Constructing 3D City Models by Merging Aerial and Ground Views

Christian Früh and Avideh Zakhor
University of California, Berkeley

The authors present an approach to automatically creating textured 3D city models using laser scans and images taken from ground level and a bird's-eye perspective.

Three-dimensional models consisting of the geometry and texture of urban surfaces could aid applications such as urban planning, training, disaster simulation, and virtual-heritage conservation. A standard technique used to create large-scale city models automatically or semi-automatically is to apply stereo vision on aerial or satellite imagery.[1] In recent years, advances in resolution and accuracy have also made airborne laser scanners suitable for generating digital surface models (DSM) and 3D models.[2] Although you can perform edge detection more accurately with aerial photos, airborne laser scans require no camera-parameter estimation and feature detection to obtain 3D geometry. Previous work has attempted to reconstruct polygonal models by using a library of predefined building shapes or by combining the DSM with digital ground plans or aerial images. While you can achieve submeter resolution with this technique, you can only capture the building roofs and not their facades.

Several research projects have attempted to create models from ground-based views at a high level of detail to enable virtual exploration of city environments. While most of these approaches result in visually pleasing models, they involve an enormous amount of manual work, such as importing the geometry obtained from construction plans or selecting primitive shapes and correspondence points for image-based modeling or complex data acquisition. Other attempts to acquire close-range data in an automated fashion have used image-based[3] or 3D laser scanner approaches.[4,5] However, these approaches don't scale to more than a few buildings because they acquire data in a slow, stop-and-go fashion.

In previous work, we proposed an automated method that would rapidly acquire 3D geometry and texture data for an entire city.[6,7] This method uses a vehicle equipped with 2D laser scanners and a digital camera to acquire data while driving at normal speeds on public roads. Other researchers proposed a similar system using 2D laser scanners and line cameras.[8] In both systems, the researchers acquire data continuously and quickly. In another article, we presented automated methods to process this type of data efficiently to obtain a detailed model of the building facades in downtown Berkeley.[9] However, these facade models don't provide information about roofs or terrain shape because they consist only of surfaces visible from the ground level.

In this article, we describe an approach to register and merge our detailed facade models with a complementary airborne model. Figure 1 shows the data-flow diagram of our method. The airborne modeling process on the left in Figure 1 provides a half-meter resolution model with a bird's-eye view of the entire area, containing terrain profile and building tops. The ground-based modeling process on the right results in a detailed model of the building facades. Using the DSM obtained from airborne laser scans, we localize the acquisition vehicle and register the ground-based facades to the airborne model by means of Monte Carlo localization (MCL). We merge the two models with different resolutions to obtain a 3D model.

Textured surface mesh from airborne laser scans

We use the DSM for localizing the ground-based data-acquisition vehicle and for adding roofs and terrain to the ground-based facade models. In contrast to previous approaches, we don't explicitly extract geometric primitives from the DSM. While we use aerial laser scans to create the DSM, it's equally feasible to use a DSM obtained from other sources, such as stereo vision.

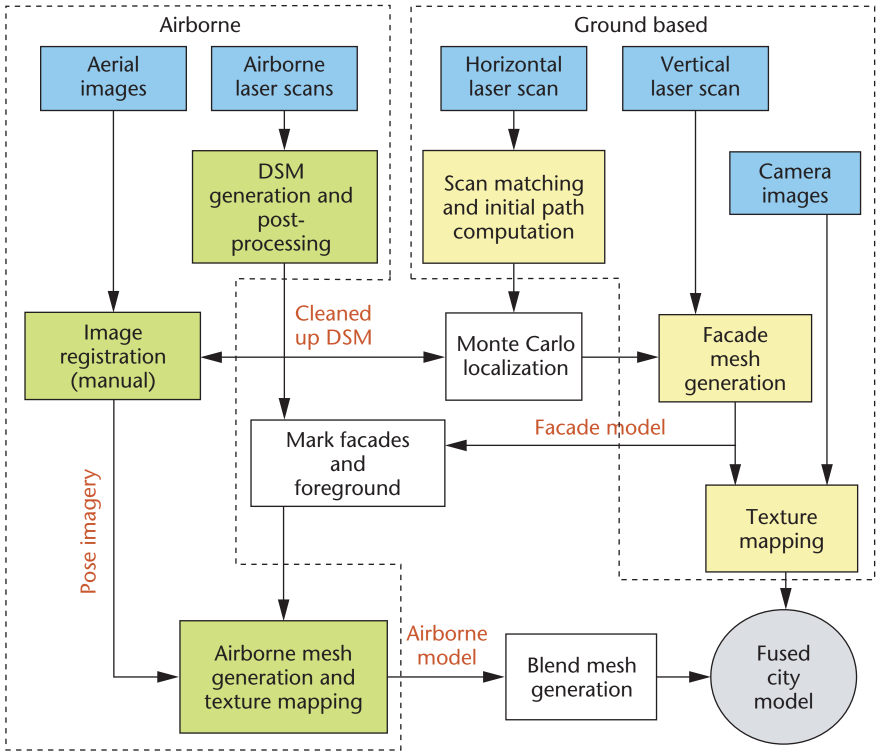
Scan point resampling and DSM generation

During airborne laser scan acquisition using a 2D scanner mounted on an airplane, the unpredictable roll and tilt motion of the plane destroyed the inherent row-column order of the scans. Thus, the scans might be interpreted as an unstructured set of 3D vertices in space, with the *x*, *y* coordinates specifying the geographical location

Published by the IEEE Computer Society

and the *z* coordinate the altitude. To process the scans efficiently, it's helpful to resample the scan points to a row-column structure even though this step could reduce the spatial resolution. To transfer the scans into a DSM—a regular array of altitude values—we define a row-column grid in the ground plane and sort the scan points into the grid cells. The density of scan points is not uniform, which means there are grid cells with no scan point and others with multiple scan points. Because the percentage of cells without scan points and the resolution of the DSM depend on grid cell size, we must compromise by leaving few cells without a sample while maintaining the resolution at an acceptable level.

In our case, the scans have an accuracy of 30 centimeters in the horizontal and vertical directions and a raw spot spacing of 0.5 meters or less. Measuring both the first and the last pulses of the returning laser light, we select a square cell size of 0.5×0.5 meters, resulting in about half the cells being occupied. We create the DSM by assigning to each cell the highest *z* value among its member points to preserve overhanging rooftops while suppressing the wall points. We fill the empty cells using nearest-neighbor interpolation to preserve sharp edges. We can interpret each grid cell as a vertex, where the *x*, *y* location is the cell center and the *z* coordinate is the altitude value—or as a pixel at *x*, *y* with a gray intensity proportional to *z*.

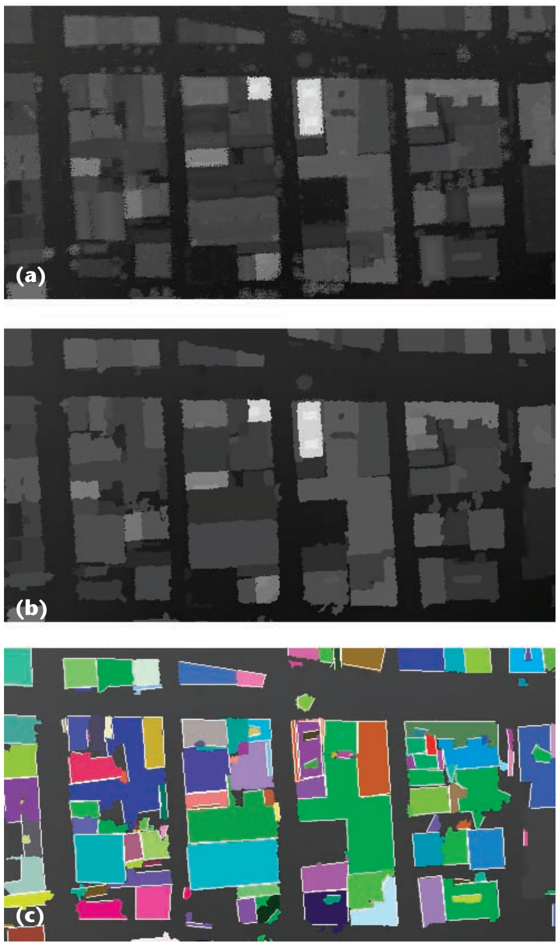


1 **Dataflow diagram of our 3D city-modeling approach. Airborne modeling steps are highlighted in green, ground-based modeling steps in yellow, and model-fusion steps in white.**

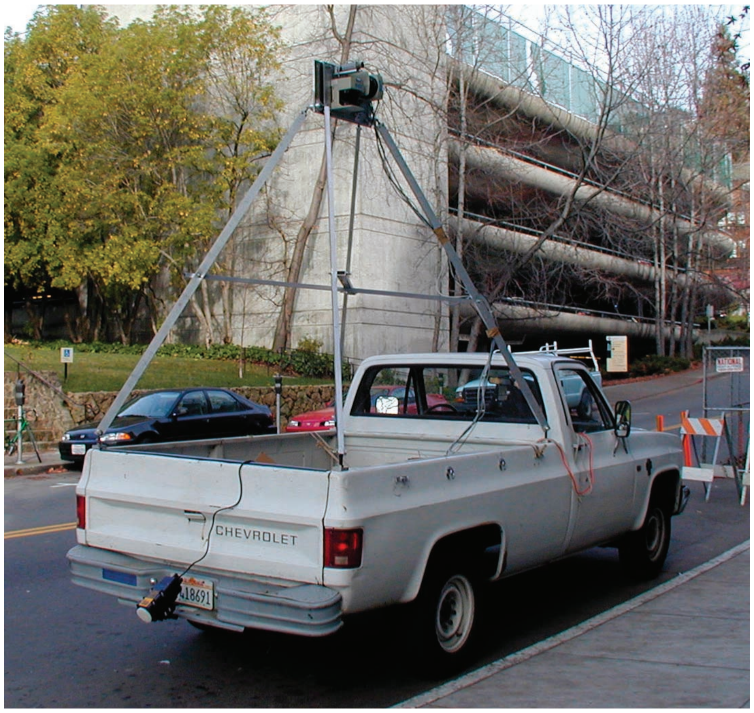
Processing the DSM

The DSM contains the plain rooftops and terrain shape as well as objects such as cars and trees. Roofs, in particular, look bumpy because of a large number of smaller objects on them, such as ventilation ducts, antennas, and railings, which are impossible to reconstruct properly at the DSM's resolution. Furthermore, scan points below overhanging roofs cause ambiguous altitude values, resulting in jittery edges. To obtain more visually pleasing roof reconstructions, we apply several processing steps.

We first target flattening the bumpy rooftops. We apply a segmentation algorithm to all nonground pixels on the basis of the depth discontinuity between adjacent pixels. We replace small, isolated regions with ground-level altitude to remove objects such as cars or trees in the DSM. We subdivide the larger regions further into planar subregions by means of planar segmentation. Then we unite small regions and subregions with larger neighbors by setting their *z* values to the larger region's corresponding plane. This procedure helps remove undesired small objects from the roofs and prevents the separation of rooftops into too many cluttered regions. Figure 2a shows the original DSM and Figure 2b the resulting processed DSM.



2 **Processing steps for DSM: (a) DSM obtained from scan-point resampling; (b) DSM after flattening roofs; and (c) segments with Ransac lines in white.**



3 Acquisition vehicle.

The second processing step straightens the jittery edges. We resegment the DSM into regions, detect the boundary points of each region, and use Ransac[10] to find line segments that approximate the regions. For the consensus computation, we also consider boundary points of surrounding regions to detect even short linear sides of regions and to align them consistently with surrounding buildings. Furthermore, we allocate bonus consensus if a detected line is parallel or perpendicular to a region's most dominant line. For each region, we obtain a set of boundary line segments that represent the most important edges. For all other boundary parts where we have not found a proper line approximation, we don't change the original DSM. Figure 2c shows the regions resulting from processing Figure 2b superimposed with the corresponding Ransac lines. Compared with Figure 2b, most edges look straightened out.

Textured mesh generation

Airborne models are commonly generated from scans by detecting features such as planar surfaces in the DSM or matching a predefined set of possible rooftop and building shapes.[2] In other words, these models decompose the buildings found in the DSM into polygonal 3D primitives. While the advantage of these model-based approaches is their robust reconstruction of geometry in spite of erroneous scan points and low sample density, they are highly dependent on shape assumptions. In particular, the results are poor if many unconventional buildings are present or if buildings are surrounded by trees—conditions that exist on the UC Berkeley campus. Although the resulting models might appear clean and precise, the reconstructed buildings' geometry is not necessarily correct if the underlying shape assumptions are invalid.

Our application makes an accurate model of the building facades readily available from the ground-based acquisition. As such, we are primarily interested in adding the complementary roof and terrain geometry, so we apply a different strategy to create a model from the airborne view. We transform the cleaned-up DSM directly into a triangular mesh and reduce the number of triangles by simplification. The advantage of this method is that we can control the mesh generation process on a per-pixel level. We exploit this property in the model fusion procedure. Additionally, this method has a low processing complexity and is robust. Because this approach requires no predefined building shapes, we can apply it to buildings with unknown shapes—even in presence of trees. Admittedly, doing so typically comes at the expense of a larger number of triangles.

Because the DSM has a regular topology, we can directly transform it into a structured mesh by connecting each vertex with its neighboring ones. The DSM for a city is large, and the resulting mesh has two triangles per cell, yielding 8 million triangles per square kilometer for the 0.5×0.5 meter grid size. Because many vertices are coplanar or have low curvature, we can drastically reduce the number of triangles without significant loss of quality. We use the Qslim mesh simplification algorithm[11] to reduce the number of triangles. Empirically, we have found it possible to reduce the initial surface mesh to about 100,000 triangles per square kilometer at the highest level of detail without noticeable loss in quality.

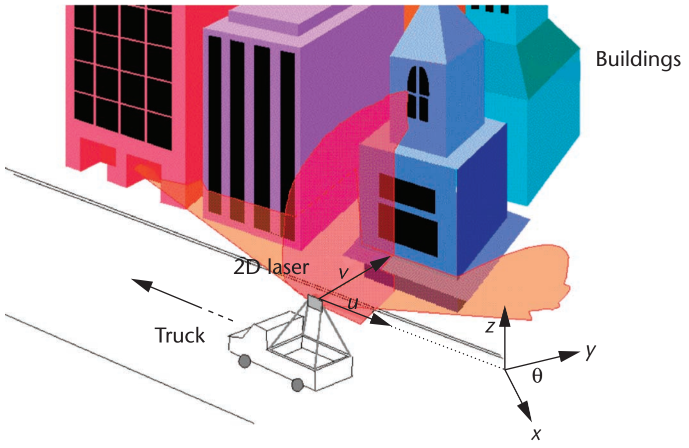
Using aerial images taken with an uncalibrated camera from an unknown pose, we texture map the reduced mesh semi-automatically. We manually select a few correspondence points in both the aerial photo and the DSM, taking a few minutes per image. Then, we compute both internal and external camera parameters and texture map the mesh. A location in the DSM corresponds to a 3D vertex in space, letting us project it into an aerial image if we know the camera parameters. We use an adaptation of Lowe's algorithm to minimize the difference between selected correspondence points and computed projections. After we determine the camera parameters for each geometry triangle, we identify the corresponding texture triangle in an image by projecting the corner vertices. Then, for each mesh triangle, we select the best image for texture mapping by taking into account resolution, normal vector orientation, and occlusions.

Ground-based modeling and model registration

In previous research, we developed a mobile ground-based data acquisition system consisting of two Sick LMS 2D laser scanners and a digital color camera with a wide-angle lens.[6] We perform the data acquisition in a fast drive-by rather than in a stop-and-go fashion. Doing so enables short acquisition times limited only by traffic conditions. As shown in Figure 3, we mount our acquisition system on a rack on top of a truck, letting us obtain measurements not obstructed by objects such as pedestrians and cars.

Both 2D scanners face the same side of the street; one is mounted horizontally, the other vertically, as shown in Figure 4. We mount the camera toward the scanners, with its line of sight parallel to the intersection between the orthogonal scanning planes. Hardware signals synchronize laser scanners and the camera. In our measurement setup, we use the vertical scanner to scan building facade geometry as the vehicle moves, and hence it's crucial to accurately determine the vehicle location for each vertical scan.

We developed algorithms to estimate relative vehicle position changes according to the horizontal scans, and we estimate the driven path as a concatenation of relative position changes. Because errors in the estimates accumulate, we apply a global correction. Rather than using a GPS sensor, which is not sufficiently reliable in "urban canyons," we use an aerial photo as a 2D global reference in conjunction with MCL. We extend the application of MCL to a global map derived from the DSM to determine the vehicle's pose in nonplanar terrain and to register the ground-based facade models with respect to the DSM.

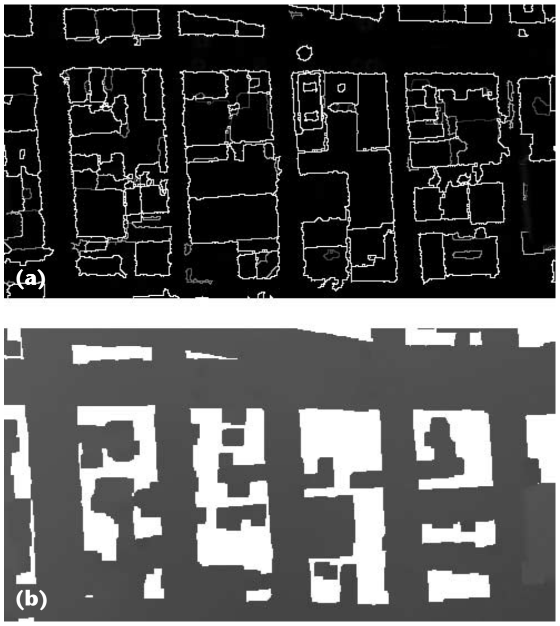


4 **Ground-based acquisition setup.**

Edge map and DTM

The MCL approach requires us to create two additional maps: an edge map that contains the location of height discontinuities and a digital terrain model (DTM) that contains terrain altitude. We apply a Sobel edge detector to a gray-scale aerial image to find edges in the city for localizing the ground-based data acquisition vehicle. For the DSM, rather than using the Sobel edge detector, we define a discontinuity-detection filter that marks a pixel if at least one of its eight adjacent pixels is more than a preset threshold below it. We are dealing with 3D height maps rather than 2D images. Hence, we only mark the outermost pixels of the taller objects, such as building tops, creating a sharper edge map than a Sobel filter. In fact, the resulting map is a global occupancy grid for walls. While for aerial photos, shadows of buildings or trees and perspective shift of building tops cause numerous false edges in the image, neither problem exists for the edge map from airborne laser scans.

The DSM contains building facade locations as height discontinuities and the street altitudes for streets on which the vehicle is driven. As such, we can assign this altitude to a vehicle's z coordinate. It's not possible to use the z value of a DSM location directly because the Lidar captures cars and overhanging trees during airborne data acquisition, resulting in z values up to several meters above the actual street level for some locations. For a particular DSM location, we estimate street level altitude by averaging the z coordinates of available ground pixels within a surrounding window, weighing them with an exponential function decreasing with distance. The result is a smooth, dense DTM as an estimate of the ground level near roads. Figures 5a and 5b show edge maps and DTMs computed from the DSM shown in Figure 2b.

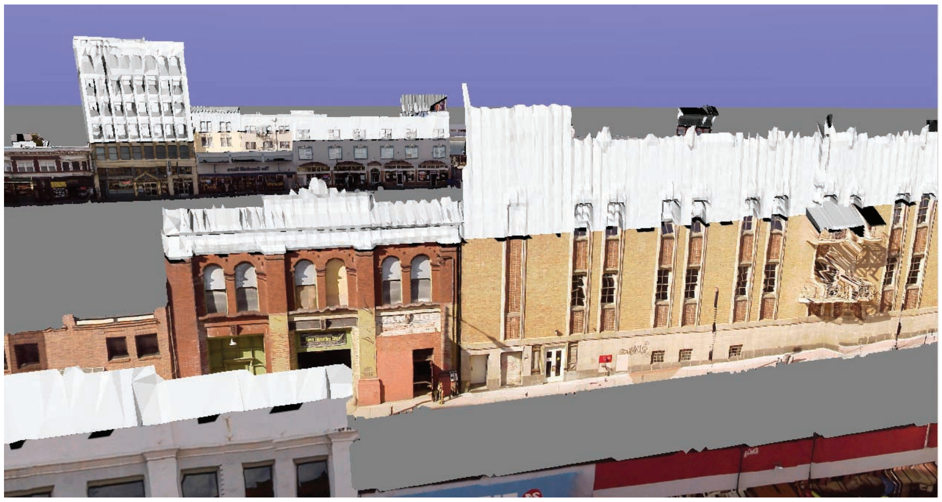


5 **Map generation for MCL: (a) edge map and (b) DTM. For the white pixels, there is no ground-level estimate available.**

Model registration with MCL

MCL is a particle-filtering-based implementation of the probabilistic Markov localization, and was introduced by another research team for tracking the position of a vehicle in mobile robotics.[12] Given a series of relative motion estimates and corresponding horizontal laser scans, the MCL-based approach we have proposed elsewhere can determine the accurate position within a city. The principle of the correction is to adjust initial vehicle motion estimates so that scan points from the ground-based data acquisition match the edges in the global edge map. The scan-to-scan matching can only estimate a 3-DOF relative motion—that is, a 2D translation and rotation in the scanner's coordinate system. If the vehicle is on a slope, the motion estimates are given in a plane at an angle with respect to the global xy-plane, and the displacement should in fact be corrected with the cosine of the slope angle.

However, because this effect is small—0.5 percent for a 10-degree slope—we can safely neglect it and use the relative scan-to-scan matching estimates as if the truck's coordinate system were parallel to the global coordinate system. Using MCL with the relative estimates from scan matching and the edge map from the DSM, we can arrive at a series of global pose probability density func-



6 **Ground-based facade models.**

tions and correction vectors for x, y, and yaw motion. We can then apply these corrections to the initial path to obtain an accurate localization of the acquisition vehicle. Using the DTM, we obtain an estimate of two more DOF: For the first, the final $z^{(i)}$ coordinate of an intermediate pose P_i in the path is set to DTM level at ($x^{(i)}$, $y^{(i)}$) location. For the second, we compute the pitch angle representing the slope as

$$pitch^{(i)} = \arctan\left(\frac{z^{(i)} - z^{(i-1)}}{\sqrt{(x^{(i)} - x^{(i-1)})^2 + (y^{(i)} - y^{(i-1)})^2}}\right)$$

That is, we compute the pitch by using the height difference and the traveled distance between successive positions. Because the resolution of the DSM is only 1 meter and the ground level is obtained via a smoothing process, the estimated pitch contains only the low-frequency components and not highly dynamic pitch changes, such as those caused by pavement holes and bumps. Nevertheless, the obtained pitch is an acceptable estimate because the size of the truck makes it relatively stable along its long axis.

We do not estimate the last missing DOF—the roll angle—using airborne data. Rather, we assume buildings are generally built vertically, and we apply a histogram analysis on the angles between successive vertical scan points. If the average distribution peak is not centered at 90 degrees, we set the roll angle estimate to the difference between histogram peak and 90 degree.

At the end of these steps, we obtain 6-DOF estimates for the global pose and apply a framework of automated processing algorithms to remove foreground and reconstruct facade models. We divide the path into easy-to-handle segments that we process individually. Additional steps include generating a point cloud, classifying areas such as facade and foreground, removing foreground geometry, filling facade holes and windows, creating a surface mesh, and mapping the textures. As a result, we obtain texture-mapped facade models like the one shown in Figure 6. We can't texture map the upper parts of tall buildings that fall outside the camera's field of view during data acquisition.

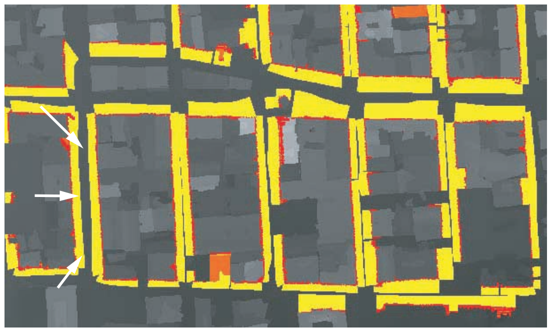
A path segment texture is typically several tens of megabytes, which exceeds the rendering capabilities of today's graphics cards. Therefore, we optimize the facade models for rendering by generating multiple levels-of-detail (LOD) so that only a small portion of the entire model is rendered at the highest LOD at any given time. We subdivide the facade meshes along vertical planes and generate lower LODs for each submesh using the Qslim simplification algorithm for geometry and bicubic interpolation for texture reduction. We combine all submeshes in a scene graph that controls the switching of the LODs according to the viewer's position. This helps us render the large amounts of geometry and texture with standard tools such as VRML players.

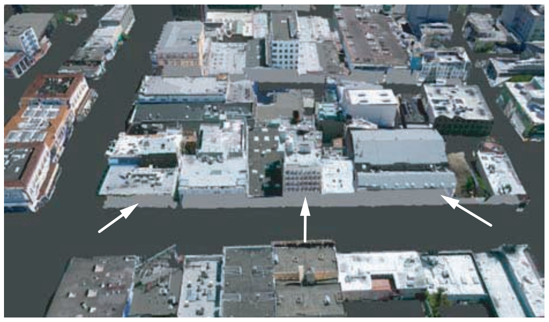
Model merging

We automatically generate both ground-based faced models and the aerial surface mesh from the DSM. Given the complexity of a city environment, it's inevitable that we capture some parts only partially or erroneously, which can result in substantial discrepancies between the two meshes. Our goal is a photorealistic virtual exploration of the city, so creating models with visually pleasing appearances is more important than CAD properties such as water tightness. Common approaches for fusing meshes, such as sweeping and intersecting contained volume[5] or mesh zippering,[13] require a substantial overlap between the two meshes. This is not the case in our application because the two views are complementary. Additionally, the two meshes have entirely different resolutions: the resolution of the facade models—at about 10 to 15 cm—is substantially higher than that of the airborne surface mesh. Furthermore, enabling interactive rendering requires that the two models fit together even when their parts have different levels of detail.

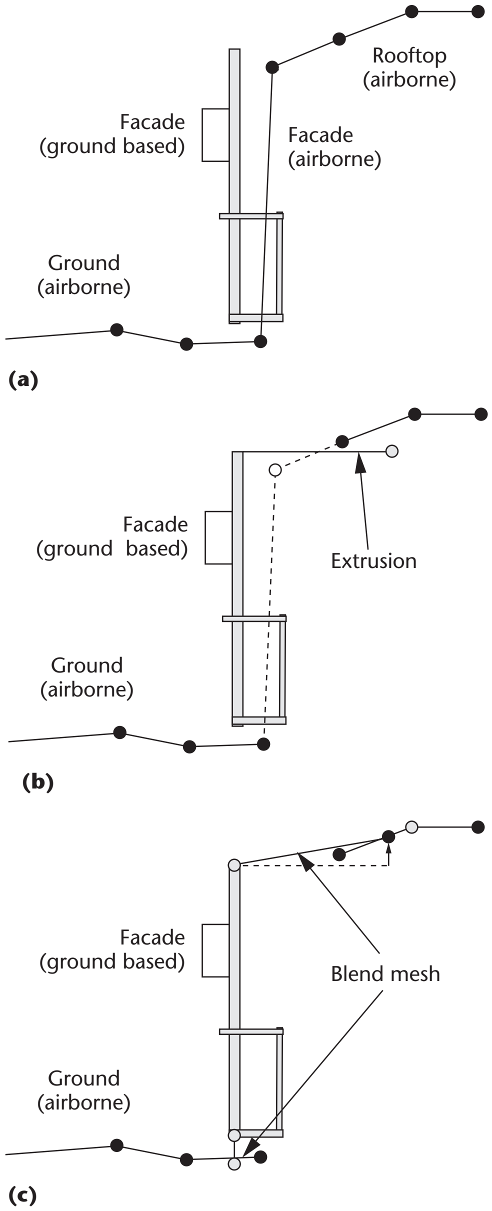
Due to its higher resolution, it's reasonable to give preference to the ground-based facades wherever available and to use the airborne mesh only for roofs and terrain shape. Rather than replacing triangles in the airborne mesh for which ground-based geometry is available, we consider the redundancy before the mesh generation step in the DSM. For all vertices of the ground-based facade models, we mark the corresponding cells in the DSM. This is possible because ground-based models and DSM have been registered through the localization techniques described earlier.

We further identify and mark those areas that our automated facade processing has classified as foreground details, such as trees and cars. These marks control the subsequent airborne mesh generation from DSM. Specifically, during the generation of the airborne mesh, we replace the z value for the foreground with the ground-level estimate from the DTM, and we don't


(a)


(b)

7 **Removing facades from the airborne model: (a) DSM with facade areas marked in red and foreground in yellow and (b) resulting mesh with corresponding facades and foreground objects removed. The arrows in (a) and (b) mark corresponding locations in the DSM and mesh.**



8 **Creation of a blend mesh based on a vertical cut through a building facade. (a) Initial registered airborne and ground-based model; (b) facade of airborne model replaced and ground-based model extruded; and (c) blending the two meshes by adjusting loose ends of extrusions to airborne mesh surface and mapping texture.**

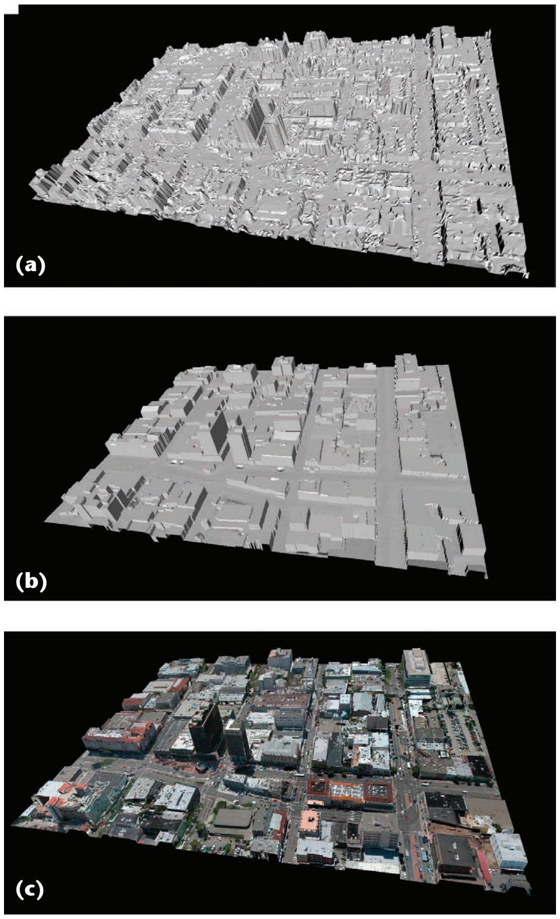
create triangles at ground-based facade positions. The first step is necessary to enforce consistency and remove those foreground objects in the airborne mesh that have already been deleted in the facade models. Figure 7a shows the DSM with marked facade areas and foreground; Figure 7b shows the resulting airborne surface mesh with the corresponding facade triangles removed and the foreground areas leveled to DTM altitude.

The facade models that we will put in place of the removed ones don't perfectly match the airborne mesh because of their different resolutions and capture viewpoints. Generally, this procedure makes the removed geometry slightly larger than the actual ground-based facade to be placed in the corresponding location. To solve this discrepancy and to make mesh transitions less noticeable, we fill the gap with additional triangles to join the two meshes. Figure 8a shows the initial registered meshes. Our approach to creating such a blend mesh is to extrude the buildings along an axis perpendicular to the facades, as shown in Figure 8b. We then shift the loose end vertices location to connect to the closest airborne mesh surface, as shown in Figure 8c. This technique is similar to the way a plumb line can help close gaps between windows and roof tiles. We finally texture map these blend triangles with the texture from the aerial photo. As they attach at one end to the ground-based model and at the other end to the airborne model, blend triangles reduce visible seams at model transitions.

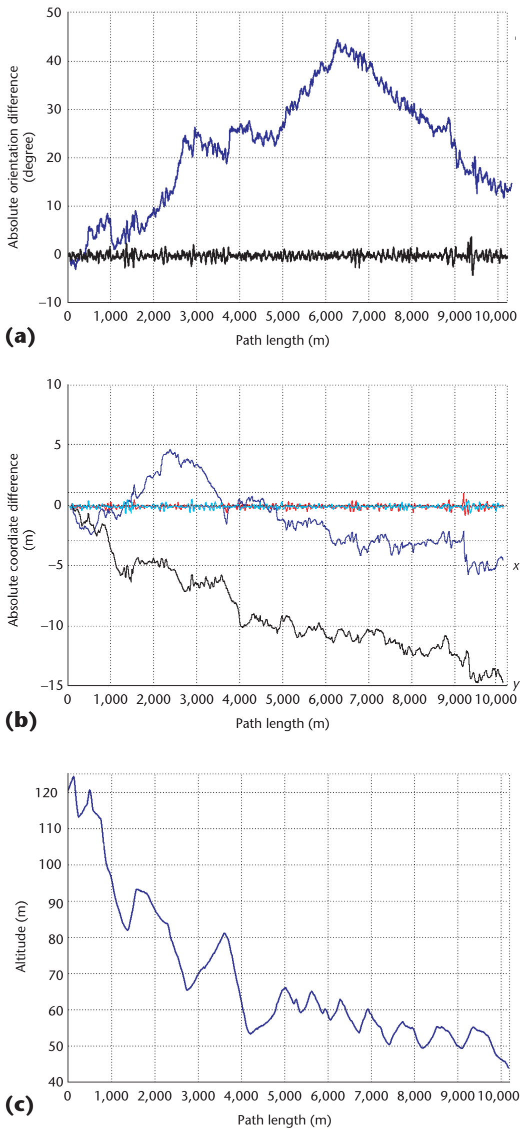
Results

We have applied our proposed algorithms on a data set covering downtown Berkeley. We acquired airborne laser scans through the company Airborne 1 Inc. We resampled the entire data set consisting of 60 million scan points to a 0.5 x 0.5 meter grid and applied the processing steps described previously to obtain a DSM, an edge map, and a DTM for the entire area. We selected feature points in 5-megapixel digital images taken from a helicopter. This registration process took about an hour for the 12 images we used for the downtown Berkeley area. Then, our system automatically triangulated the DTM, simplified the mesh, and texture-mapped the model.

Figure 9a (next page) shows the surface mesh obtained from directly triangulating the DSM. Figure 9b shows the triangulated DSM after the processing steps, and 9c shows the texture-mapped model. It's difficult to evaluate the accuracy of this airborne model because no ground truth with sufficient accuracy is



9 **Airborne model: (a) DSM directly triangulated, (b) triangulated after postprocessing, and (c) model texture mapped.**



10 **Global correction for path one: (a) yaw angle difference between initial path and global estimates before and after correction; (b) differences of *x* and *y* coordinates before and after correction; and (c) assigned *z* coordinates.**

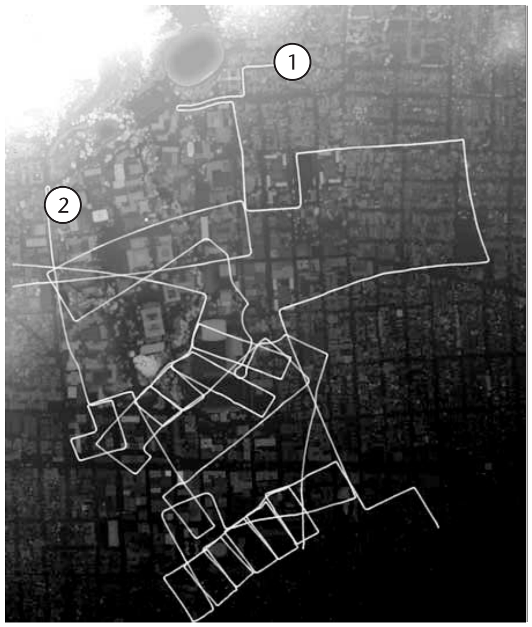
readily available—even at the city's planning department. However, we admittedly sacrificed accuracy for the sake of visual appearance of the texture-mapped model. Thus, our approach combines elements of model- and image-based rendering. While this is undesirable in some applications, we believe it's appropriate for interactive visualization applications.

We acquired the ground-based data during two measurement drives in Berkeley. The first drive took 37 minutes and was 10.2 kilometers long, starting from a location near the hills, going down Telegraph Avenue, and in loops around the central downtown blocks. The second drive was 41 minutes and 14.1 km, starting from Cory Hall at UC Berkeley and looping around the remaining downtown blocks. We captured a total of 332,575 vertical and horizontal scans consisting of 85 million scan points along with 19,200 images.

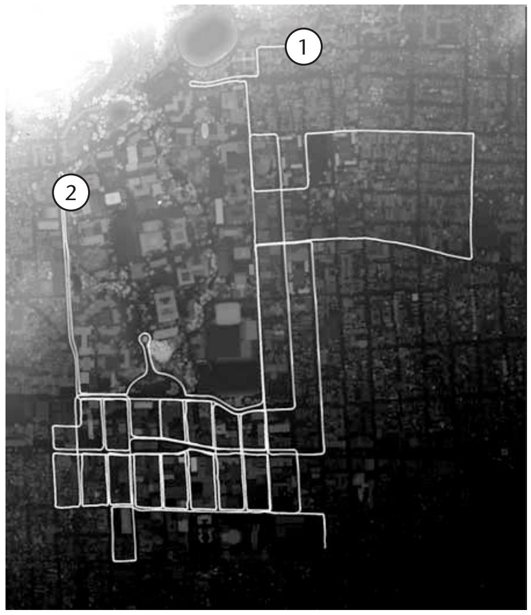
In previous MCL experiments based on edge maps from aerial images with 30-centimeter resolution, we found an enormous localization uncertainty at some locations because of false edges and perspective shifts. In the past, we had to use 120,000 particles during MCL to approximate the spread-out probability distribution appropriately and track the vehicle reliably. For the edge map derived from airborne laser scans, we found that despite its lower resolution, the vehicle could be tracked with as few as 5,000 particles.

As Figure 10 shows, we applied the global correction first to the yaw angles as shown in Figure 10a, then recomputed the path and applied the correction to the *x* and *y* coordinates, as shown in Figure 10b. As expected, the global correction substantially modifies the initial pose estimates, thus reducing errors in subsequent processing. Figure 10c plots the assigned *z* coordinate, clearly showing the slope from our starting position at a higher altitude near the Berkeley Hills down toward the San Francisco Bay, as well as the ups and downs on this slope while looping around the downtown blocks.

Figure 11a shows uncorrected paths one and two superimposed on the airborne DSM. Figure 11b shows the paths after global correction, and Figure 12 shows the ground based horizontal scan points for the corrected paths. As the figures demonstrate, the path and horizontal scan points match the DSM closely after applying the global corrections.

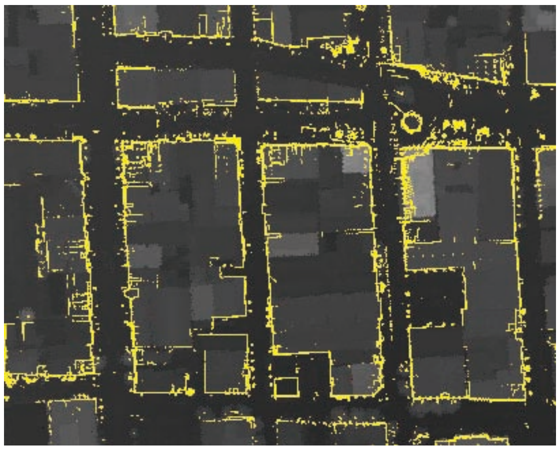


(a)

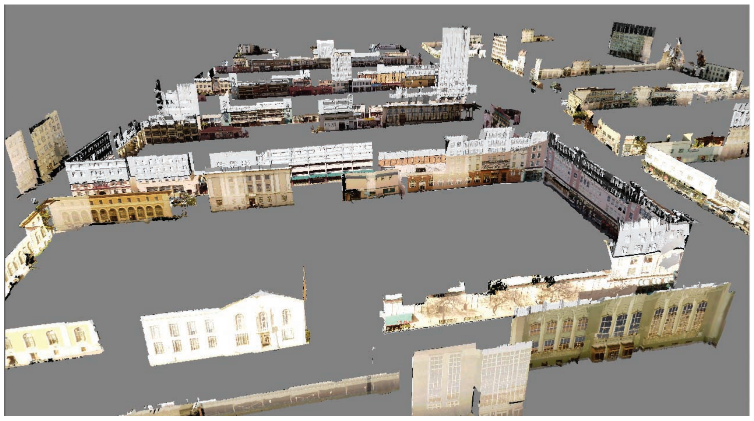


(b)

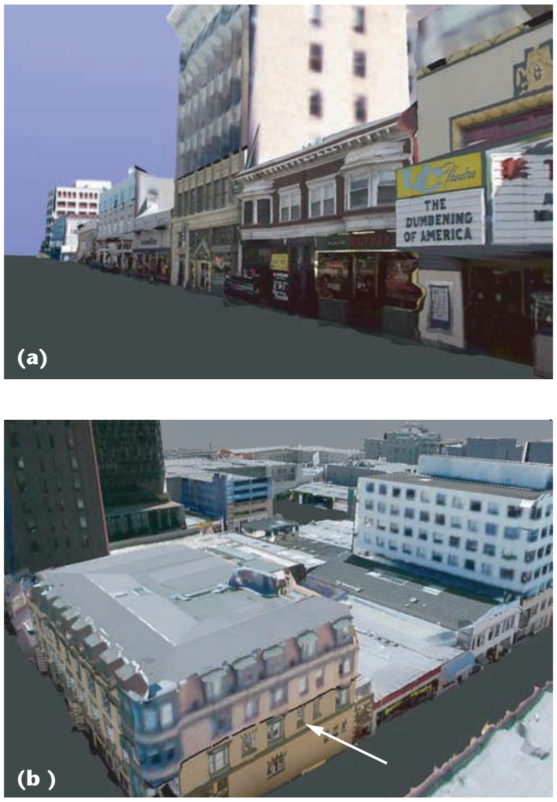
11 Driven paths superimposed on top of the DSM (a) before correction and (b) after correction. The circles denote the starting position for paths one and two.



12 Horizontal scan points for corrected paths.



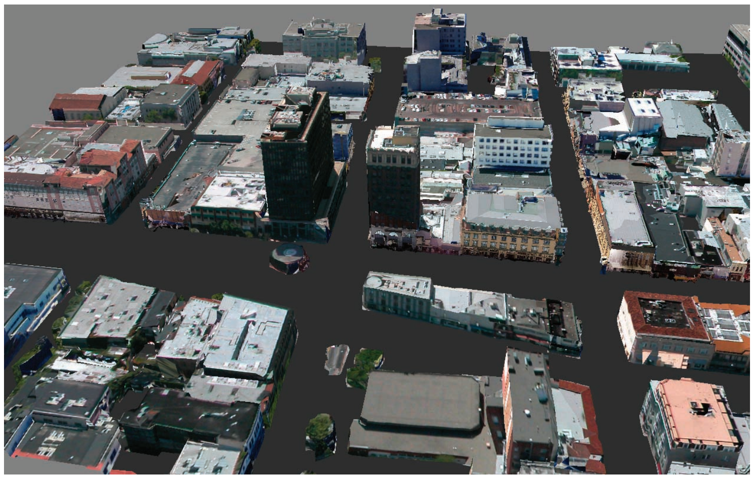
13 Facade model for the downtown Berkeley area.



14 Walk-through view of the model (a) as seen from the ground level and (b) as seen from the rooftop of a building.

Due to the MCL correction, all scans and images are georeferenced and we generate a facade model for the 12 street blocks of the downtown area using the processing steps described earlier. Figure 13 shows the resulting facades. The acquisition time for the 12 downtown Berkeley blocks was only 25 minutes. This is the time portion of both paths that it took to drive the total of 8 kilometers around these 12 blocks under city traffic conditions.

Because the DSM acts as the global reference for MCL, we registered the DSM and facade models with each other, meaning we can apply the model merging steps as described previously. Figure 14a shows the resulting com-



15 Bird's eye view of the model.

Table 1. Processing times for the downtown Berkeley blocks.

Process	Time (minutes)
Vehicle localization and registration with DSM	164
Facade model generation and optimization for rendering	121
DSM computation and projecting facade locations	8
Generating textured airborne mesh and blending	26
Total processing time	319

bined model for the looped downtown Berkeley blocks as viewed in a walk-through or drive-through, and Figure 14b shows a view from the rooftop of a downtown building. Because of the limited field of view of the ground-based camera, we texture mapped the upper parts of the building facades with aerial imagery. The noticeable difference in resolution between the upper and lower parts of the texture on the building in Figure 14b emphasizes the necessity of ground-based facade models for walk-through applications. Figure 15 shows the same model in a view from the top as it appears in a fly through. You can download the model for interactive visualization from www-video.eecs.berkeley.edu/~frueh/3d/.

Our approach to city modeling is automated as well as quick computationally. As Table 1 shows, the total time for the automated processing and model generation for the 12 downtown blocks is around five hours on a 2-GHz Pentium 4 PC. Because the complexity of all developed algorithms is linear in area and path length, our method is scalable to large environments.

Discussion and future work

While we have shown that our approach results in visually acceptable models for downtown environments, one of its limitations is the way it handles foreground objects, such as cars and trees. In particular, we currently remove such objects to avoid the problem of reconstructing and rendering them. If too many trees are present—as is often the case in residential areas—our underlying assumptions about dominant building planes are often not met and filling in facades is no longer reliable. Therefore, in residential areas, the resulting model contains some artifacts. We would like to include common city objects, such as cars, street lights, signs, and telephone lines because they substantially contribute to a high level of photo realism. Our future work will address reconstructing and adding 3D and 4D foreground components by using multiple scanners at different oblique directions.

Selecting correspondence points for registering the aerial imagery is the only manual step in our entire processing chain. We plan to automate this process as well, a goal we could achieve by using an accurate GPS unit or applying model-based vision methods, such as finding vanishing points or matching features in DSM and images. Finally, the high level of detail of our method results in enormous amounts of data. For some applications, a lower resolution could be sufficient. Furthermore, the large data size, in particular the amount of high-resolution texture, makes rendering a challenging task. Future work must address data management issues related to rendering models that are many orders of magnitude larger than the memory of existing computer systems. ■

Acknowledgment

This work was sponsored by Army Research Office contract DAAD19-00-1-0352.

References

1. Z. Kim, A. Huertas, and R. Nevatia, "Automatic Description of Buildings with Complex Rooftops from Multiple Images," *Proc. Computer Vision and Pattern Recognition*, IEEE CS Press, vol. 2, 2001, pp. II-272-279.
2. C. Brenner, N. Haala, and D. Fritsch: "Towards Fully Automated 3D City Model Generation," *Proc. Workshop Automatic Extraction of Man-Made Objects from Aerial and Space Images III*, Int'l Soc. Photogrammetry and Remote Sensing, 2001.
3. A. Dick et al., "Combining Single View Recognition and Multiple View Stereo for Architectural Scenes," *Proc. IEEE Int'l Conf. Computer Vision*, IEEE CS Press, vol. 1, 2001, pp. 268-274.
4. V. Sequeira and J.G.M. Goncalves, "3D Reality Modeling: Photo-Realistic 3D Models of Real World Scenes," *Proc. 1st Int'l Symp. 3D Data Processing Visualization and Transmission*, IEEE CS Press, 2002, pp. 776-783.
5. I. Stamos and P.K. Allen, "Geometry and Texture Recovery of Scenes of Large Scale," *Computer Vision and Image Understanding* (CVIU), vol. 88, no. 2, Nov. 2002, pp. 94-118.
6. C. Früh and A. Zakhor, "Fast 3D Model Generation in Urban Environments," *Proc. IEEE Conf. Multisensor Fusion and Integration for Intelligent Systems*, IEEE CS Press, 2001, pp. 165-170.

7. C. