REAL-TIME VIDEO-BASED RECONSTRUCTION OF URBAN ENVIRONMENTS

P. Mordohai[1], J.-M. Frahm[1], A. Akbarzadeh[2], B. Clipp[1], C. Engels[2], D. Gallup[1], P. Merrell[1], C. Salmi[1], S. Sinha[1], B. Talton[1], L. Wang[2], Q. Yang[2], H. Stewénius[2], H. Towles[1], G. Welch[1], R. Yang[2], M. Pollefeys[1] and D. Nistér[2]

[1]University of North Carolina, Chapel Hill, NC, USA
{mordohai, jmf, bclipp, gallup, pmerrell, salmi, taltonb, herman, welch, marc}@cs.unc.edu
[2] University of Kentucky, Lexington, KY, USA
{amir, engels, qingxiong.yang, stewe}@vis.uky.edu, {lwangd, ryang, dnister}@cs.uky.edu

KEY WORDS: computer vision, 3D model, stereo reconstruction, large-scale reconstruction, real time processing, geo-registration

ABSTRACT:

We present an approach for automatic 3D reconstruction of outdoor scenes using computer vision techniques. Our system collects video, GPS and INS data which are processed in real-time to produce geo-registered, detailed 3D models that represent the geometry and appearance of the world. These models are generated without manual measurements or markers in the scene and can be used for visualization from arbitrary viewpoints, documentation and archiving of large areas. Our system consists of a data acquisition system and a processing system that generated 3D models from the video sequences off-line but in real-time. The GPS/INS measurements allow us to geo-register the pose of the camera at the time each frame was captured. The following stages of the processing pipeline perform dense reconstruction and generate the 3D models, which are in the form of a triangular mesh and a set of images that provide texture. By leveraging the processing power of the GPU, we are able to achieve faster than real-time performance, while maintaining an accuracy of a few cm.

1 INTRODUCTION

Detailed 3D models of cities were usually made from aerial data, in the form of range or passive images combined with other modalities, such as measurements from a Global Positioning System (GPS). While these models can be useful for navigation, they provide little additional information compared to maps in terms of visualization. Buildings and other landmarks cannot be easily recognized since the facades are poorly reconstructed from aerial images due to highly oblique viewing angles. To achieve high-quality ground-level visualization one needs to capture data from the ground.

Recently, the acquisition of ground-level videos of urban environments using cameras mounted on vehicles has been the focus of research and commercial initiatives, such as Google Earth and Microsoft Virtual Earth, that aim at generating high-quality visualizations. Such visualizations can be in the form of panoramic mosaics (Teller et al., 2003; Román et al., 2004), simple geometric models (Cornelis et al., 2006) or accurate and detailed 3D models. The construction of mosaics and simple geometric models is not very computationally intensive, but the resulting visualizations are either as simple as the employed models or limited to rotations around pre-specified viewpoints. In this paper, we investigate the reconstruction of accurate 3D models from long video sequences with emphasis on urban environments. These models offer considerably more flexible visualization in the form of walk-throughs and fly-overs. Moreover, due to the use of GPS information, the models are accurately geo-located and measurements with a precision of a few cm can be made on them. Screenshots of our 3D reconstructions can be seen in Figure 1. The massive amounts of data needed in order to generate 3D models of complete cities pose significant challenges for both the data collection and processing systems, which we address in this paper.

We introduce a large-scale, 3D reconstruction system that is able to reconstruct detailed 3D models in the form of textured polygonal meshes at speeds that are close to or exceed real-time. The data acquisition part of the system collects video streams from up to eight cameras, as well as data from a Global Positioning System (GPS) and an Inertial Navigation System (INS). Our approach uses computationally efficient components and computation strategies to achieve real-time processing for the enormous amounts of video data required to reconstruct entire cities.

We have developed a novel two-stage strategy for dense 3D reconstruction that allows us to achieve high processing rates. By decoupling the problem into the reconstruction of depth maps from sets of images followed by the fusion of these depth maps, we are able to use simple fast algorithms that can be implemented on the Graphics Processing Unit (GPU). The depth maps are reconstructed using an extended plane sweeping stereo technique operating on frames of a single video-camera (Gallup et al., 2007). Since this stereo algorithm does not perform costly global optimization, depth maps often include erroneous depth estimates for

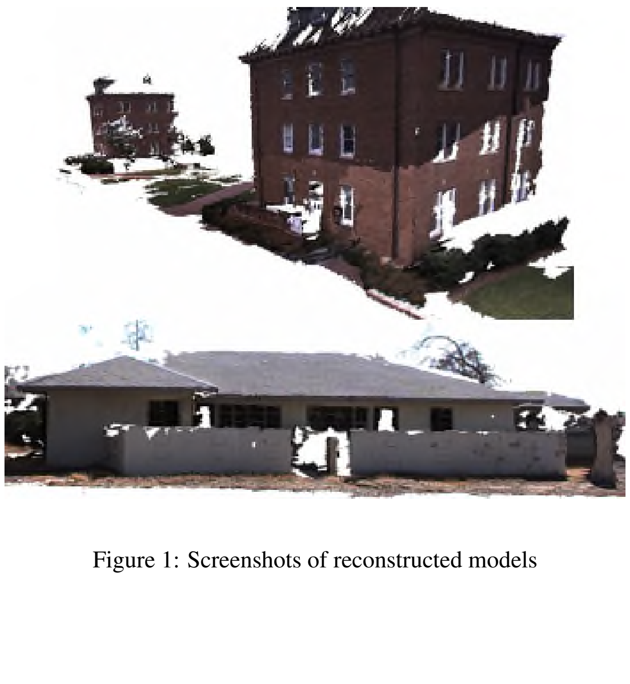


Figure 1: Screenshots of reconstructed models

several pixels. We address these errors in a depth map fusion step that combines multiple depth maps into one consistent depth map, thus increasing accuracy and decreasing redundancy. It delivers a compact and geometrically consistent representation of the 3D scene, which is especially important for long video sequences. Figure 1 shows screenshots of representative models reconstructed by our system.

