

# Alignment of Continuous Video onto 3D Point Clouds

W. Zhao<sup>1</sup> , D. Nister<sup>2</sup> , and S. Hsu

Sarnoff Corporation

201 Washington Road

Princeton, NJ 08540, USA

email: { wzhao, dnister, shsu }@sarnoff.com

Tel: 609-734-2725

Fax: 609-734-2662

<sup>1</sup> Corresponding Author

<sup>2</sup> The author is currently with Universtiy of Kentucky

## Abstract

*We propose a general framework for aligning continuous (oblique) video onto 3D sensor data. We align a point cloud computed from the video onto the point cloud directly obtained from a 3D sensor. This is in contrast to existing techniques where the 2D images are aligned to a 3D model derived from the 3D sensor data. Using point clouds enables the alignment for scenes full of objects that are difficult to model, for example, trees. To compute 3D point clouds from video, motion stereo is used along with a state-of-the-art algorithm for camera pose estimation. Our experiments with real data demonstrate the advantages of the proposed registration algorithm for texturing models in large-scale semi-urban environments.*

*The capability to align video before a 3D model is built from the 3D sensor data offers new practical opportunities for 3D modeling. We introduce a novel modeling-through-registration approach that fuses 3D information from both the 3D sensor and the video. Initial experiments with real data illustrate the potential of the proposed approach.*

**Index Terms** — Alignment, pose estimation, motion stereo, range data, sensor fusion, 3D model & visualization

## I. INTRODUCTION

This paper presents a framework to automatically align continuous video onto 3D point clouds that represent the geometry of the 3D world. In particular, we demonstrate its application to aligning *oblique* video onto *nadir-view* point clouds directly obtained from a 3D sensor such as LiDAR (Light Detection and Ranging). Figure 1 shows such an example where the 3D world has been observed with different sensors (video vs. LiDAR) at different viewing angles (oblique vs. nadir) and different times. The task of automatically registering 2D images onto a 3D model of a scene has been recognized as a key step in many applications: 3D modeling of urban environments [9], [33], [12] where 3D geometry and photometric information of the real world are recovered and registered; Virtual Environment (VE) applications [6] where realistic synthetic views of existing scenes from a sparse set of still photographs can be created. The common goal of these applications is to quickly build photo-realistic 3D models with correct geometry.

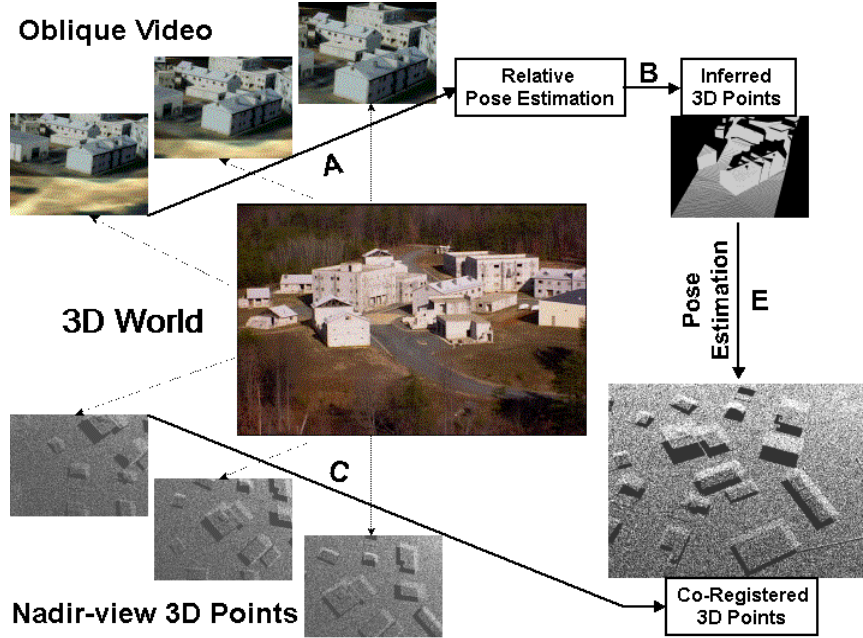


Fig. 1. A given scene and its different observations: oblique-view video and nadir-view 3D sensor points. The proposed alignment framework for building the 3D model consists of solving the following tasks in sequential order: 1) Task A is to recover relative camera poses; 2) Task B is to compute 3D point clouds from the video; 3) Task E is to determine the final camera poses by aligning point clouds. Task C for co-registering the 3D sensor points is outside the scope of this paper (To differentiate, we call the registration between data sets of the same original format co-registration.). Task D for building a 3D geometric model is *not* required in the proposed alignment framework; hence, not illustrated.

Existing approaches to building such a model mainly have three separate steps: 1) Build a 3D model using data from the 3D sensor; 2) Align 2D images onto the 3D model; 3) Texture the 3D model using the aligned images. Though there are systems [32] where alignment is not required since two sensors are rigidly attached to the same platform, we are interested in the general case where there is an increasing need for *automatic* alignment [9], [12].

For other applications such as AVE (Augmented Virtual Environment) applications [27], [30], a rapid and accurate dynamic appearance update of the model, e.g., *dynamic texture projection*, is crucial to present observers with a single coherent and evolving view. Again, the alignment of 2D images onto a 3D model is critical.

The alignment framework presented in this paper is based on aligning 3D point clouds from

different sensors, the most general form of common 3D information<sup>1</sup>. More specifically, we perform the following steps:

- Recover the relative camera poses in real-time from a video sequence (Task A in Fig. 1).
- Compute 3D point clouds from the video using motion stereo (Task B in Fig. 1).
- Determine the camera poses by aligning 3D point clouds from the video and the 3D sensor using ICP (Iterative Closest Point) (Task E in Fig. 1).

To see the impact of accurate alignment, we overlay the motion stereo point cloud onto a 3D model based on initial camera pose and camera pose refined via ICP in Fig. 2. Unlike the proposed framework, most existing alignment frameworks assume that a 3D geometric model has been built from the 3D sensor data [13] and features, e.g., 2D and 3D lines, are extracted to compute/refine the camera poses with respect to the 3D model [33], [27].

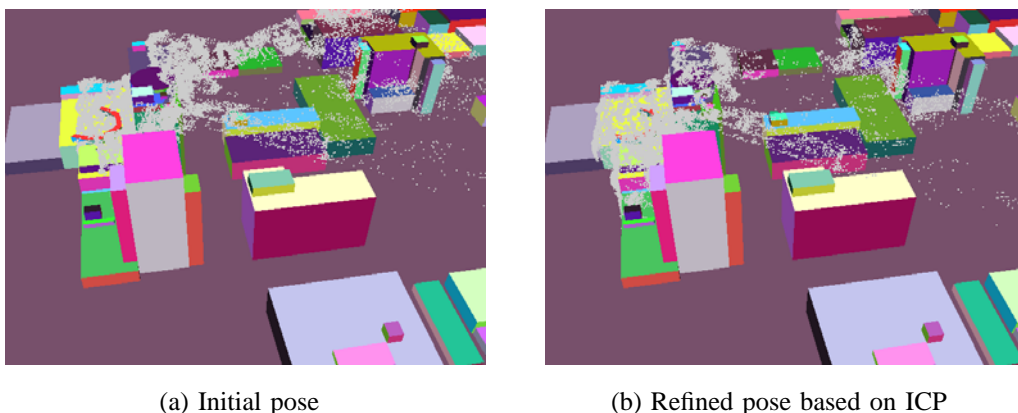


Fig. 2. Overlaying the motion stereo point cloud onto a 3D model based on (a) initial camera pose, and (b) ICP-refined pose. The translational corrections of the camera position are -5, -1.3 and -1.5 meters in three respective directions. To better appreciate the importance of accurate camera pose, please see the different results of texturing the 3D model in Fig. 5. To make the before- and after-refinement difference more visible, the plots here have a side-viewing angle that is different from that in Fig. 5.

The capability to align video before a 3D model is built from the 3D sensor data offers many unique features. It is possible to estimate camera poses for scenes full of objects that are difficult to model, for example, trees. As a result, continuous pose estimation of a moving camera is possible even in dynamic complex scenes where new buildings could emerge and/or

<sup>1</sup>Throughout this paper, a 3D model is **not** used for alignment in the proposed alignment framework, but it will be used for the purposes of texturing and comparison.

old buildings could be demolished. In addition, a point cloud from motion stereo can be directly combined with 3D sensor data to create a new model since they are already aligned.

The remainder of this paper is organized as follows: We first review related work in Section II. In Section III, we present the framework for aligning video onto a 3D point cloud, including first the alignment of a pair of frames and then that of continuous video. After presenting experimental results on real data in Section IV, we present a novel modeling-through-registration approach and promising initial experimental results in Section V. Finally, we conclude our paper in Section VI with extended discussions on further improvements.

