$ as well as a corresponding confidence map $W_D$, and incrementally updates them at each key frame. For a 3D point projected from pixel $\mathbf{x}$, we use $\mathcal{X}$ to represent its projection on the virtual ground plane. Our approach updates the DSM using the following rules:

$$M_D(\mathcal{X}) = \frac{W'_D(\mathcal{X})M'_D(\mathcal{X}) + W_k(\mathbf{x})H_k(\mathbf{x})}{W'_D(\mathcal{X}) + W_k(\mathbf{x})} \quad (5)$$

$$W_D(\mathcal{X}) = W'_D(\mathcal{X}) + W_k(\mathbf{x}) \quad (6)$$

where $M'_D$ and $W'_D$ are the DSM and the confidence map before update at key frame $I_k$. In areas where multiple inconsistent height values are projected to the same location $\mathcal{X}$, *e.g.* building side-walls, we reset $M_D(\mathcal{X})$ using the maximum height. In the process of updating $M_S$ and $M_D$, we bilinearly interpolate $H_k(\mathbf{x})$ and $W_k(\mathbf{x})$ to achieve sub-pixel accuracy.

### 3.4. Camera pose estimation

We aim to estimate the $6D$ camera pose whenever a new frame arrives. Camera pose drift is a common problem when estimating camera poses sequentially, as error accumulates along the path. In standard SfM works, 2D-2D feature matches between images are used, with a global Bundle Adjustment involving all cameras implemented to address the drift problem. Our approach utilizes a different strategy to estimate the camera pose at each new frame using 2D-3D Image-Model feature matches. When a new frame $I_k$ arrives, 2D SIFT feature points in $I_k$ are matched with the 3D feature points stored in $M_S$. The camera pose of $I_k$ is then estimated using PnP method with RANSAC scheme. Using these 2D-3D Image-Model feature matches, the camera pose estimation of each frame is bundled with the maintained 3D model $M_S$, which effectively reduces drift.

#### 3.4.1 Optical flow guided feature matching

Feature matching is one of the most time-consuming step of standard SfM [24]. To make use of the small variance between successive frames in video sequences, we use dense optical flow to guide the feature matching between $I_k$ and $M_S$. The feature matching process is composed of 3 steps as illustrated in Figure 4.

**1. back-projection**: for each 3D feature point $X$ stored in $M_S$, we back-project it onto the previous frame $I_{k-1}$ and

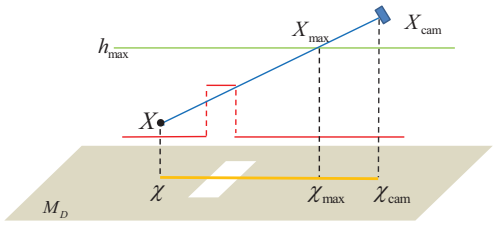



Figure 5. Illustration of occlusion handling when back-projecting a 3D feature point $X$ from $M_S$

obtain its projection $\mathbf{x}_{k-1}$. Occlusion is handled in this step as described in section 3.4.2.

**2. dense optical flow**: we compute the optical flow between $I_{k-1}$ and $I_k$ after a step of 2-D stabilization, as detailed in section 3.4.3. Then, the projection of $X$ on $I_k$ can be evaluated by adding the displacement to its projection on $I_{k-1}$:

$$\mathbf{x}_k = \mathbf{x}_{k-1} + \mathbf{u}^k_{k-1}(\mathbf{x}_{k-1}) \tag{7}$$

where $\mathbf{u}^k_{k-1}(\mathbf{x}_{k-1})$ represents the flow vector from $I_{k-1}$ to $I_k$ at point $\mathbf{x}_{k-1}$. In all experiments, the optical flow between the frames down-sampled to half size is accurate enough to provide a good approximation.

**3. local search**: we now search in a small region around $\mathbf{x}_k$ on $I_k$ to find the 2D SIFT feature match of $X$, which is very efficient. A feature match is found if the similarity is larger than 0.7, which is defined as the dot product of normalized feature descriptors. In all experiments, the search region is defined as a circle with radius of 3 pixels centered around $\mathbf{x}_k$. If multiple features on $I_k$ are found to match with $X$, we drop them all to avoid false matches with ambiguous features, *e.g.*, in regions with homogeneous texture.