The paper is organized as follows: the next section briefly reviews some related work; Section 3 is an overview of the complete system; Section 4 presents the data acquisition platform; Section 5 discusses pose estimation using the GPS/INS data; Section 6 describes the core of the stereo reconstruction module; Section 7 shows the principles of our approach for depth map fusion; Section 8 discusses the mesh generation process; Section 9 shows experimental results; Section 10 describes quantitative results using our evaluation method and Section 11 concludes the paper.

2 RELATED WORK

The research community has devoted a lot of effort to the modeling of man-made environments using both active and passive sensors. Here, we briefly review work relying on ground-based imaging since it is more closely related to our project. An equal, if not larger, volume of work exists for aerial imaging. The goal is accurate reconstruction of urban or archaeological sites, including both geometry and texture, in order to obtain models useful for visualization, for quantitative analysis in the form of measurements at large or small scales and potentially for studying their evolution through time.

Combining active range scanners and digital cameras is a design choice that satisfies the requirement of simultaneously modeling the geometry and appearance of the environment. Stamos and Allen (2002) used such a combination, while also addressing the problems of registering the two modalities, segmenting the data and fitting planes to the point cloud. El-Hakim et al. (2003) propose a methodology for selecting the most appropriate modality among range scanners, ground and aerial images and CAD models. Früh and Zakhor (2004) developed a system that is similar to ours since it is also mounted on a vehicle and captures large amounts of data in continuous mode, in contrast to the previous approaches of (Stamos and Allen, 2002; El-Hakim et al., 2003) that scan the scene from a set of pre-specified viewpoints. A system with similar configuration, but smaller size, that also operates in continuous mode was presented by Biber et al. (2005).

Laser scanners have the advantage of providing accurate 3D measurements directly. On the other hand, they can be cumbersome and expensive. Researchers in photogrammetry and computer vision address the problem of reconstruction relying solely on passive sensors (cameras) in order to increase the flexibility of the system while decreasing its size, weight and cost. The challenges are due mostly to the well-document inaccuracies in 3D reconstruction from 2D measurements. This is a problem that has received a lot of attention from computer vision and photogrammetry. Due to the size of the relevant literature we refer interested readers to the books by Faugeras (1993) and Hartley and Zisserman (2000) for computer vision based treatments and the Manual of Photogrammetry (American Society of Photogrammetry, 2004) for the photogrammetric approach. It should be noted here that while close-range photogrammetry solutions would be effective for 3D modeling of buildings, they are often interactive and would not efficiently scale to a complete city.

Among the first attempts for 3D reconstruction of large scale environments using computer vision techniques was the MIT City Scanning project (Teller, 1998). The goal of the project was the construction of panoramas from a large collection of images of the MIT campus. A semi-automatic approach under which simple geometric primitives are fitted to the data was proposed by Debevec et al. (1996). Compelling models can be reconstructed even though fine details are not modeled but treated as texture instead. Rother and Carlsson (2002) showed that multiple-view reconstruction can be formulated as a linear estimation problem given a known fixed plane that is visible in all images. Werner and Zisserman (2002) presented an automatic method, inspired by (Debevec et al., 1996), that fits planes and polyhedra on sparse reconstructed primitives by examining the support they receive via a modified version of the space sweep algorithm (Collins, 1996). Research on large scale urban modeling includes the 4D Atlanta project described by Schindler and Dellaert (2004), which also examines the evolution of the model through time. Recently, Cornelis et al. (2006) presented a system for real-time modeling of cities that employs a very simple model for the geometry of the world. Specifically, they assume that the scenes consist of three planes: the ground and two facades orthogonal to it. A stereo algorithm that matches vertical lines across two calibrated cameras is used to reconstruct the facades, while objects that are not consistent with the facades or the ground are suppressed.

We approach the problem using passive sensors only, building upon the experience from the intensive study of structure from motion and shape reconstruction within the computer vision community (Pollefeys et al., 1999; Nistér, 2001; Pollefeys et al., 2003, 2004; Nistér et al., 2006). The emphasis in our project is on the development of a fully automatic system that is able to operate in continuous mode without user intervention or the luxury of precise viewpoint selection since capturing is performed from a moving vehicle and thus constrained to the vantage points of the streets. Our system design is also driven by the performance goal of being able to process the large video datasets in a time at most equal to that of acquisition.

3 SYSTEM OVERVIEW

Our complete system consists of the data acquisition system and the processing pipeline. The two parts are separated for various reasons, which include less complex design and development, reduced power, cooling and processing requirements for the acquisition vehicle and increased flexibility in the processing pipeline. The latter refers to providing the user with the option to invoke slower but more accurate processing modules.

The data acquisition system consists of eight cameras, an INS, GPS antennas and a set of hard-drives. It collects data and streams them to the hard drives after very little processing.

Processing entails estimating the vehicle's trajectory, recovering the camera poses at the time instances when frames were captured, computing dense depth maps and finally generating a model in the form of a polygonal mesh with associated texture maps. The dense reconstruction module consists of two stages: multiple-view stereo that computes depth maps given images and the corresponding camera poses and depth map fusion that merges depth maps to increase accuracy and reduce redundancy. We show that this two-stage strategy allows us to achieve very fast performance by implementing fast algorithms on the GPU without compromising the quality of the models. The massive amounts of data needed in order to generate large-scale 3D models pose significant challenges for the processing system, which we address in this paper.

4 DATA ACQUISITION SYSTEM

The on-vehicle video acquisition system consists of two main sub-systems: an eight-camera digital video recorder and an Applanix GPS/INS (model POS LV) navigation system. The cameras are connected to PC boards that stream the raw images to disk, and the Applanix system tracks the vehicle's position and orientation. The DVR is built with eight Point Grey Research Flea 1024×768 color cameras, with one quadruple of cameras for each side of the vehicle. Each camera has a field-of-view of approximately $40^\circ \times 30^\circ$, and within a quadruple the cameras are arranged with minimal overlap in field-of-view. Three cameras are mounted in the same plane to create a horizontal field of view of approximately 120°. The fourth camera is tilted upward to create an overall vertical field of view of approximately 60° with the side-looking camera. The cameras are referred to as the forward, side, backward and upward camera in the remainder of the paper.