## II. RELATED WORK

The goal of video-to-3D registration is to determine the pose of each video frame from a moving camera with respect to the given 3D scene. Pose can be established by several principal means, for example, physical measurement and 2D frame to 3D scene matching constraints. Not directly applying these physical measurements (i.e., imaging function), one significantly different and interesting alignment framework is to explore the predictability of image from the model (i.e., entropy) [38]. This framework has been widely adopted in the medical imaging community but is beyond the scope of this paper. We refer interested readers to [38] for more details.

Physical measurements include GPS and inertial sensors (INS) co-located with the moving video camera. High pose accuracy and measurement frequency can be achieved with complex and costly airborne instrumentation in conjunction with ground surveys and differential GPS base stations, as usually practiced in aerial LiDAR acquisition (Task C in Fig. 1 is typically completed this way). However, in many airborne video scenarios, GPS/INS pose is not available continuously for every frame and is distorted by significant outliers, biases, and drift. Thus, GPS/INS pose measurements alone are not sufficient in practice for aligning video to 3D scenes accurately. Nevertheless, they provide important approximate information to bootstrap the video-based alignment.

In the class of 2D frame to 3D scene matching approaches, image appearance features like high contrast points and lines are matched to geometric scene features like corners and edges. If enough correct correspondences are identified manually [9], [12] or automatically [27], [33], the pose parameters can be estimated for the frame [21].

Most existing work for pose estimation belongs to this category and we describe a few representative ones: registration of lines [18] and curves [21], registration of rectangular structures [34] and registration of cliff maps [28].

In [28], a method was proposed to align far-range video to a DEM (digital elevation map) via first recovering 3D geometry from the video by applying stereoscopic analysis on a pair of adjacent images and then registering it onto the DEM to obtain the camera position and heading. Based on the assumption that both the video camera and photographic camera point downwards and move in parallel with the ground, it is reasonable to represent the 3D information in a 2D image. The authors hence propose a compact 2D representation, i.e., the so-called cliff maps that consist of edges and high-curvature points [28]. The drawback of this approach is that the reduction of full 3D information to 2D cliff images will limit the algorithm's capability of handling oblique 3D pose.

In [18], 2D and 3D lines are registered to align video to the 3D model. In addition, bundle adjustment [37] can be implemented to obtain more consistent and accurate pose estimation for a sequence of images. Specifically, the method [18] first projects selected 3D line segments onto the image via the current camera pose. It then refines the pose by maximizing the integral of the gradient energy map (computed from the image) along the projected 3D line segments. In [21], 3D curves/lines are projected onto 2D image too. However, the cost function now changes to the sum of distance measures from the image edge points to the closest projected curves/lines.

In [34], more constraints are explored in addition to the lines by restricting the application scenarios: parallelism and orthogonality constraints that naturally exist in urban buildings. More specifically, 3D scene directions are matched with 2D image vanishing points formed by parallel lines to solve for camera rotation. Next, clusters of parallel 2D & 3D lines are matched for initial estimation of translation. Finally, cluster of 3D lines are projected onto 2D image to estimate the final pose.

One common limitation of these 2D-3D approaches is that they can not handle scenes lacking easily extracted lines and curves. In addition, it is not easy for these approaches to correct for large 3D pose error due to possible significant occlusions. Finally, it is time-consuming and sometimes difficult to obtain accurate 3D lines from noisy range images since planar segmentation of range data is a prior step [33].

The framework we propose belongs to a class of algorithms that have not been carefully

investigated. In this class of approaches, multiple frames around the frame of interest are first used together to reconstruct 3D structure in a local coordinate system, and then the 3D reconstruction is matched to the known 3D scene in order to estimate the transformation between local and scene coordinate systems. In this paper, we use the most general format of 3D representation, 3D point clouds, for 3D-3D registration.

Similar to our 3D-3D registration framework, there exist object tracking works that perform full 3D pose estimation based on range sensor rather than monocular video camera [31], [16]. In [31], a high speed VLSI range sensor based on projecting light stripes is used to acquire 3D information ( $32 \times 32$  resolution) of an object (e.g., face) in real-time (10 Hz). To track the object, acquired 3D data is matched against a triangular mesh model of the object at a pose based on previous estimation. In [16], a stereo-based range sensor is used to obtain frame-rate depth information of a tracked face object. Rather than directly registering 3D data sets, a new depth constraint equation is integrated with the classical brightness change constraint for robust 3D pose tracking. In these applications [31], [16], the focus is on tracking close-range objects rather than texturing large scale semi-urban environments. One reason is that small fixed baseline limits the application scenarios of a stereo-based range sensor [23].

### III. ALIGNMENT OF CONTINUOUS VIDEO ONTO 3D POINT CLOUDS

#### A. Framework for frame alignment

Our framework for aligning a pair of video frames can be summarized as follows (Fig. 3): 1) Recovery of camera pose; 2) Motion stereo to recover dense 3D point sets; 3) ICP registration of 3D data for pose estimation. The motivations for choosing these algorithms can be summarized as follows:

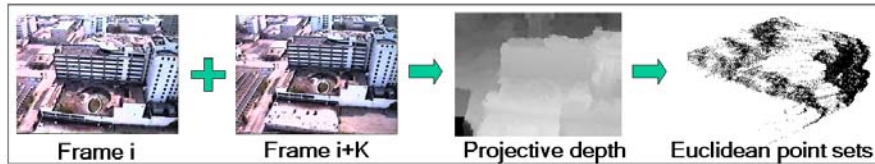
- 3D point cloud for maximal flexibility
- ICP has proven to be effective in aligning 3D data
- Dense motion stereo provides large amounts of point clouds good for ICP alignment
- Camera pose estimation is required to compute motion stereo based on a moving monocular camera

These considerations are central to our motivations. In the following subsections we give them in more detail. We will also discuss the choice of the particular algorithms used at each step.

## 1. Camera Pose Estimation



## 2. Motion Stereo



## 3. Matching/Pose Estimation

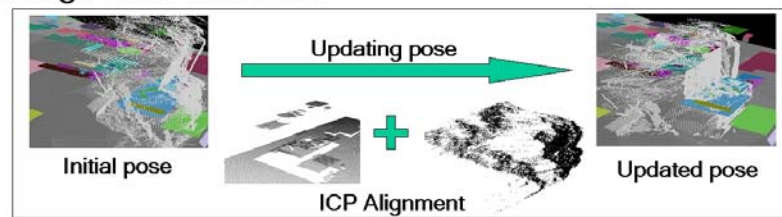


Fig. 3. Flow chart of the proposed three-step framework for frame alignment. Notice that in each step, there are a few small steps involved. Motivations for choosing these algorithms can be summarized as follows: 1) 3D point cloud for maximal flexibility; 2) ICP has proven to be effective in aligning 3D data; 3) Dense motion stereo provides large amounts of point clouds good for ICP alignment; 4) Camera pose estimation is required to compute motion stereo based on a moving monocular camera.

1) *Why 3D point representation:* To register video frames onto 3D sensor data, we must use the only common feature, 3D geometry. However, from the above brief description of existing algorithms for monocular camera based video-to-3D registration, we notice that none of them perform alignment based on full 3D information. For example, extracted lines, rectangles and cliff maps were used to represent 3D data. In this paper, we propose using point clouds for 3D-3D registration. There are several advantages with such 3D-3D registration. The first one is its capability to handle large pose variation. This is significant since in practice the initial pose estimate often deviates from the true value and 2D representations could appear dramatically different due to pose variation. The second one is that it removes the demand for building a 3D model (or at least segmenting planar surfaces for extracting lines) from 3D sensor data prior to registration. This suggests that we can align video onto 3D sensor data in areas where it was not

possible before, for example, areas with trees. Finally, it offers the opportunity of dynamically building complete geometry from both video and 3D sensors. Like existing work, we assume the existence of an initial pose estimate from physical sensors or manual calibration (e.g., Fig 2(a)).

2) *Why ICP for alignment:* There exist many methods for 3D registration. Please refer to a recent survey paper [3] for details. Over the past 10 years, the ICP (Iterative Closest Point) algorithm has become the dominant method among various 3D alignment techniques. There are several reasons for the wide applications of the ICP algorithm that was pioneered by Besl and McKay [2]. In [2], ICP is proven to converge to a local minimum. In practice, it was shown that the global minimum can be achieved with a good initial alignment [22], [5], [29]. However, starting from an arbitrary point, only a local minimum will be achieved in general. Second, ICP can work with wide range of data representations: point sets, polylines, curves, surfaces. Finally, the implementation of ICP is straightforward and there exist many variants tailored towards specific tasks at hand: employing adaptive distance threshold during ICP iterations [39], point-to-plane based cost function [4], random sampling of point sets at each iteration [22], the use of closest compatible points that satisfy the additional constraints of similarity in attributes [10]. For review and comparison of efficient ICP variants, please see a recent article [29].

3) *Why dense motion stereo:* Assuming that we choose ICP as the main engine for registering 3D data in various formats including raw point clouds and polygonal 3D models, the remaining issue is how to recover 3D information from video sequences. For the success of ICP alignment, it is best to have full description of 3D structures in a scene that could be very complicated. For a counter example where sparse 3D representations caused ICP alignment failure, please refer to Fig. 4(b) for the sparse points reconstructed from Structure and Motion based on feature points. Dense 3D structure from given pairs of frames can be recovered by dense motion stereo. Depending on the scene structure and the chosen motion stereo algorithm, erroneous 3D reconstructions could occur in certain regions. However, ICP can handle outliers as long as we have a sufficient number of inlier points. A prior step to motion stereo is the recovery of relative camera pose.

The remaining of this Section is organized according to the logical flow of the proposed frame alignment framework (Fig. 3) and its extension to video alignment.

## B. Recovery of camera pose

Before 3D structure can be reconstructed from video, the camera pose must be estimated. The camera pose can be expressed using a 3 by 4 camera matrix  $p$ , which can be further decomposed into extrinsic and intrinsic camera parameters as

$$p = KW = K[R \mid T] \quad (1)$$

where  $K$  is the intrinsic 3 by 3 *calibration matrix*,  $W$  is the extrinsic orientation matrix where  $R$  is the extrinsic rotation matrix (3 by 3) and  $T$  is the translation vector (3 by 1). All these parameters need to be determined for each frame.

The most favorable case of pose estimation is when we know the  $K$  matrix, for example, through a prior calibration procedure. In cases where we do not know  $K$ , we can estimate it, but this is less stable than pose estimation with efficient use of a known  $K$ .

We use the real-time method described in [25] for pose estimation where the intrinsic parameters of the camera, such as focal length, are assumed known a priori. Specifically, an efficient algorithmic solution to the classical five-point relative pose problem [7] is presented. Feature point matches (Harris corners [14]) are extracted for consecutive image pairs by appearance-based matching. A robust hypothesize-and-test scheme based on a fast version [25] of RANSAC [8] is used to estimate the best relative pose. To reduce the ambiguity of motion estimation, more than two views are used.

Given the camera poses  $p$  and  $p'$  of two frames, we can perform triangulation [15] to determine the 3D position  $X$  corresponding to two matching points  $\mathbf{x}$  and  $\mathbf{x}'$ . However, only the relative positions of the 3D points and cameras can be recovered from images. The local and world Euclidean coordinate systems are related by a 4 by 4 similarity matrix

$$H = \begin{bmatrix} sR & T \\ 0 \ 0 \ 0 & 1 \end{bmatrix} \quad (2)$$

where  $s$ ,  $R$  and  $T$  represent scale, rotation and translation respectively. If we have GPS/inertial sensors, we can obtain the initial absolute camera positions in the world coordinate system. The remaining steps are to refine the absolute camera pose via ICP alignment of 3D points.

## Discussions

Based on the extensive experimental results [25], it was shown that the five-point algorithm

is good at recovering pose and that using knowledge of the intrinsic parameters is important. Though in practice pose estimation is challenging for cases of weak geometry in the scene, the calibrated approach can deal with planar scenes correctly in contrast with the uncalibrated approach.

However, like any pose estimation algorithms, this algorithm suffers from correspondence failure due to significant motion blur and/or low texture. In addition, the issue of inaccurate intrinsic parameters (e.g., camera focal length) is significant. If the initial calibration is inaccurate, one could apply iterative refinement from the estimate obtained with approximate intrinsic parameters.

Finally, the well-known fact that estimation of rotation is in general more accurate than estimation of translation [36], [25] is explored in our application to increase the efficiency of ICP alignment.

### *C. Recovery of 3D structure*

Given camera poses, motion stereo methods try to recover dense 3D information for the whole field of view. They typically rely on correlation of corresponding small image regions. To prevent erroneous 3D reconstruction, areas with low confidence can be discarded, resulting in less than 100% dense 3D reconstruction. Though earlier stereo methods depend on parallel/non-convergent camera configuration, imagery from non-parallel camera configurations can be rectified by using the (estimated) camera poses [15]. In this paper, we use the popular plane-plus-parallax framework [20], [19] to compute depth. Combined with the so-called direct methods that work directly on images, it can be implemented efficiently in a multi-resolution paradigm. Though there exist more elaborate methods for 3D reconstruction, we choose to use this algorithm in order to have an efficient framework.

To convert the depth map of motion stereo into a point cloud, we simply convert each pixel in the depth map into a 3D point using the camera matrix (Eq. 1). Note that this point cloud is in a local Euclidean coordinate system and needs to be converted into a global point cloud via multiplying the similarity matrix (Eq. 2) for matching against a global 3D sensor data.

For comparison, typical 3D reconstructions from two-frame based Motion Stereo (MS) and thirty-frame based Structure and Motion (SM) are shown in Fig. 4. From the plot, two comparisons can be made here: 1) The number of selected feature points in SM is barely enough to

describe the complex scene (about 350 points for 30 frames in Fig. 4(b)) while MS gives a dense reconstruction; 2) Most of the strong features with repetitive patterns (windows on the wall) are not selected for SM (Fig. 4(d)) but are excellent for stereo matching once the poses have been established and when an efficient multi-resolution representation is used in direct methods.

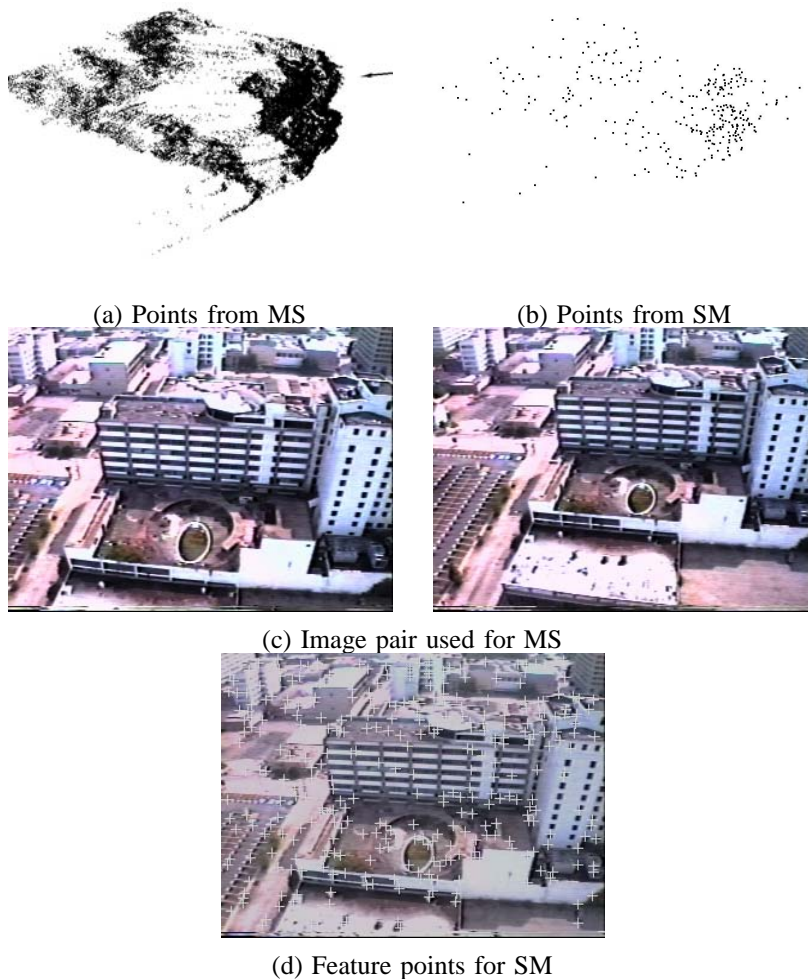


Fig. 4. 3D reconstruction from two frames: (a) Dense reconstruction based on **two** frames using MS [20] (small arrow indicating the viewing direction). To convert the depth map of MS into a point cloud, we simply convert each pixel in the depth map into a 3D point. (b) SM reconstruction based on feature points (roughly 350 points) across 30 frames [25]. (c) Image pair used to compute stereo; (d) Selected feature points on one of the frames used for SM. Based on our experiments, the application of ICP for alignment based on SM-based point clouds (b) is not successful.

## Discussions

The main reason that we choose multi-resolution plane-plus-parallax framework is its efficiency

and the ability to handle relatively large motion. Though this stereo algorithm does not produce the best quality depth, it turns out to be quite effective for our application of alignment. However, for other applications such as building 3D geometry (Section V) where 3D reconstruction should contain smaller noise and preserve sharp edges, more elaborate methods are required.

Like many stereo algorithms, this algorithm could fail for certain images/scenes, for example, those containing largely textureless regions except a few reliable feature points. Under such cases, feature-based methods are more appropriate for computing the sparse structure. However, under these circumstances, we do not recommend using the proposed 3D-3D framework for ICP alignment. The apparent reason is that dense 3D information can not be recovered reliably.

The impact of inaccurate intrinsic parameters upon the 3D reconstruction is significant. For example, inaccurate focal length leads to distorted reconstruction, including scale change and non-uniform 3D distortion. As for the final goal of alignment, our experience is that the proposed framework can tolerate inaccurate values to some extent (refer to Fig. 8 for an example).

### **Baselines for pose recovery and dense motion stereo**

Compared to fixed-baseline stereo, motion stereo can choose desired baseline adaptively based on application scenarios. For example, in our application the camera is about 90 meters above the ground, making it impossible to implement a practical fixed-baseline stereo system that can achieve sufficient accuracy. On the other hand, we can easily adapt the baseline for motion stereo by simply selecting two frames that are sufficiently separated.

The issue of selecting a good baseline exists in both pose recovery and dense motion stereo. With frames too far apart, very few correspondences exist between the frames. With frames too close, the pose estimate will be geometrically weak and inaccurate. Hence it is in general important to adaptively select baseline [26] in estimating camera pose. However, in our particular application, we can use domain knowledge to determine good frames to work with. We approximately know the flight altitude (90 meters above the ground) and velocity (15 meters per second), and we also know the video frame rate (15 fps).

Different considerations for pose recovery and dense motion stereo lead to different choices of baselines. For pose recovery, an additional constraint is imposed to make sure that the pose algorithm [25] can track/view the same points across multiple selected frames. As a result, we choose computing pose every three frames (a baseline of 5 meters) to induce sufficient parallax

and sufficient overlap among multiple views. For dense motion stereo, we only need to consider two selected frames and the accuracy in 3D reconstruction is important. As a result, we choose computing motion stereo every nine frames (a baseline of 15 meters).

#### D. 3D-3D Registration and Pose Estimation

Within the general approach of 3D-3D registration, there are several variants where 3D information from video and 3D sensor can be in the raw format of point cloud or the 3D model format separately. We argue that among these variants the best choice is point-based 3D-3D registration.

First, registration of raw 3D point clouds offers the most flexibility and applicability among all variants. For example, all 3D data including the 3D model can be represented as a raw point cloud. Second, recovering a clean 3D model from video frames is time-consuming and sometimes difficult. Finally, it offers a great opportunity for fusing 3D information from both the 3D sensor and the video to build a final 3D model. We will come back to this important issue in Section V. On the other hand, if a 3D model already exists before video is available, point-to-plane metric might be a better choice in terms of ICP convergence speed [29].

1) *Implementing the ICP algorithm:* ICP is a popular algorithm for matching point clouds with unknown matching point pairs [2], [4], [39]. There are many variations but the basic version of ICP consists of iterating the following two steps: 1) Selecting point pairs by finding the closest points; and 2) Computing the best transformation between all chosen point pairs. The closest point pairs can be found via an efficient KD-tree algorithm [24]. In the motion-updating step, the 3D transformation  $H$  between the two matched point sets can be efficiently updated using a closed-form formula [17]. One key parameter of the ICP algorithm is the distance threshold  $d_T$  for finding closest points. To have a stable ICP [39], we should employ a varying  $d_T^i$  that decreases at each iteration  $i$ .

In our experiments with real data, the initial positions of MS point clouds are tens of meters away from the 3D sensor point cloud. To increase the capturing range of ICP, we employ the following two-step strategy for motion-updating

- First update the translation  $T$  using large distance threshold  $d_T^0$ , say, 35 meters.
- Then update the full transformation  $H$  using a smaller  $d_T^0$ , say, 5 meters.

The rationale for this two-step approach is the fact that rotation estimate is much more accurate than translation estimate during camera pose recovery (Section III-B).

2) *Pose estimation*: After ICP alignment, we finally update the absolute camera pose. Recall that the absolute camera pose can be obtained from recovered camera pose  $p$  (Section III-B) via the similarity matrix  $H$ . After ICP alignment, an updated similarity matrix  $H$  can be obtained, yielding the updated absolute camera pose,

$$P = p(H)^{-1} = K[R \mid T](H)^{-1} \quad (3)$$

## Discussions

We describe the details of key parameters used in our ICP implementation and a method of handling border effect. The key parameters used in our implementation include: 1) tolerance threshold  $t_T$  used in building approximate nearest neighbor (ANN) KD-trees from point clouds, 2) distance threshold  $d_T$  used in finding closest points, 3) number of total iterations  $N$ , and 4) number of randomly sampled points  $S$  at each iteration.

The tolerance threshold  $t_T$  in building a ANN KD-tree affects the accuracy of accessing closest points [24]. Only zero tolerance threshold guarantees that exact closest points can be found. However, our experience indicates that ANN KD-trees based on several non-zero values work well for matching. The distance threshold  $d_T$  is clearly related to the capturing range of the ICP algorithm. Assuming the convergence of ICP, larger  $d_T$  implies larger capturing range. For a decaying  $d_T$ , the alignment accuracy of ICP is related to the final value of  $d_T$  but limited to the noise present in point clouds. To ensure a stable convergence of ICP, the decaying speed of  $d_T$  should be limited. Based on our experiments, 50 iterations typically provide good convergence with the final value of  $d_T$  being one-tenth to one-fifth of its initial value. For the two-step ICP alignment, the initial value of  $d_T$  in step-II ICP should be larger than the final value  $d_T$  of step-I ICP. Finally, random sampling of points at each iteration is used to improve the algorithm efficiency. Our experiments indicate that it is sufficient to sample 2000 points as recommended in [29]. Since video-inferred point clouds are noisy compared to 3D sensor point clouds (at least in our application), we warp video point clouds to the coordinate of 3D sensor point clouds for matching.

Though there exist many variants of ICP algorithm, there are limitations of this algorithm. For example, it is in general impossible to align two planar scenes. In practice, serious errors

in 3D reconstruction can also cause ICP to diverge. For example, when camera is viewing small 3D structures or scenes with weak geometry, reconstruction errors due to pose estimation (Section III-B) and structure recovery (Section III-C) could overwhelm the true 3D structures in terms of the influence upon alignment (refer to Fig. 9 for an example). One solution to this problem is to apply bundle adjustment (refer to Section III-E for details) assuming other frames in the video can recover good 3D structures.

### **Handling border effect**

One important issue in matching two sets of point clouds is the *border effect* caused by border-points that lie on the borders of each point cloud. Meanwhile, there are plenty of object boundaries in a scene full of objects and points from these object boundaries are boundary points. In practice, border-points could be boundary points or non-boundary points since frames have different viewing angles and/or different field of views. For example, overlapping image borders of video cameras are often converted to non-boundary border-points through motion stereo. The existence of these non-boundary border-points leads towards biased motion estimation. To eliminate such bias without automatically determining whether the border-points are boundary points or not, we do not include any border-points from two corresponding point clouds in the process of selecting closest point pairs. This way, all the border-points, especially those non-boundary points, will not contribute towards motion computation.

The border effect happens as follows: First, border-points  $X_i$  from one point cloud A are warped to the outside of the border regions of the other point cloud B. Then, closest points of  $X_i$  from B can only be found in one direction, i.e., from the inside of the border regions of B. This is different from normal cases where closest points could come from all directions in a region of B surrounding warped  $X_i$ . As a result, the motion estimated between A and B is leaning towards to pulling borders from A and B closer at each iteration.

### *E. Continuous video alignment*

Up to now, we have been focusing on aligning pairs of frames. But how can we extend this approach to compute the camera poses of a video sequence? The simplest approach is to choose a pair of key frames every  $K$  frames and apply frame alignment.

To recover the poses for non-key frames, bundle adjustment [37], a standard non-linear minimization procedure used in photogrammetry, is used. We apply bundle adjustment (Levenberg-

Marquardt algorithm) to enforce global consistency of the estimated camera poses based on relative camera poses (Section III-B) and frame alignment (Section III-A). As a result of bundle adjustment, we can not only interpolate the camera poses for non-key frames but also smooth the poses for key frames. In Figure 9, a real example is shown where ICP registration fails to converge to the global minimum and bundle adjustment helps to make a correction. More specifically, let  $\hat{H}_i$  represent the unknown similarity transformation from camera-centered local coordinate system in frame  $i$  to world coordinates, and let  $H_i$  be the similarity obtained by ICP frame alignment (Eq. 3). Let  $W_{i,i+K}$  represent the rigid transformation between adjacent cameras  $i, i + K$  as estimated by pose recovery (Eq. 1). Let  $\{u_{im}\}$  be a set of 3D test points in frame  $i$  that lie in the vicinity of the scene structure (any 3 or more well-distributed points suffice). We jointly estimate the unknown  $\{\hat{H}_i\}$ 's by minimizing the cost function

$$E = \sum_i E_i + \lambda \sum_i E_{i,i+K} \quad (4)$$

where

$$E_i = \sum_m \left| \hat{H}_i(u_{im}) - H_i(u_{im}) \right|^2 \quad (5)$$

penalizes inconsistency with frame alignment, and

$$E_{i,i+K} = \sum_m \left| \hat{H}_i(u_{im}) - \hat{H}_{i+K}(W_{i,i+K}(u_{im})) \right|^2 \quad (6)$$

penalizes inconsistency with pose recovery, and  $\lambda$  is the ratio of frame alignment variance to pose recovery variance.

In summary, we process video frames continuously using the following steps:

- Perform frame alignment (Eq. 3) to obtain the pose  $H_i$  for currently chosen key frame, initialized with the pose cascaded from previous key frame with frame-to-frame motions.
- Apply bundle adjustment (Eq. 4) to obtain the poses for both non-key frames and key frames.

One advantage of the proposed framework is that only close frames are used to compute the depth maps; hence, the algorithm is not very sensitive to accumulated errors of pose estimation in a long sequence. However, to handle cases when camera pose estimation has drifted off is beyond the scope of this paper and left as a future research topic.

## IV. EXPERIMENTS

In our experiments, 3D sensor data of Baton Rouge was acquired by a LiDAR scanner commercially employed for aerial topographic mapping. A single laser beam is scanned in a whiskbroom pattern, i.e. scanned along track by platform motion and across track by a one-axis scanning mirror. The average ground sampling distance is 50cm. By application of high accuracy GPS/INS, the LiDAR points are georeferenced directly to world coordinates without need for data-driven registration. For the purpose of visualization, we derive 3D models interactively from these point clouds with user inputs (e.g., Fig. 2). The motivation for doing so is that automatic algorithms such as point-surflet-surface approaches often result in many small and irregularly oriented facets to represent building surfaces. Since the laser beam stays near nadir, vertical surfaces are usually missing. Currently, the missing walls are defined by extruding the roofs down to ground level.

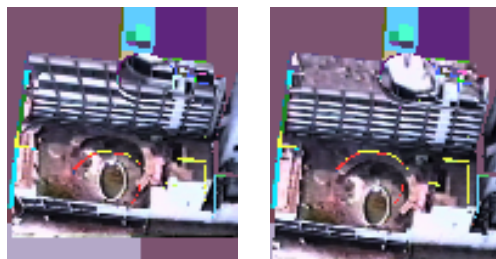
The video data was acquired separately and digitized at 15 frames per second with a frame size of 320 by 240. Based on the registration result, the camera platform is found to be about 90 meters above the ground. In our experiments, we perform manual calibration of two views to obtain the initial approximate  $H$  matrix.

### A. Recovery of camera poses

In our experiments, we use the real-time system developed in [25] to recover pose every three frames. Its performance has been extensively documented in [25].

Recall that motion stereo gives 3D reconstruction up to similarity unless we have the absolute camera positions in the world. Currently, the video captured does not come with GPS/inertial information, hence we must manually calibrate the video with two views in order to obtain the global/absolute orientation and global scale. We first click corresponding points and lines on video frames and the derived 3D model. We then perform a calibration algorithm to compute the parameters  $s$ ,  $R$ ,  $T$ . We have developed a method that gives stable results even when the input data is approximate. A search over 100 focal length values is performed, while the other intrinsic parameters are held fixed at approximate values. For each focal length value, the camera is considered calibrated and multiple hypotheses for the pose are generated from random triplets of points and random triplets of lines. The solutions from all triplets and all focal lengths are

scored based on their least squares reprojection error over the whole set of observations. The best solution is then iteratively refined.



Initial pose

ICP-refined pose

(a) Alignment comparison in the building region



(b) Based on ICP-refined pose (Fig. 2(b))

Fig. 5. Projecting the aligned image to texture a 3D model. Camera poses w.r.t. the 3D model are from Fig. 2. The small arrow in (b) indicates one of many regions, a building, where a clear difference due to alignment is visible. For better comparison in this region, we show zoomed-up view texturing results in (a). Note that when the initial pose is used, some of the windows get pasted onto the roof.

### B. Frame alignment

We have tried to align point clouds recovered based on MS and SM to the 3D sensor point cloud. In most cases, ICP converges with the dense MS point clouds, but diverges with the sparse SM point clouds.

Based on our experiments, up to 25 meter displacement errors can be corrected by ICP. For example, in Fig. 2, the translational correction of the camera position is  $\sqrt{5^2 + 3^2 + 1.5^2} \approx 7$  meters. Recall that one of the main goals of aligning video onto a geometric model is to texture the geometric model obtained from other sensors such as LiDAR. To verify the alignment accuracy and texturing quality, we should render the textured 3D model from different views

than that of the camera. In Fig. 5, we render the textured 3D model based on the alignments in Figs. 2(a) and (b). Notice that the windows are misaligned to texture the top of the high-rise building in Fig. 5(a). This is clearly incorrect and it has been corrected in Fig. 5(b) after ICP refinement.

For comparison, we have also tried a line-based alignment algorithm [18] on the same data. Unlike the proposed alignment algorithm, a 3D geometry model is required for 2D-3D line matching. To visually inspect the alignment result, we project and overlay 3D lines onto the image. A clear difference can be seen from the plots in Fig. 6 where the line-based algorithm [18] appears to have difficulty in converging from an initial position that is far from the true position.

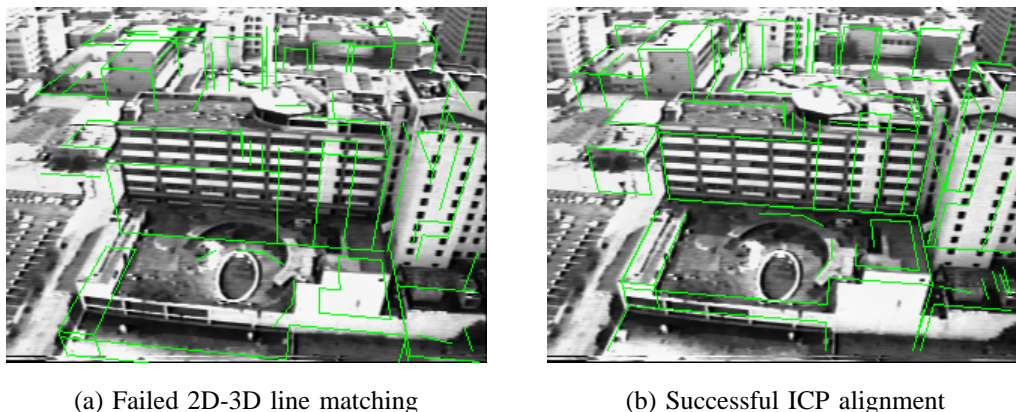


Fig. 6. Projecting and overlaying 3D model lines onto the image for comparison of alignment results. Note that while ICP-based alignment is successful (the same alignment result but different display from that of Fig. 5), line-based alignment [18] fails to converge from an initial position that is far from the true position. **Difference in alignment:** A line-based algorithm needs only one image and a 3D geometry model while ICP needs two images but no 3D model.

1) *Alignment of irregular structures:* One advantage of the proposed framework is the capability to align irregular 3D structures such as trees and bridges that are hard to model. In Fig. 7, we plot the result of aligning a scene that mainly consists of a bridge and trees. Clear improvements due to ICP alignment can be seen by comparing Fig. 7(c) and (d). Again, the alignment is based on 3D point clouds, not a 3D model that is hard to build for irregular structures. However, to make the rendering more realistic, we manually created an over-simplified 3D model for the bridge.

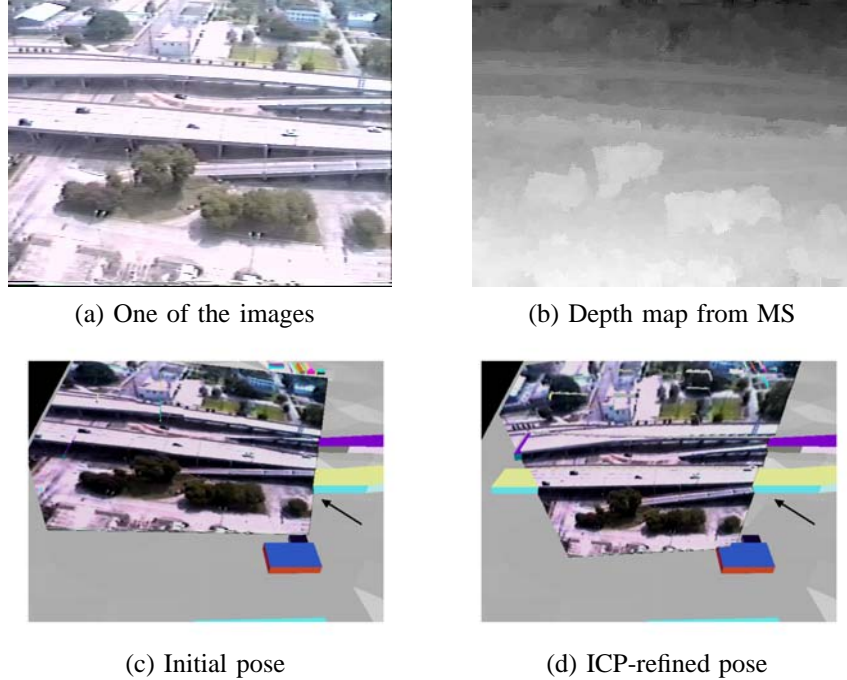
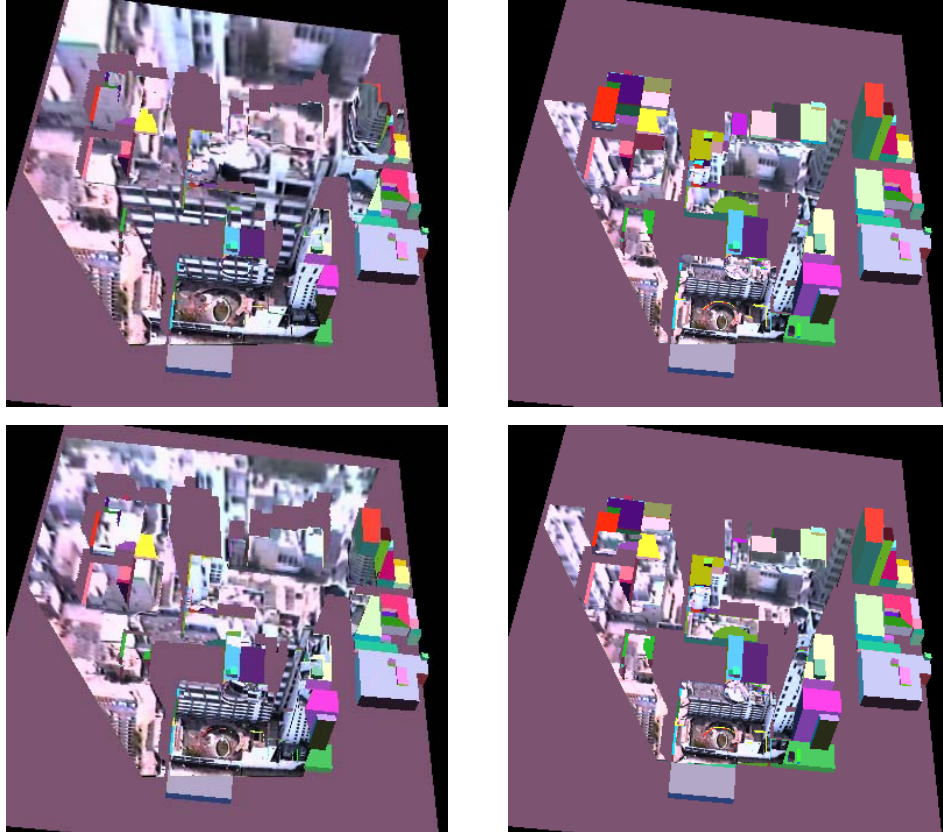


Fig. 7. Aligning irregular structures: (a) One image of the image pair; (b) Depth map computed from MS; Projecting image (a) onto a 3D model based on (c) initial camera pose and (d) ICP-refined pose. To better visualize the projection, we manually created an over-simplified model for the bridge in the scene. Small arrows indicate the place where a clear difference due to alignment is visible.

2) *Alignment sensitivity to calibration:* There are two issues related to camera calibration. One is the quality of 3D reconstruction. For example, only perfect focal length could lead to perfect 3D reconstruction (lens distortion is not considered in this paper). Here we are more interested in the other issue, the alignment. To test the robustness of the proposed pose-stereo-ICP framework to inaccurate focal length, we have tried different focal lengths with the same sets of data. The sensitivity depends on the data content. For scenes of very oblique view or without rich 3D structure, inaccurate focal length can make the ICP alignment to converge to an incorrect local minimum. For scenes with rich 3D structure, alignments based on a focal length within a range of 15% from the nominal value give reasonable rendering results (lens distortion is not corrected) that are better than results prior to alignments (Fig. 8).

3) *Alignment sensitivity to scene:* As pointed out in Section III-C, the proposed frame alignment could fail, especially when a moving monocular camera is viewing small 3D structures or scenes of weak geometry with respect to its field of view. This is because practicably inevitable errors in 3D reconstruction could overwhelm the small/weak true 3D structures in terms of the



(a) Inaccurate focal length (15% smaller) (b) Inaccurate focal length (15% larger)

Fig. 8. Impact of inaccurate focal length upon alignment. The top row plots are texturing results before ICP alignment while the bottom row plots are after ICP alignment. In the first column (a), a focal length that is 15% smaller than the nominal value is used. In the second column (b), a focal length that is 15% larger than the nominal value is used. Note that in both cases, significant improvements on camera pose are obtained. This is most prominent in the high-rise building region.

influence upon alignment. Figure 9 shows such an example where frame alignment failed.

### C. Extension to video alignment

We have worked with two video sequences of different lengths from the Baton Rouge data set. The short sequence **A** has 100 frames and the long sequence **B** has about 300 frames. For each video sequence, we choose two key frames that are nine frames apart and compute motion stereo after the poses have been recovered. We then move along the video sequence to select another pair of key frames.



ICP-refined pose      Bundle-adjusted pose  
 (a) Alignment comparison in the building region



(b) One of the images      (c) Texturing result based on ICP-refined pose

Fig. 9. Failure example of frame alignment when recovered region only contains small 3D structures/weak geometry with respect to the camera’s field of view. Please note that point clouds are not available for walls (partially visible walls of the large high-rise building) from the 3D sensor due to its nadir viewing angle. Notice that bundle adjustment is able to propagate the good alignment results from other key-frames to correct for this failed frame alignment.

### Rendering of the textured 3D model

For sequence **A**, we already showed one result for a key frame in Fig. 5. Now we give one result for non-key frame in Fig. 10 where no ICP alignment is carried out. Instead, bundle adjustment is able to propagate the good alignment results from close-by key-frames. Bundle adjustment can also be used to correct for erroneous key-frame alignment. In Fig. 9 ICP alignment failure in the very beginning of sequence **A** is corrected by bundle adjustment.

Sequence **B** covers an area 750 meters long on the ground. Figure 11 shows the result of overlaying five aligned images onto a 3D model. As usual, the improvement in the estimate of individual camera positions is in the range of tens of meters. To consider the accumulated error in camera pose estimation (Section III-B), we measure the error in the distance between the footprints of the first frame and the last frame. The error in this distance is estimated to be around 120 meters (Fig. 11).

Although the improvement in alignment is significant, the approach has at least two limitations: 1) It is still expensive to process a long video sequence. 2) The estimation of continuous poses

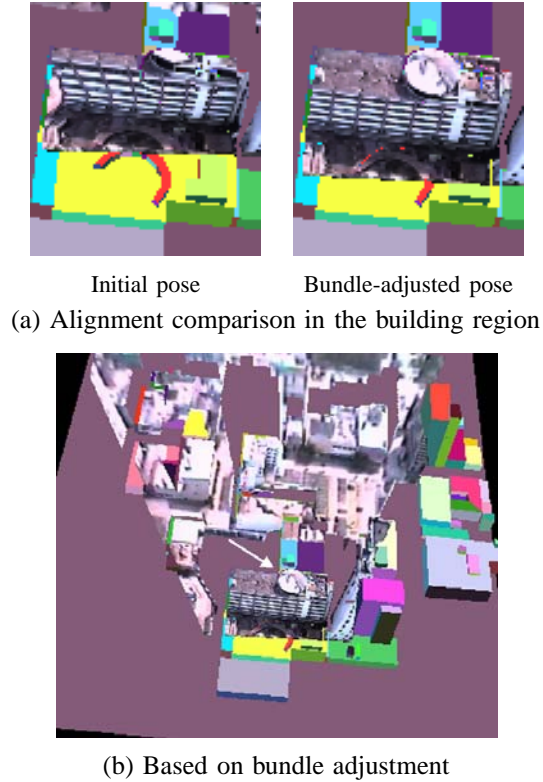
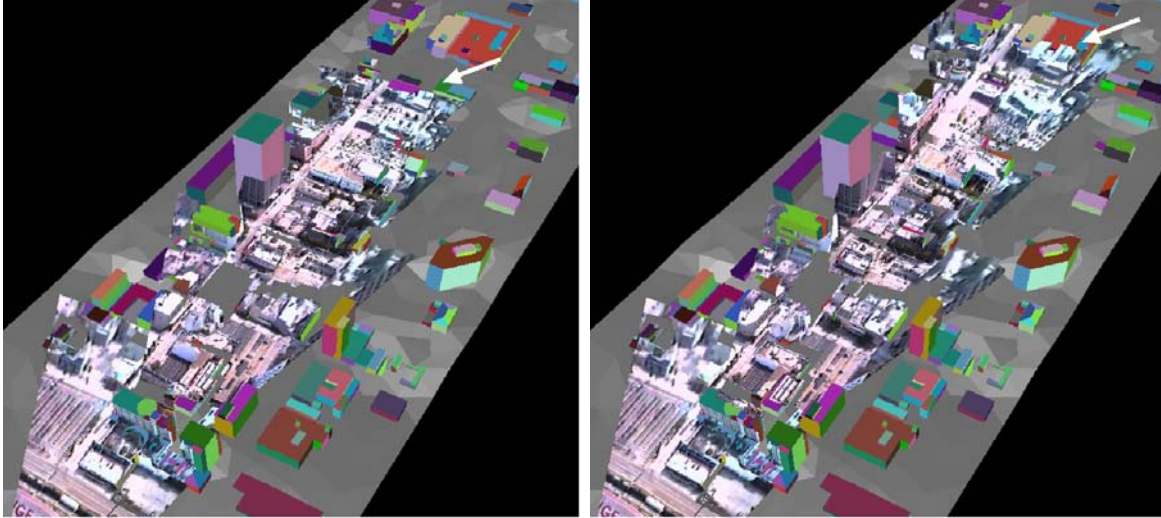


Fig. 10. Sequence A: overlaying intermediate video frames between key frames onto a 3D model. The small arrow in (b) indicates one of many regions, a building, where a clear difference due to alignment is visible. For better comparison in this region, we show zoomed-up view texturing results in (a). Bundle adjustment is able to propagate the good alignment results from close-by key-frames such as those in Fig. 5.

based on ICP-alignment is not as smooth/self-consistent as that from camera pose recovery (Section III-B).

## V. MODELING THROUGH REGISTRATION

In this section, we introduce a novel modeling-through-registration approach that fuses 3D information from both the 3D sensor and the video to build the final 3D model (Fig. 12(a)). For comparison, we also depict the traditional 3D modeling approach in Fig. 12(b). This novel approach is now possible because the 3D model is not required for alignment in the proposed framework. The potential advantages of this modeling approach are two-fold: 1) Missing information from the 3D sensor can be complemented by the video camera; 2) Dynamic modeling is possible that, for example, models new 3D structures appearing only in recently acquired video.



(a) Based on initial pose

(b) Based on continuous video alignment

Fig. 11. Sequence **B**: overlaying five video frames onto a 3D model based on (a) initial pose and (b) video-to-3D alignment. Small arrows indicate the locations where the last frame is projected onto the 3D model. Comparison of the two locations in (a) and (b) indicates that an initial pose based error of around 120 meters, measured as the error in the distance between the footprints of the first frame and the last frame, has been corrected using continuous video alignment.

In the following, we show a real example where point clouds from video and LiDAR are fused nicely to build a model (Fig. 13). Since we can not show the 3D sensor point cloud from our customer, we interactively built a raw model without inferring missing walls and then synthesized the point cloud (Fig. 13(d)). In brief summary, video-inferred 3D information can be used to compensate for the missing information from 3D sensor data.

To directly utilize the 3D geometry information derived from video, we need higher-quality depth than that used for alignment. For example, the MS point cloud based on the plane-plus-parallax framework (Fig. 4(a)) is sufficient for ICP alignment. But it is clear that the 3D points are noisy and sharp edges are not well recovered. To improve the 3D reconstruction of MS, we apply the color-segmentation based stereo algorithm [35] to obtain higher quality stereo (Fig. 13(b)), and hence a higher quality point cloud that preserves sharp edges (Fig. 13(c)). After we use ICP to align the video-inferred point cloud onto the LiDAR point cloud (Fig. 13(d)), we can create a composite point cloud (Fig. 13(e)) and use it to build a 3D model interactively (Fig. 13(f)).

To compare against existing approaches where only 3D sensor data is used in modeling, we

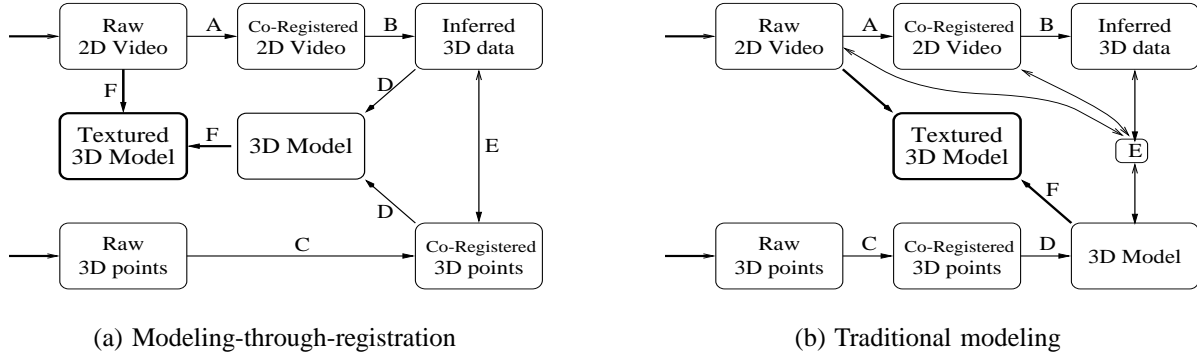


Fig. 12. 3D modeling from 3D sensors and video: (a) The modeling-through-registration approach; (b) The traditional approach. Refer to Fig. 1 for the explanation of Tasks A-E. Task F is to build a photo-realistic 3D model.

plot 3D modeling and texturing results in Fig. 14.

## VI. DISCUSSIONS AND CONCLUSIONS

We have presented a general framework for performing automatic video-to-3D registration and demonstrated results using real video and LiDAR data. Using pairs of frames, significant improvement over the initial camera poses has been obtained by frame alignment. This has been verified quantitatively and through rendering textured 3D models. We then extend the frame alignment to register video sequences onto 3D point clouds. Clear improvements have been demonstrated on real video sequences of different lengths.

### A. Discussions

One immediate issue that we plan to address is how to speed up the process of alignment. Various techniques have been suggested to speed up the alignment process. In [2], speed-up was achieved by extrapolating motion based on the trajectory of previous three iterations. Multi-resolution techniques also exist that decimate point clouds [11] for faster convergence. In [31], a real-time system (100-300ms per frame) was reported that uses 256 point pairs. The idea of motion extrapolation [2] was extended to handle rotation and translation separately. In addition, one key point for speed-up is the use of closest-point cache.

To further improve the alignment accuracy (under the case when model is available), we find that the combination of 3D alignment first and then line-based matching provides both large capturing range and high accuracy. In addition, line-based algorithm can refine the focal length during matching.

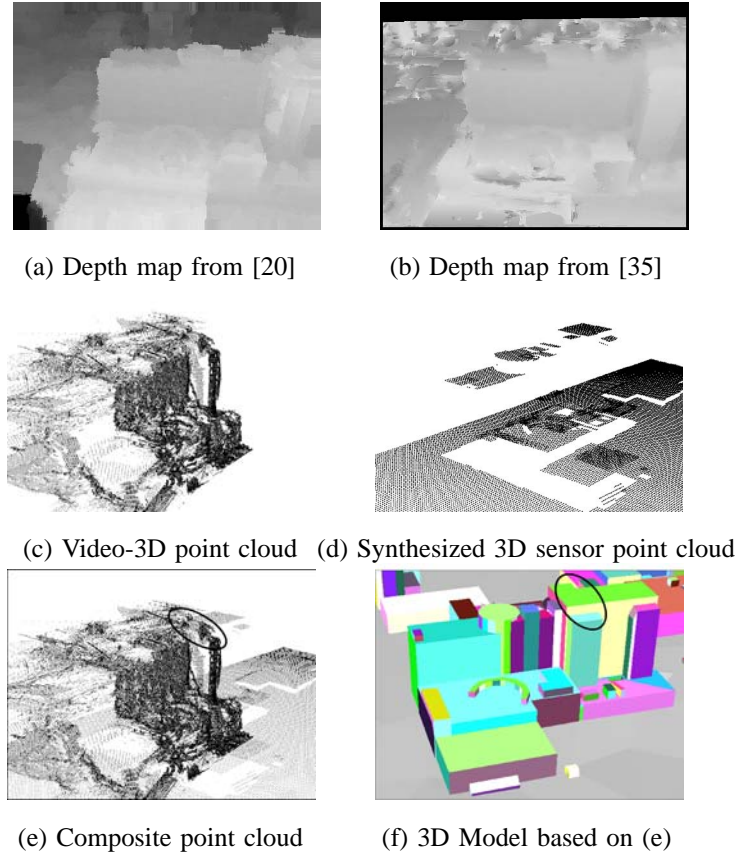


Fig. 13. Combining 3D point clouds from video and 3D sensors. The translational corrections of camera pose are 8, 4 and 1 meters in X-, Y-, and Z-directions respectively. For detailed comparison in the circled region where missing information from 3D sensor has been filled in by the video-inferred point cloud, please see Fig. 14.

### B. Future directions

For future research directions, we would like to address the following issues:

- Automatic refinement of camera focal length through registration. For accurate 3D Euclidean reconstruction, accurate estimate of focal length is necessary in addition to the removal of lens distortion.
- Designing an efficient sequential statistical framework [1], [23] that takes advantage of all information: camera pose estimate, GPS/INS measurements, and ICP-alignment results.
- Designing a unified sequential statistical framework that combines registration and modeling.

### REFERENCES

- [1] B.D.O. Anderson and J.B. Moore, *Optimal Filtering*, NJ:Prentice-Hall Inc. 1979.

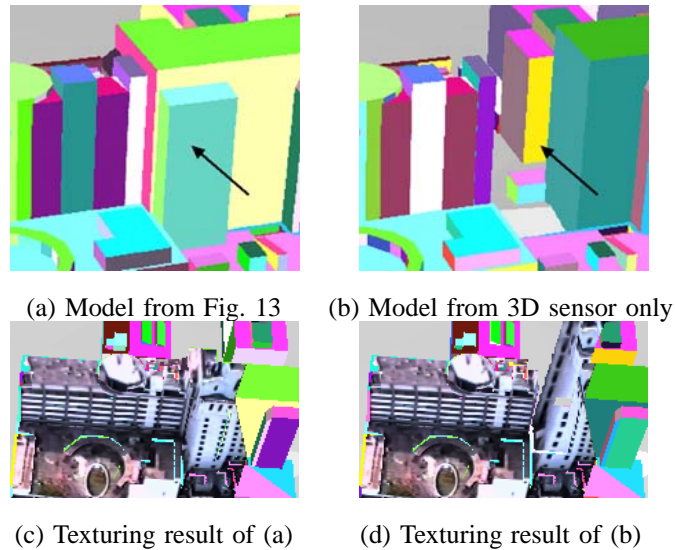


Fig. 14. Close-up comparison of 3D models: (a) model built from both video and 3D sensors, (b) model built from 3D sensor only. Notice that the arrow indicates where the information missing from the 3D sensor data (b) is filled in by video-inferred point cloud (a). To see the impact of improved modeling, we plot the texturing results of (a) and (b) in (c) and (d) respectively.

- [2] P. J. Besl and N. D. McKay, "A method for registration of 3-D shapes," *IEEE Trans. Pattern Anal. Mach. Intell.*, 14, 1992, 239-256.
- [3] R.J. Campbell and P.J. Flynn, "Survey of Free-Form Object Representation and Recognition Techniques," *Computer Vision and Image Understanding*, 81, 166-210, 2001.
- [4] Y. Chen and G. Medioni, "Object modeling by registration of multiple range images," *Image Vision Comput.*, 10, 1992, 145-155.
- [5] C. Chen, Y. Hung, and J. Cheng, "RANSAC-Based DARCES: A New Approach to Fast Automatic Registration of Partially Overlapping Range Images," *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol. 21, No. 11, 1999.
- [6] P.E. Debevec, C.J. Taylor, and J. Malik, "Modeling and Rendering Architecture from Photograph: A hybrid geometry- and image-based approach," *Proc. SIGGRAPH'96*, pp. 11-20, 1996.
- [7] O.D. Faugeras, *Three-Dimensional Computer Vision: a Geometric Viewpoint*. MIT Press, 1993.
- [8] M. Fischler and R. Bolles, "Random Sample Consensus: a Paradigm for Model Fitting with Application to Image Analysis and Automated Cartography," *Commun. Assoc. Comp. Mach.*, 24:381-395, 1981.
- [9] C. Frueh and A. Zakhor, "Constructing 3D City Models by Merging Ground-Based and Airborne Views," in *Proc. CVPR*, 2003.
- [10] G. Godin, M. Rioux, and R. Baribeau, "Three-dimensional registration using range and intensity information," in *Proc. SPIE Videometrics III*, vol. 2350, Boston, pp. 279-290, 1994.
- [11] S. Granger and X. Pennec, "Multi-scale EM-ICP: A Fast and Robust Approach for Surface Registration," in *Proc. ECCV*, 2002.
- [12] C. Brenner, N. Haala, and D. Fritsch, "Towards fully automated 3D city model generation," *Workshop on Automatic Extraction of Man-Made Objects from Aerial and Space Images III*. 2001.
- [13] N. Haala and C. Brenner, "Generation of 3D city models from airborne laser scanning data," in *Proc. EARSEL Workshop on LIDAR Remote Sensing on Land and Sea*, Tallin, Esonia, pp. 105-112, 1997.

- [14] C. Harris and M. Stephens, "A combined corner and edge detector," in *Proc. Fourth Alvey Vision Conference*, pp. 147-151, 1998.
- [15] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, Cambridge University Press, 2000.
- [16] M. Harville, A. Rahimi, T. Darrell, G. Gordon, and J. Woodfill, "3D Pose Tracking with Linear Depth and Brightness Constraints," in *Proc. ICCV*, 1999.
- [17] B.K.P. Horn, "Closed-form solution of absolute orientation using unit quaternions," *Journal of Optical Society of America A*, vol 4, pp 629-642.
- [18] S. Hsu, S. Samarasekera, R. Kumar, and H. Sawhney, "Pose Estimation, Model Refinement, and Enhanced Visualization Using Video", in *Proc. CVPR*, 2000.
- [19] M. Irani and P. Anandan, "Parallax Geometry of Pairs of Points for 3D Scene Analysis," in *Proc. ECCV*, 1996.
- [20] R. Kumar, P. Anandan, and K. Hanna, "Direct Recovery of shape from multiple views: a parallax based approach," in *Proc. ICPR*, 1994.
- [21] D. Lowe, "Fitting parameterized three-dimensional models to images," *IEEE Trans. on PAMI*, Vol. 13, pp. 441-450.
- [22] T. Masuda, K. Sakaue, and N. Yokoya, "Registration and Integration of Multiple Range Images for 3D Model Construction," in *Proc. CVPR*, 1996.
- [23] L. Matthies, T. Kanade, and R. Szeliski, "Kalman Filter-based Algorithms for Estimating Depth from Image Sequences," *International Journal of Computer Vision*, No. 3, pp. 209 - 236, 1989.
- [24] D. M. Mount and S. Arya. "ANN: A library for approximate nearest neighbor searching," Center for Geometric Computing 2nd Annual Workshop on Computational Geometry, 1997.
- [25] D. Nister, "An Efficient Solution to the Five-Point Relative Pose Problem," *IEEE Trans. on PAMI*, Vol. 26, pp. 756-770.
- [26] D. Nister, "Reconstruction from UNcalibrated Sequences with a Hierarchy of Trifocal Tensors," in *Proc. ECCV*, 2000.
- [27] U. Neumann et al, "Augmented Virtual Environments (AVE): Dynamic Fusion of Imagery and 3d Models," in *Proc. Virtual Reality Conference*, 2003.
- [28] J. Rodriguez and J.K. Aggarwal, "Matching Aerial Images to 3-D Terrain Maps," *IEEE Trans. on PAMI*, Vol. 12, pp. 1138-1149, 1990.
- [29] S. Rusinkiewicz and M. Levoy, "Efficient Variants of the ICP Algorithm," in *Proc. Third International Conference on 3D Digital Imaging and Modeling*, 2001.
- [30] H. Sawhney, A. Arpa, R. Kumar, S. Samarasekera, M. Aggarwal, H. Hsu, D. Nister, and K. Hanna, "Video Flashlights – Real Time Rendering of Multiple Videos for Immersive Model Visualization," in *Proc. Eurographics Workshop on Rendering Techniques*, pp. 157-168, 2002.
- [31] D. Simon, M. Hebert, and T. Kanade, "Real-time 3-D Pose Estimation Using a High-Speed Range Sensor," in *Proc. IEEE International Conf. Robotics and Automation*, Vol. 3, pp. 2235-2241, 1994.
- [32] V. Sequiera, K. Ng, E. Wolfart, J. Concalves, and D. Hogg, "Automated reconstruction of 3D models from real environments," *ISPRS J. Photogrammetry Remote Sensing*, Vol. 54, pp. 1-22, 1999.
- [33] I. Stamos and P. Allen, "Geometry and Texture Recovery of Scenes of Large Scale," *Computer Vision and Image Understanding*, Vol. 88, pp. 94-118, 2002.
- [34] I. Stamos and P. Allen, "Automatic Registration of 2-D with 3-D Imagery in Urban Environments," in *Proc. ICCV*, 2001.
- [35] H. Tao, H. Sawhney, and R. Kumar, "A global matching framework for stereo computation," in *Proc. ICCV01*, 2001.
- [36] T.Y. Tian, C. Tomas, and D. Heeger, "Comparison of Approaches to Egomotion Computation," in *Proc. CVPR*, 1996.
- [37] Bill Triggs, Philip McLauchlan, Richard Hartley, and Andrew Fitzgibbon 2000. "Bundle Adjustment - A modern Synthesis," B. Triggs, A. Zisserman, R. Szeliski (Eds.): *Vision Algorithms'99*, pp. 298-372.
- [38] P. Viola and W. Wells, "Alignment by Maximization of Mutual Information," *International Journal of Computer Vision*, 24(2), pp. 137-154, 1997
- [39] Z. Zhang, "Iterative point matching for registration of free-form curves and surfaces," *International Journal of Computer Vision*, 13, pp. 119-152, 1994,