### 3.4.2 Occlusion handling

Occlusion needs to be handled when back-projecting a 3D feature point $X$ from $M_S$ onto $I_{k-1}$. We make the assumption that no objects are floating, which is reasonable in urban environments. As illustrated in Figure 5, occlusion is handled by checking the occluding status along the optical ray ($X$,$X_{cam}$), where $X_{cam}$ is the camera center of $I_{k-1}$ with its projection $\mathcal{X}_{cam}$ on the virtual ground plane. To check the visibility of $X$ whose projection is $\mathcal{X}$ on the virtual ground plane, we only need to check the occluding status of line segment $(\mathcal{X}, \mathcal{X}_{cam})$ on the $M_D$: $X$ is visible if all height values on this line segment are known and do not exceed the occluding height at that location. Furthermore, with a maximum height prior $h_{max}$ of the target scene, we can find the highest possible occluding point $X_{max}$ on the optical ray with height $h_{max}$ and its projection $\mathcal{X}_{max}$ on the virtual ground plane. To check the visibility of $X$, we only need to check the occluding status of line segment $(\mathcal{X}, \mathcal{X}_{max})$ on $M_D$ instead.

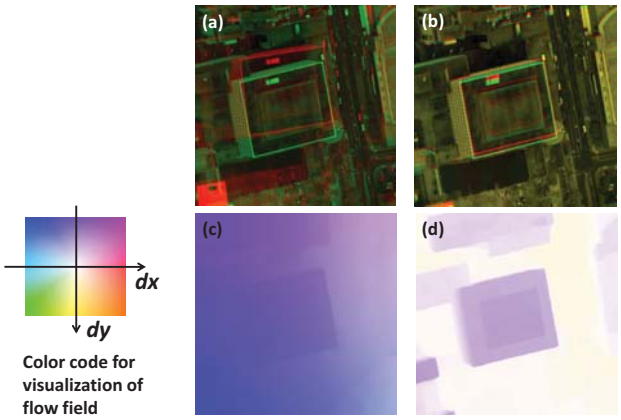



Figure 6. Illustration of frame pairs and flow fields. Consecutive frame pairs are shown as red-green anaglyph images. (a) original frame pair. (b) stabilized frame pair. (c) flow field between original frame pair. (d) flow field between stabilized frame pair.

### 3.4.3 Stabilized optical flow

The best and most recent method to compute optical flow is the Total-Variation $L_1$ (TV-$L_1$) optical flow as described in [26], which minimizes an $L_1$ norm data term and a total-variation regularization term:

$$\mathbf{E}_\mathbf{u} = \sum_\Omega (\lambda \left| I_i(\mathbf{x}) - I_j(\mathbf{x} + \mathbf{u}^j_i(\mathbf{x})) \right| + \left| \nabla \mathbf{u}^j_i \right|) d\mathbf{x} \tag{8}$$

However, applying it directly to the original frames does not give very good results because the flow is due to both camera motion and parallax (Figure 6 (c)), which does not necessarily fit the small motion assumption and piecewise linear constraint in Equation 8. Instead, we first stabilize successive frames in terms of a dominant plane using the homography estimated based on SIFT features with RANSAC scheme, and compute the optical flow between the stabilized frames. After stabilization, the displacement of an image point is directly proportional to its height above the dominant plane, and inversely proportional to its distance from the camera [11]. For aerial images, the displacement can be considered proportional to the height since the distance to the camera is very large. As shown in Figure 6 (d), the flow field between the stabilized frames is piecewise linear with small displacement proportional to the height, which fits the optical flow formulation well. Then, we compute the flow field between the original frames by compensating for the homography used for stabilization. The estimated dominant plane for stabilization between each pair of successive frames is not required to be either stable, or the same as the virtual ground plane $G$. In cases where a real dominant plane does not exist, *e.g.* mountainous terrain case, our approach still works as well.

| Dataset | Image Size | Key/All frames | Our approach | | | | Comparison | |
|---|---|---|---|---|---|---|---|---|
| | | | Init. | Pose. | Model update | Total | Global SfM | Speedup |
| Rochester | 4872×3248 | 8 / 400 | 20s | 2.0 s/fr. | 75s /key fr. | 43m 33s | 4h 13m 11s | 5.8 × |
| WPAFB 2009 | 4872×3248 | 8 / 300 | 20s | 2.4 s/fr. | 77s /key fr. | 32m 47s | 1h 23m 02s | 2.5 × |
| Providence (site22) | 1280×720 | 10 / 180 | 8s | 0.8 s/fr. | 8s /key fr. | 5m 47s | 23m 57s | 4.1 × |
| Synthetic terrian | 2048×2048 | 8 / 200 | 16s | 1.6 s/fr. | 38s /key fr. | 20m 11s | 1h 18m 05s | 3.9 × |

Table 1. Running times. **Init.:** time cost in the initialization step. **Pose.:** average time cost on each frame for camera pose estimation. **Model update:** average time cost on each key frame for multi-view stereo and model update.

| Dataset | Seq. SfM | | Our approach | |
|---|---|---|---|---|
| | Trans. | Rot. | Trans. | Rot. |
| Rochester video | 0.071 | 0.56° | 0.020 | 0.08° |
| WPAFB 2009 | 0.065 | 0.36° | 0.009 | 0.09° |
| Providence (site22) | 0.273 | 0.19° | 0.027 | 0.15° |

Table 2. Quantitative comparisons against Global SfM. **Trans.**: average distance difference of camera positions. Due to the lack of physical scale, the distance is measured in the same coordinate units. **Rot.**: average angular difference of viewing directions.

### 3.5. Urban modeling

In the maintained 2.5D DSM, the geometric information of terrain and building roofs are well represented while the structures of building side-walls and regions with multiple heights are discarded. The last step of our approach is to build a full 3D model by integrating the depth maps of key frames from different views. We use the volumetric integration method with truncated signed distance function (TSDF) proposed in [5]: depth maps as well as the confidence maps are fused into a weighted discrete voxel grid with the value of each voxel representing its distance to the closest surface. The final mesh model is extracted from the voxel grid via marching cubes [14]. Due to the limitation of memory size, we process the 3D model block by block. The size of voxel grid for each block is adjusted automatically according to the height values in the DSM $M_D$.

## 4. Experiments

**Datasets.** We evaluate our approach on threel real-world datasets: **Rochester video** contains an aerial video of Rochester, NY, USA; **WPAFB 2009** contains aerial imagery captured flying over Wright-Patterson Air Force Base, OH, USA; **Providence (site22)** [21] is a smaller-scale aerial video of Providence, RI, USA. We also generate **Synthetic terrain**, a synthetic video of mountainous terrain in order to quantitatively evaluate the performance of our approach and its flexibility for terrain. The video is rendered following a camera motion path of a flyover around the target scene. More dataset details are listed in Table 1.

**Implementation.** Experiments are performed on a machine with Intel Xeon 3.6GHz quad-core processor and GTX 590 graphics card. We use FlowLib [1] and SiftGPU [2] to compute TV-$L_1$ optical flow and detect SIFT features, which are both parallelized on GPU. The multi-view stereo algorithm is also parallelized on GPU with CUDA implementation. In multi-view stereo, we process the high-resolution image tile by tile in order to fit the limited GPU memory. Other modules are parallelized on a multicore processor.

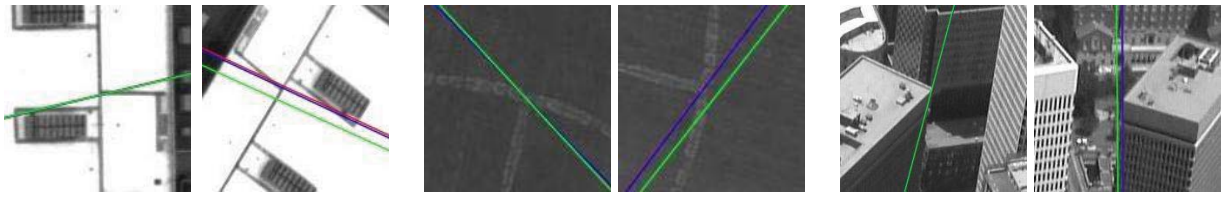
### 4.1. Camera pose estimation results

Due to the lack of ground truth, we generate camera poses on the real-world datasets using standard SfM algorithm with Bundle Adjustment for evaluation. Since our approach is parallelized on GPU and multicore processor, in order to evaluate the efficiency fairly, we use VisualSFM [3] which is also highly parallelized for feature detection and Bundle Adjustment [2, 25]. We generate the "gold-standard" camera poses using pairwise feature matching between all frames (Global SfM). We also generate camera poses using sequential feature matching between successive frames (Sequential SfM) for comparison.

For real-world datasets, we use several different methods to evaluate the accuracy of estimated camera poses. As shown in Figure 7, we use epipolar lines to check the consistency of camera poses. The epiporlar lines are consistent between frame pairs for the "gold-standard" camera poses from Global SfM as well as the poses estimated using our approach. Large errors exist for the camera poses estimated from Sequential SfM. As shown more clearly in Figure 8, the camera path estimated from our approach aligns well with the "gold-standard" camera path, while error accumulates in the camera path estimated from Sequential SfM. Quantitative comparisons also validate the performance of our approach as shown in Table 2. For **Synthetic terrain** where ground-truth camera poses are available, the camera position error of our approach is less than 0.1m which is small considering the video is rendered following a circular camera trajectory with diameter larger than 1km.

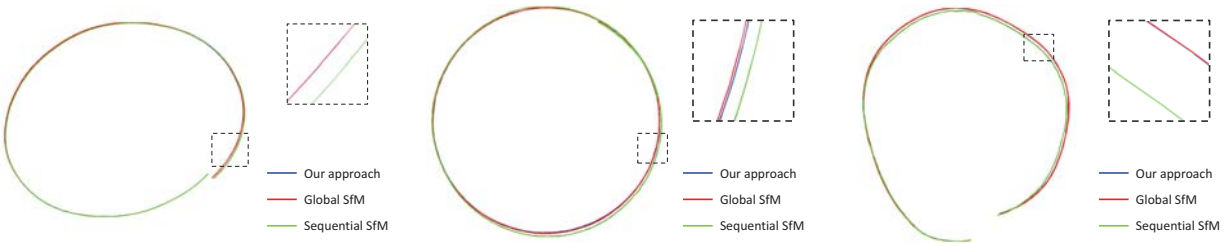
### 4.2. Reconstruction results

**Synthetic terrain data.** To quantitatively evaluate the reconstruction performance of our approach, we compare the reconstructed full 3D model to the ground-truth syn-



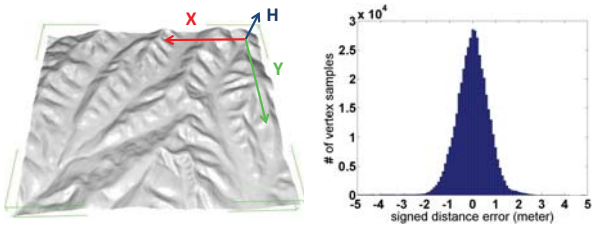
(a) Rochester video (b) WPAFB 2009 (c) Providence (site 22)

Figure 7. Display of epipolar lines: blue lines from out approach; red lines from Global SfM; green lines from Sequential SfM. Epipolar lines from our approach and Global SfM are consistent and overlapped. Large errors exist in the epipolar lines from Sequential SfM.




(a) Rochester video (b) WPAFB 2009 (c) Providence (site22)

Figure 8. Display of camera paths on different datasets. The camera paths from our approach and Global SfM are overlapped.




(a) Synthetic terrain model (b) Histogram of signed distance

Figure 9. (a) The model spans horizontally in a 2.5km $\times$ 2.3km region with a height range of 230m. (b) The signed distance error is unbiased with a mean of -0.01m and the RMS error of 0.72m.

thetic model of **Synthetic terrain**. The synthetic model (Figure 9 (a)) is built from a geo-tagged textured terrain DSM, which has 8.4-meter lateral spacing and 1-meter elevation precision. For evaluation, we measure the signed distance of the full 3D model reconstructed to the ground-truth synthetic model. As shown in Figure 9 (b), the reconstructed 3D model fits the ground-truth model very well with an unbiased signed distance error and a small RMS error considering the size of the synthetic model.

**Real-world dataset.** Due to the lack of ground-truth, we evaluate the reconstruction results on real-world datasets qualitatively. The reconstructed full 3D model on Rochester video are shown in Figure 10. More close-up views are shown in Figure 11. The geometric details, *e.g.*, chimney, bridge, are well captured in the reconstructed full 3D model. More results are provided in the supplementary material.

### 4.3. Running time analysis.

The running-times are shown in Table 1. Our approach estimates camera poses as accurately as Global SfM without drift along the path. At the same time, it works efficiently in an online fashion. Moreover, our approach provides not only accurate camera poses but also dense 3D geometric information during the process, *i.e.*, the DSM and the dense height maps of all key frames. By contrast, standard SfM outputs the camera poses and a sparse point cloud. In order to get the dense 3D information as accomplished by our approach, a separate dense reconstruction procedure is still needed to complete the pipeline.

## 5. Conclusion and Future work

We present an approach to solve camera pose estimation and dense reconstruction from WAAS videos. Our approach estimates camera poses as accurately as standard Global SfM algorithm and remain efficient as an online approach without global Bundle Adjustment. We also produce a highly-detailed full 3D model. There are large overlap between all frames in a WAAS video, which is the key for the success of our approach using the 2D-3D image-model feature matches. Our future work is to extend our approach to camera paths more complex than the hovering ones used here. We are also interested in evaluating the reconstruction performance quantitatively on real-world datasets where ground-truth models are available.

Figure 10. Reconstructed full 3D model on Rochester video (textured model and shaded model)

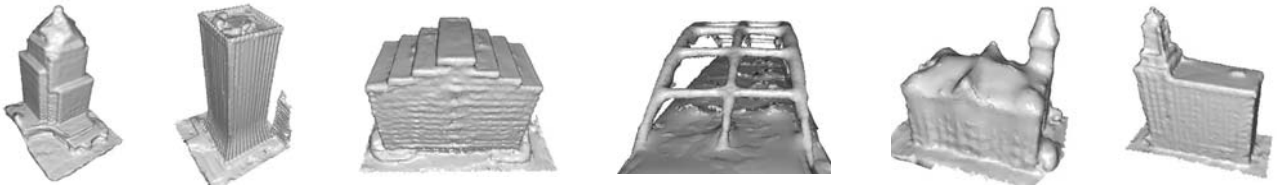


Figure 11. Closeup views of the results on Rochester video (shaded model)

## 6. Acknowledgments

This work was supported by grant DE-NA0001683 from the U.S. Department of Energy.

## References

[1] FlowLib. www.gpu4vision.org. 6

[2] SiftGPU. http://cs.unc.edu/~ccwu/siftgpu. 6

[3] VisualSFM. http://ccwu.me/vsfm/. 6

[4] F. Calakli, A. O. Ulusoy, M. I. Restrepo, J. L. Mundy, and G. Taubin. High Resolution Surface Reconstruction from Multi-view Aerial Imagery. In *3DIMPVT*, 2012. 2

[5] B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 303–312. ACM, 1996. 6

[6] Y. Furukawa, B. Curless, S. M. Seitz, and R. Szeliski. Manhattan-world stereo. In *CVPR*, 2009. 2

[7] Y. Furukawa and J. Ponce. Accurate camera calibration from multi-view stereo and bundle adjustment. *IJCV*, 84(3):257–268, 2009. 2

[8] Y. Furukawa and J. Ponce. Accurate, dense, and robust multiview stereopsis. *PAMI*, 32(8):1362–1376, 2010. 2

[9] V. H. Hiep, R. Keriven, P. Labatut, and J.-P. Pons. Towards high-resolution large-scale multi-view stereo. In *CVPR*, 2009. 2

[10] G. Klein and D. Murray. Parallel tracking and mapping for small ar workspaces. In *ISMAR*, 2007. 2

[11] R. Kumar, P. Anandan, and K. Hanna. Direct recovery of shape from multiple views: A parallax based approach. In *ICPR*, 1994. 5

[12] H. Li. Multi-view structure computation without explicitly estimating motion. In *ICCV*, 2010. 2

[13] H. Liao, Y. Lin, and G. Medioni. Aerial 3D reconstruction with line-constrained dynamic programming. In *ICCV*, 2011. 2

[14] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *ACM Siggraph Computer Graphics*, volume 21, pages 163–169. ACM, 1987. 6

[15] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004. 1

[16] R. Newcombe, S. Lovegrove, and A. Davison. DTAM: Dense tracking and mapping in real-time. In *ICCV*, 2011. 2, 3

[17] D. Nistér. An efficient solution to the five-point relative pose problem. *PAMI*, 26(6):756–770, 2004. 2

[18] T. Pollard and J. L. Mundy. Change detection in a 3-d world. In *CVPR*, 2007. 2

[19] M. Pollefeys, D. Nistér, J. Frahm, A. Akbarzadeh, P. Mordohai, B. Clipp, C. Engels, D. Gallup, S. Kim, P. Merrell, et al. Detailed real-time urban 3d reconstruction from video. *IJCV*, 78(2):143–167, 2008. 2

[20] J. Prokaj and G. Medioni. Using 3d scene structure to improve tracking. In *CVPR*, 2011. 1

[21] M. I. Restrepo, B. A. Mayer, A. O. Ulusoy, and J. L. Mundy. Characterization of 3-d volumetric probabilistic scenes for object recognition. *IEEE Journal of Selected Topics in Signal Processing*, 6(5):522–537, 2012. 6

[22] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon. Bundle adjustment - a modern synthesis. *Vision algorithms: theory and practice*, pages 153–177, 2000. 1

[23] A. Wendel, M. Maurer, G. Graber, T. Pock, and H. Bischof. Dense reconstruction on-the-fly. In *CVPR*, 2012. 2

[24] C. Wu. Towards linear-time incremental structure from motion. In *3DV*, 2013. 4

[25] C. Wu, S. Agarwal, B. Curless, and S. M. Seitz. Multicore bundle adjustment. In *CVPR*, 2011. 6

[26] C. Zach, T. Pock, and H. Bischof. A duality based approach for realtime TV-L1 optical flow. In *DAGM*, 2007. 5

[27] X. Zhao and G. Medioni. Robust unsupervised motion pattern inference from video and applications. In *ICCV*, 2011. 1