The CCD exposure on all cameras is synchronized using hardware synchronization units and timestamps are recorded with the images. The GPS timestamps are used to recover the position and orientation of the vehicle at the corresponding time from the post-processed trajectory. Recovering camera positions and orientations, however, requires knowledge of the hand-eye calibration between the cameras and the GPS/INS system that one establishes during system calibration.

The first stage of calibration is the estimation of the intrinsic parameters of the camera using a planar calibration object and applying the algorithm of Zhang (2000). We do not attempt to estimate the external relationships of cameras since there is little or no overlap in their fields of view. Instead, we estimate the calibration of the cameras relative to the GPS/INS coordinate system. This process is termed *hand-eye* or *lever-arm* calibration. The hand-eye transformation, which consists of a rotation and a translation, between the camera coordinate system and the vehicle coordinate system is computed by detecting the positions of a few surveyed points in a few frames taken from different viewpoints. The pose of the camera can be determined for each frame from the projections of the known 3D points and the pose of the vehicle is given by the sensor measurements. The optimal linear transformation can be computed using this information. This allows us to compute the positions of cameras and 3D points in UTM coordinates given by the GPS sensor. After the hand-eye transformation for each camera has been computed, all 3D positions and orientations of points, surfaces or cameras can be transformed to UTM coordinates. This is accomplished using the timestamps, which serve as an index to the vehicle's trajectory, to retrieve the vehicle's pose at the time instance an image was captured. We can, then, compute the pose of a particular camera by applying the appropriate hand-eye transformation to the vehicle pose.

5 POSE ESTIMATION

In the first stage of processing, the GPS/INS data are filtered to obtain a smooth estimate of the vehicle's trajectory. Since these signals are collected at $200Hz$, we can recover the position and orientation of the vehicle at the time instance a frame was captured with high accuracy. Estimates of position and orientation are obtained by interpolating the trajectory between the dense sample positions.

Precise camera poses are critical for 3D reconstruction. We pursue three options for pose estimation: one entirely based on visual tracking which has reduced hardware requirements, but cannot determine the scale or geo-location of the models and suffers from drift; one that uses accurate GPS/INS data only; and one that combines vision-based tracking with the GPS/INS measurements. The first option suffers from drift especially when applied on very long video sequences, such as the ones we show here that consist of several thousand frames. The third option is computationally more expensive and makes small improvements in pose estimation. In addition, it is a safeguards against rare errors in the GPS/INS trajectory. Here we employ the second option of relying solely the GPS/INS data for pose estimation. Our experiments and quantitative evaluations against surveyed buildings (see Section 10) have shown that this option allows us to obtain accurate 3D reconstructions in real time.

We require that the baseline between two consecutive frames exceeds a certain threshold before a frame is accepted for further processing. If the baseline is too short, the frame is by-passed and the next frame is examined. This step, on one hand, quickly detects whether the vehicle has stopped and prevents unnecessary processing and, on the other, guarantees that images used for stereo have sufficiently different viewpoints. The latter is important to reduce the geometric uncertainty of the reconstruction. Typical values for the minimum baseline between two frames are 10 and $40cm$. We typically use 5 to 15 frames for stereo. For a typical depth range of the scene between 3 to $20m$, the triangulation angles vary between 30° and 7° for 7 frames to the left and right of the reference view spaced at least $35cm$ apart.

6 STEREO RECONSTRUCTION

The stereo reconstruction module takes as input camera poses and the corresponding frames from the video and produces a depth map for each frame. As mentioned in Section 4, the cameras in our system have minimal overlap and their purpose is to provide a wider field of view. Thus, stereo is performed on consecutive frames of a single camera as it moves. It uses a modified version of the plane-sweep algorithm of (Collins, 1996), which is an efficient multi-image matching technique. Conceptually, a plane is swept through space in steps along a predefined direction, typically parallel to the optical axis. At each position of the plane, all views are projected on it. If a point on the plane is at the correct depth, then, according to the brightness constancy assumption, all pixels that project on it should have consistent intensities. We measure this consistency as the sum of absolute intensity differences (SAD). We select the hypothesis with the minimum SAD as the depth estimate for each pixel. One of the main reasons we opted for the plane-sweep algorithm is that it can be easily parallelized and thus is amenable to a very fast GPU implementation as shown by Yang and Pollefeys Yang and Pollefeys (2005).

6.1 Plane-Sweeping Stereo with Multiple Sweeping Directions

We have made several augmentations to the original algorithm to improve the quality of the reconstructions and to take advantage of the structure exhibited in man-made environments, which typically contain planar surfaces that are orthogonal to each other. Due to lack of space, we refer interested readers to (Gallup et al., 2007) for more details on analysis of sparse information and methods for selecting the sweeping directions and planes for stereo. Sparse features are not reconstructed by the system described in this paper since the poses are computed from the GPS/INS data. Thus, we rely on simple heuristics to determine the optimal directions in which to sweep planes. The ground is typically approximately horizontal and very often visible in the videos we process. Thus, we sweep horizontal planes at varying heights. We also make the assumptions that the facades in an urban environment

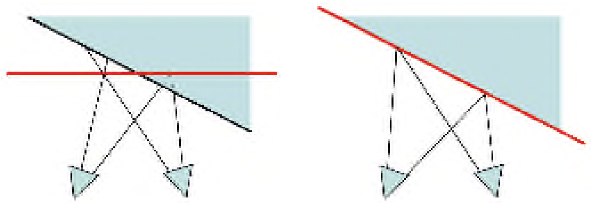


Figure 2: Implications of cost aggregation over a window. Left: Slanted surfaces with fronto-parallel plane-sweeping. Not all points over the window are in correspondence. Right: Surface-aligned sweeping plane to handle slanted surfaces correctly.

are typically parallel to the street, that buildings are orthogonal and that streets also intersect at right angles. Under these assumptions, the most likely vertical planes in the scene are either parallel or orthogonal to the trajectory of the vehicle. Violations of these assumptions, however, do not significantly degrade the quality of the reconstruction.

Sweeping planes along three orthogonal directions, even if the scene surfaces are not parallel to any of them, improves the results of stereo since it reduces misalignments between the plane on which the cost is evaluated and the actual surface. Alignment with the surface is necessary because the pixel dissimilarity measure is aggregated in windows. Since the measurement at a single pixel is in general very sensitive to noise, several measurements from neighboring pixels are combined by summing the absolute intensity differences in a window centered at the pixel under consideration. This reduces sensitivity to noise but introduces an additional requirement for the stereo system: in order to detect the best correspondence for the current pixel, all pixels in the window need to be in correspondence. This is not always the case in many stereo algorithms that use windows of constant disparity since these are optimal only for surfaces that are parallel to the image plane (fronto-parallel). Figure 2 shows the problems caused by sweeping fronto-parallel planes when the scene contains non-fronto-parallel surfaces and our proposed solution, which is to modify the direction of the sweep to match that of the scene surface. Even if the matching is not accurate, increasing the number of sweeping directions reduces this type of errors.

Once the sweeping directions have been selected, we generate a family of planes for each, according to:$\Pi_m = \left[\begin{array}{cc} n_m^T & -d_m \end{array} \right]$ for $m = 1, \dots, M$, where n_m is the unit length normal of the plane and d_m is the distance of the plane to the origin which is set at the center of the camera when the reference view was captured. The range $[d_{near}, d_{far}]$ can be determined either by examining the points obtained from structure from motion or by applying useful heuristics. For example, in outdoor environments, it is usually not useful for the ground plane family to extend above the camera center. The spacing of the planes in the range is uniform in the majority of stereo algorithms. However, it is best to place the planes to account for image sampling (pixels). Ideally, when comparing the respective image warpings induced by consecutive planes, the amount of pixel motion should be less than or equal to 1 according (Szeliski and Scharstein, 2004). This is particularly important when matching surfaces that exhibit high-frequency texture.

In addition to the reference image, N other images are used as target images for the computation of each depth map. Typical values of N are between 4 and 14. We denote their projection matrices P_k and use P_{ref} for the reference image. The reference camera is assumed to be the origin of the coordinate system and so its projection matrix is $P_{ref} = K_{ref} \left[\begin{array}{cc} \mathcal{I}_{3\times 3} & 0 \end{array} \right]$. In order to test the plane hypothesis Π_m for a given pixel (x, y) in the reference view I_{ref}, the pixel is projected into the other images $k = 1, \dots, N$. The planar mapping from the image plane of the reference image P_{ref} to the image plane of the camera P_k can be described by the homography H_{Π_m, P_k} induced by the plane Π_m:

$$H_{\Pi_m, P_k} = K_k \left(R_k^T + \frac{R_k^T C_k n_m^T}{d_m} \right) K_{ref}^{-1}. \qquad (1)$$

where R_k and C_k are the rotation and translation of the camera at frame k with respect to the reference frame P_{ref}, K_k is the corresponding camera calibration matrix (allowing for varying but known intrinsic parameters between frames), n_m is the normal of the plane and d_m is its distance from the origin. The location (x_k, y_k) in image I_k of the mapped reference pixel (x, y) is computed by:

$$\left[\begin{array}{ccc} \tilde{x} & \tilde{y} & \tilde{w} \end{array} \right]^T = H_{\Pi_m, P_k} \left[\begin{array}{ccc} x & y & 1 \end{array} \right]^T \qquad (2)$$
$$\text{with } x_k = \tilde{x}/\tilde{w}, \quad y_k = \tilde{y}/\tilde{w}.$$

If the plane is close to the surface that projects to pixel (x, y) in the reference view and assuming the surface is Lambertian, the colors of $I_k(x_k, y_k)$ and $I_{ref}(x, y)$ are similar. We use the absolute intensity difference as the dissimilarity measure and aggregate several measurements in a window $W(x, y)$ centered at the pixel (x, y). To increase robustness against occlusion, we adopt an algorithm proposed by Kang et al. (2001). For each pixel we compute the cost for each plane using the left and right subset of the images and select the minimum as the cost of the pixel. This scheme is very effective against occlusions, since typically the visibility of a pixel changes at most once in a sequence of images. Therefore, the pixel should be visible in either the entire left of right set of images. The depth for each pixel is chosen by simply assigning it to the plane with the minimum SAD.

6.2 GPU Implementation of Plane-Sweeping Stereo

To achieve real-time performance, we implemented our stereo algorithm on the GPU. Plane-sweeping stereo involves numerous rendering operations (the mapping of a region from an image to another via the homographies) and the GPU is optimized for this type of operations. Cost aggregation for each pixel is performed by boxcar filtering which can also be performed efficiently by the GPU.

Images captured by our system, as with most cameras, are affected by radial distortion of the lenses. Stereo algorithms assume that the images are corrected for radial distortion. An additional benefit that comes from using the GPU for stereo computations is that correction for radial distortion can be performed with negligible increase in processing time. In our processing pipeline the only step that requires the radially undistorted images is stereo. Therefore, we decided to compensate for it only in stereo processing. As each new image is loaded on the GPU for stereo processing, radial undistortion is accomplished by a vertex shader. The shader performs a non-linear mapping with bilinear interpolation that essentially moves the pixels to the positions they would have if the camera obeyed the pinhole model. This step produces images without radial distortion and all subsequent operations can be performed linearly.

Performing plane-sweeping on the GPU using 7 512×384 images, including the reference image, and 48 planes takes $24ms$ on an NVidia GeForce 8800 series card. While an increased number of planes and/or images results in improved reconstruction quality, these settings produce depth maps of satisfactory quality in real-time. Quality is significantly improved in the depth map fusion stage that is described in the following section.

7 DEPTH MAP FUSION

Due to the speed of the computation, the raw stereo depth maps contain errors and do not completely agree with each other. These conflicts and errors are identified and resolved in the depth map fusion stage. In this step, a set of Q depth maps from neighboring camera positions are combined into a single depth map for one of the views. The end result is a fused depth map from the perspective of one of the original viewpoints, which is called the reference view. The central image in the set is typically used as the reference view. An additional benefit of the fusion step is that it produces a more compact representation of the data because the number of fused depth maps is a fraction of the number of the input raw depth maps. Much of the information in the original depth maps is redundant since many of the closely-spaced viewpoints observe the same surface. We opted for this type of viewpoint-based approach since its computational complexity has a fixed upper-bound, for a given Q, regardless of the size of the scene to be modeled. Many of the surface fusion algorithms reported in the literature are limited to single objects and are not applicable to our datasets due to computation and memory requirements.

At each fusion step, there are a total of Q depth maps to be fused denoted by $D_1(\mathbf{x}), D_2(\mathbf{x}), \ldots, D_Q(\mathbf{x})$ which record the depth estimate at pixel $\mathbf{x}$[†] at each viewpoint. One of the viewpoints, typically the central one, is selected as the reference viewpoint. The fusion algorithm returns an estimate of the depth of each pixel of the reference view. The current estimate of the 3D point seen at pixel $\mathbf{x}$ of the reference view is called $\hat{F}(\mathbf{x})$. $R_i(\mathbf{X})$ is the distance between the center of projection of viewpoint i and the 3D point $\mathbf{X}$.

The first step of fusion is to render each depth map into the reference view. When multiple depth values are projected on the same pixel, the nearest depth is kept. Let $D_i^{ref}(\mathbf{x})$ be the depth map from D_i rendered into the reference view at pixel $\mathbf{x}$. We also need a notation for the depth value stored at the pixel of depth map D_i on which a 3D point $\mathbf{X}$ projects. This depth is in general different from the distance of $\mathbf{X}$ from the camera center if $\mathbf{X}$ has not been generated by camera i. Let $P_i(\mathbf{X})$ be the image coordinates where the 3D point $\mathbf{X}$ is seen in viewpoint i. To simplify the notation, the following definition is used $D_i(\mathbf{X}) \equiv D_i(P_i(\mathbf{X}))$ which in general is not equal to $R_i(\mathbf{X})$.

Our approach considers three types of visibility relations between the hypothesized depths in the reference view and the depths originating from other views. These are illustrated in Figure 3. In view i, the observed surface passes behind A as A' is observed in this view. There is a conflict between these two measurements since view i would not be able to observe A' if there was truly a surface at A. We say that A violates the free space of A'. This occurs when $R_i(A) < D_i(A)$. In Figure 3, B' is in agreement with B as they are in the same location. In practice, we define points B and B' as being in agreement when $\frac{|R_i(B)-R_i(B')|}{R_i(B)} < \epsilon$. In the reference view C', rendered from view i, is in front of C. There is a conflict between these two measurements since it would not be possible to observe C if there was truly a surface at C'. We say that C' occludes C. This occurs when $D_i^{ref}(\mathbf{x}) < D_{ref}(\mathbf{x})$. Note that occlusions are defined on rays of the reference view and free space violations with respect to rays of other depth maps.

Taking into account the visibility relations introduced in the previous paragraph, the depth maps can be fused as follows. For each pixel, we count for each candidate depth, how many views result in an occlusion and how many result in a free-space violation. We retain the closest depth for which the number of occlusions is larger or equal to the number of free-space violations. This depth is balanced in the sense that the amount of evidence that indicates it is too close is equal to the amount of evidence that indicates it is too far away.

[†]$\mathbf{x}$ is used in this section instead of (x, y) to denote pixel coordinates in order to simplify the notation

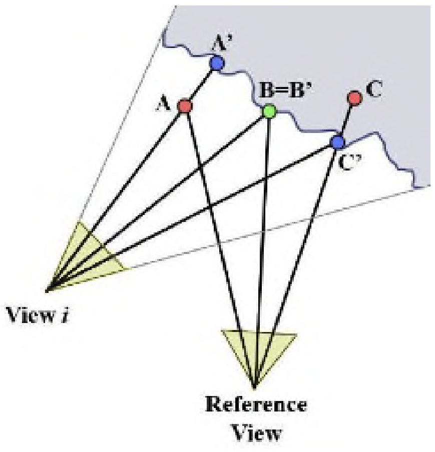


Figure 3: The point A' seen in view i has its free space violated by the point A seen in the reference view. Point B' supports the point B'. Point C seen in the reference view is occluded by point C'.

The most frequent type of computations required for this process is rendering a depth map observed in one viewpoint into another viewpoint. These type of computations can be performed very quickly on the GPU in order to achieve real time performance. Our algorithm fuses 11 depth maps in $193ms$ (or $17.5ms$ per frame) on a high-end GPU.

8 MESH GENERATION

The depth map fusion step generates a depth estimate for each pixel of the reference view. Then, each fused depth map is passed to the mesh generation module along with the image taken from that viewpoint. The output of this module is a triangular mesh and the image that is used for texture-mapping. Exploiting the fact that the image plane can serve as reference for both geometry and appearance, we can construct the triangular mesh very quickly. We employ a multi-resolution quad-tree algorithm in order to minimize the number of triangles while maintaining geometric accuracy as in (Pajarola, 2002a). Since the number of triangles that need to be formed and evaluated is much smaller using a top-down approach than a bottom-up alternative, we choose the former. Starting from a coarse resolution, we attempt to form triangles making sure that they correspond to planar parts of the depth map and do not bridge depth discontinuities. If these conditions are not met, the quad, which is formed by two adjacent triangles, is subdivided. The process is repeated on the subdivided quads until the finest resolution, in which a binary decision to add triangles to the model or not is made using the same criteria. A part of a multi-resolution mesh can be seen in Figure 4.

We begin by dividing the reference depth map into fairly large quads and attempt to form triangles using as vertices the 3D reconstructed points corresponding to the corners of the quads. If the triangles pass the planarity test they are kept, otherwise the

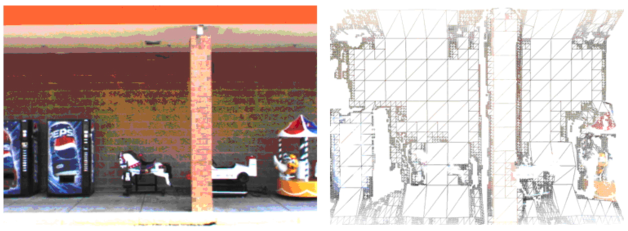
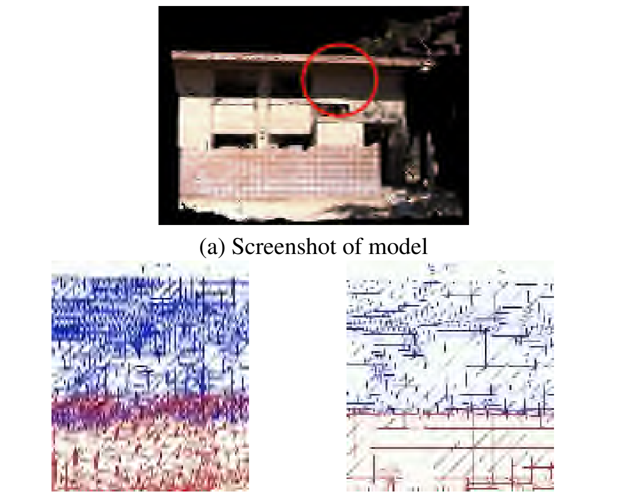


Figure 4: Left: Input image. Right: Illustration of multi-resolution triangular mesh



(a) Screenshot of model

(b) Overlapping triangles (c) After removal of duplicates

Figure 5: Duplicate surface removal for the circled area of the model in (a), which consists of six sub-models, three each from the side and upward cameras. (b) shows the overlapping meshes as wireframes. Triangles in red are generated by the side camera and the ones is blue by the upward camera. The results of duplicate surface removal are shown in (c).

quad is subdivided and the process is repeated at a finer resolution. We use the following simple planarity test proposed in (Pajarola, 2002b) for each vertex of each triangle:

$$\left| \frac{z_{-1} - z_0}{z_{-1}} - \frac{z_0 - z_1}{z_1} \right| < t. \tag{3}$$

Where z_0 is the z-coordinate, in the camera coordinate system, of the vertex being tested and t is a threshold. z_{-1} and z_1 are the z-coordinates of the two neighboring vertices of the current vertex on an image row. (The distance between two neighboring vertices is equal to the size of the quad's edges, which is more than a pixel at coarse resolutions) The same test is repeated along the image column. If either the vertical or the horizontal tests fails for any of the vertices of the triangle, the triangle is rejected since it is not part of a planar surface and the quad is subdivided. For these tests, we have found that 3D coordinates are more effective than disparity values. Since we do not require a manifold mesh and we are more interested in fast processing speeds, we do not maintain a restricted quad-tree (Pajarola, 2002a). Typical values for the coarsest and finest quad size are 32×32 pixels and 2×2 pixels respectively. Mesh generation is performed on the CPU and takes approximately $30ms$ per fused depth map, or under $3ms$ per frame.

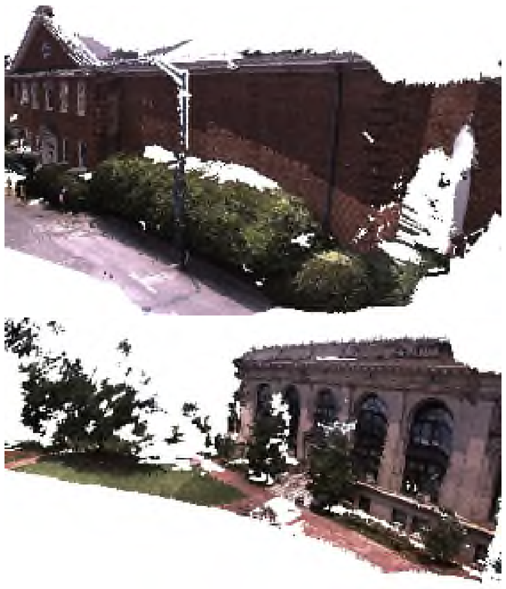


Figure 6: Screenshots of reconstructed buildings

8.1 Model Clean-up

The model generation module also suppresses the generation of duplicate representations of the same surface. This is necessary since surfaces are typically visible in more than one fused depth map. Considering that the distance between the camera and the scene is initially unknown and varies throughout the video stream, these overlaps are hard to prevent a priori. In order to accomplish the removal of duplicate surface representations, a record of the previous fused depth map is kept in memory. When a new fused depth map becomes available, the previous fused depth map is rendered onto it and pixels that have been explained by the model that has already been written to disk are masked out. The 3D vertices that correspond to the masked out points are not used for triangle generation. An illustration of the duplicate surface removal process can be seen in Figure 5. The model shown in Figure 5(a) was reconstructed using the side and upward camera. Since the cameras are synchronized an the relative transformations from one coordinate system to the other are known, we can register the partial reconstructions in the same coordinate system. Figure 5(b) shows the overlapping meshes that are generated for six fused depth maps from the two video streams. Our clean-up scheme is able to remove most of the duplicate representations and produce a simpler mesh shown in Figure 5(c). Note that globally the generated mesh is still not a manifold.

Despite its simplicity, this scheme is effective when the camera maintains a dominant direction of motion. It does not handle, however, the case of a part of the scene being revisited in a later pass, since the entire reconstructed model cannot be kept in memory. A more efficient representation for parts of the model potentially in the form of a bounding box is among our future research directions.

9 RESULTS

Our processing pipeline can achieve very high frame rates by leveraging both the CPU and GPU of a high-end personal computer. An implementation of our software targeted towards speed can generate 3D models at rates almost equal to those of video collection. This can be achieved with the following settings: GPS/INS-only pose estimation, 7 images with a resolution of 512×384 as input to the multiple-view stereo module which

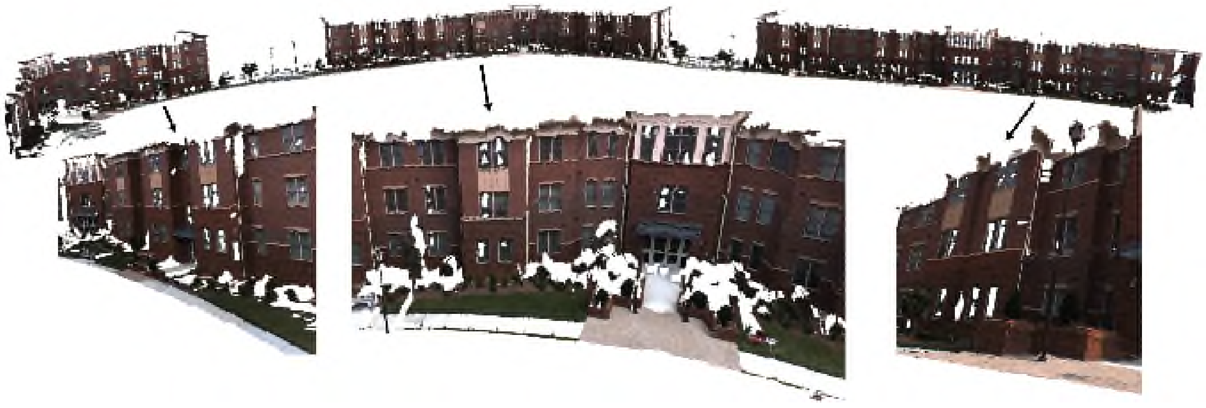


Figure 7: A two-camera reconstruction comprising more than 2,600 frames. Close-ups show that features such as the ground plane which is viewed at an angle from the camera and thin objects such as the light pole on the right are reconstructed well.

evaluates 48 plane positions and 11 depth maps as input to the fusion stage. This configuration achieves a processing rate of $23Hz$ on a PC with a dual-core AMD Opteron processor at 2.4GHz with an NVidia GeForce 8800 series GPU. Since frames are not used if the baseline with the previous frame is not large enough according to Section 5, the effective frame rate we achieve can be as high as $50Hz$ for typical driving speeds on streets with other vehicles and intersections. Screenshots of 3D models generated by our system can be seen in Figures 6 and 7.

10 QUANTITATIVE EVALUATION

To evaluate the quality of our 3D models, we reconstructed a 3D model of a building for which an accurately surveyed model has been made available to us. The surveyed model is of a Firestone building which is $80 \times 40m$ and has been surveyed at a $6mm$ accuracy. The input to our reconstruction software is 3,000 frames of video of the exterior of the Firestone store which were captured using two cameras (side and upward) and GPS/INS measurements. One of the cameras was pointed horizontally towards the middle and bottom of the building and the other was pointed up 30° towards the building. The videos from each camera were processed separately and the reconstructed model was directly compared with the surveyed model (see Figure 8). Our evaluation software computes two metrics for the reconstructed model: accuracy and completeness.

The surveyed model had to be pre-processed to remove inconsistencies with the video, since the survey and the video collection were conducted on different days. There are several objects such as parked cars that are visible in the video, but were not surveyed. Several of the doors of the building were left open causing some of the interior of the building to be reconstructed. Since reconstructions of these objects were not available in the ground truth they were not considered in the evaluation. In addition, the ground which is accurately reconstructed is not included in the ground truth model. We added a ground plane to the ground truth model and exclude from the evaluation points that are consistent with it. For all of the remaining parts, the distance from each point to the nearest point on the ground truth model was calculated to measure the *accuracy* of the reconstruction. A visualization showing these distances is provided in Figure 8, in which blue, green and red denote errors of $0cm$, $30cm$ and $60cm$ or above, respectively. The median error was 2.60cm, the average error was $6.60cm$ and 83% of the points were reconstructed within 5cm of the ground truth model. We also evaluated the *completeness* of our model by reversing the roles of the model and the ground truth. In this case, we measure the distance from each point in the ground truth model to the nearest point in the reconstructed model. Points in the ground truth that have been reconstructed poorly or have not been reconstructed at all, result in large values for this distance. Figure 8 also shows a visualization of the completeness evaluation using the same color scheme as the accuracy evaluation. The completeness of the model shown in the figure was 73% using a threshold of $50cm$. Note that some parts of the building remain invisible during the entire sequence.

11 CONCLUSIONS

We have described a system aimed at real-time, detailed, geo-registered, 3D reconstruction of urban environments from video captured by a multi-camera system and GPS/INS measurements. The quality of the reconstructions, with respect to both accuracy and completeness, is very satisfactory, especially considering the very high processing speeds we achieve. The system can scale to very large environments since it does not require markers or manual measurements and is capable of capturing and processing large amounts of data in real-time by utilizing both the CPU and GPU.

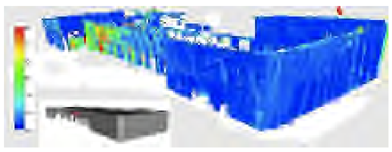
The models can be viewed from arbitrary viewpoints thus providing more effective visualization than mosaics or panoramas. One can virtually walk around the model at ground-level, fly over it in arbitrary directions, zoom in or out. Since the models are metric, they enable the user to make accurate measurements by selecting points or surfaces on them. Parts of a model can be removed or other models can be inserted to augment the existing model. The additions can be either reconstructed models from a different location or synthetic. Effective visualizations of potential modifications in an area can be generated this way. Finally, our system can be used for archiving and change detection.

Future challenges include a scheme for better utilization of multiple video streams in order to reconstruct each surface under the optimal viewing conditions while reducing the redundancy of the overall model. Along a different axis, we intend to investigate ways of decreasing the cost and size of our system by further exploring the implementation that uses a low-end INS and GPS or even only GPS instead of the Applanix system and by compressing the videos on the data acquisition platform.

ACKNOWLEDGEMENTS

This work has been supported in part by the DARPA UrbanScape program and the DTO VACE program.

(a) Reconstructed Model


(b) Accuracy evaluation. White indicates unsurveyed areas. (Inset) Surveyed Model

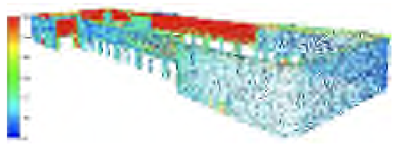

(c) Completeness of the Firestone building. Red areas mostly correspond to unobserved or untextured areas.

Figure 8: Firestone Building Accuracy and Completeness Evaluation. (b,c) Blue, green and red indicate errors of $0cm$, $30cm$ and 60+cm, respectively. Please view on a color display.

References

American Society of Photogrammetry, 2004. Manual of Photogrammetry (5th edition). Asprs Pubns.

Biber, P., Fleck, S., Staneker, D., Wand, M. and Strasser, W., 2005. First experiences with a mobile platform for flexible 3d model acquisition in indoor and outdoor environments – the waegele. In: ISPRS Working Group V/4: 3D-ARCH.

Collins, R., 1996. A space-sweep approach to true multi-image matching. In: Int. Conf. on Computer Vision and Pattern Recognition, pp. 358–363.

Cornelis, N., Cornelis, K. and Van Gool, L., 2006. Fast compact city modeling for navigation pre-visualization. In: Int. Conf. on Computer Vision and Pattern Recognition.

Debevec, P., Taylor, C. and Malik, J., 1996. Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach. In: SIGGRAPH, pp. 11–20.

El-Hakim, S., Beraldin, J.-A., Picard, M. and Vettore, A., 2003. Effective 3d modeling of heritage sites. In: 4th International Conference of 3D Imaging and Modeling, pp. 302–309.

Faugeras, O., 1993. Three-Dimensional Computer Vision: A Geometric Viewpoint. MIT Press.

Früh, C. and Zakhor, A., 2004. An automated method for large-scale, ground-based city model acquisition. Int. J. of Computer Vision 60(1), pp. 5–24.

Gallup, D., Frahm, J.-M., Mordohai, P., Yang, Q. and Pollefeys, M., 2007. Real-time plane-sweeping stereo with multiple sweeping directions. In: Int. Conf. on Computer Vision and Pattern Recognition.

Hartley, R. and Zisserman, A., 2000. Multiple View Geometry in Computer Vision. Cambridge University Press.

Kang, S., Szeliski, R. and Chai, J., 2001. Handling occlusions in dense multi-view stereo. In: Int. Conf. on Computer Vision and Pattern Recognition, pp. 103–110.

Nistér, D., 2001. Automatic dense reconstruction from uncalibrated video sequences. PhD Thesis, Royal Institute of Technology KTH, Stockholm, Sweden.

Nistér, D., Naroditsky, O. and Bergen, J., 2006. Visual odometry for ground vehicle applications. Journal of Field Robotics.

Pajarola, R., 2002a. Overview of quadtree-based terrain triangulation and visualization. Technical Report UCI-ICS-02-01, Information & Computer Science, University of California Irvine.

Pajarola, R., 2002b. Overview of quadtree-based terrain triangulation and visualization. Technical Report UCI-ICS-02-01, Information & Computer Science, University of California Irvine.

Pollefeys, M., Gool, L. V., Vergauwen, M., Cornelis, K., Verbiest, F. and Tops, J., 2003. Image-based 3d recording for archaeological field work. Computer Graphics and Applications 23(3), pp. 20–27.

Pollefeys, M., Koch, R. and Van Gool, L., 1999. Self-calibration and metric reconstruction inspite of varying and unknown intrinsic camera parameters. Int. J. of Computer Vision 32(1), pp. 7–25.

Pollefeys, M., Van Gool, L., Vergauwen, M., Verbiest, F., Cornelis, K., Tops, J. and Koch, R., 2004. Visual modeling with a hand-held camera. Int. J. of Computer Vision 59(3), pp. 207–232.

Román, A., Garg, G. and Levoy, M., 2004. Interactive design of multi-perspective images for visualizing urban landscapes. In: IEEE Visualization, pp. 537–544.

Rother, C. and Carlsson, S., 2002. Linear multi view reconstruction and camera recovery using a reference plane. Int. J. of Computer Vision 49(2-3), pp. 117–141.

Schindler, G. and Dellaert, F., 2004. Atlanta world: an expectation maximization framework for simultaneous low-level edge grouping and camera calibration in complex man-made environments. In: Int. Conf. on Computer Vision and Pattern Recognition, pp. 203–209.

Stamos, I. and Allen, P., 2002. Geometry and texture recovery of scenes of large scale. Computer Vision and Image Understanding 88(2), pp. 94–118.

Szeliski, R. and Scharstein, D., 2004. Sampling the disparity space image. IEEE Trans. Pattern Anal. Mach. Intell. 26(3), pp. 419–425.

Teller, S., 1998. Automated urban model acquisition: Project rationale and status. In: Image Understanding Workshop, pp. 455–462.

Teller, S., Antone, M., Bodnar, Z., Bosse, M., Coorg, S., Jethwa, M. and Master, N., 2003. Calibrated, registered images of an extended urban area. Int. Journ. of Computer Vision 53(1), pp. 93–107.

Werner, T. and Zisserman, A., 2002. New techniques for automated architectural reconstruction from photographs. In: European Conf. on Computer Vision, pp. 541–555.

Yang, R. and Pollefeys, M., 2005. A versatile stereo implementation on commodity graphics hardware. Journal of Real-Time Imaging 11(1), pp. 7–18.

Zhang, Z., 2000. A flexible new technique for camera calibration. IEEE Trans. on Pattern Analysis and Machine Intelligence 22(11), pp. 1330–1334.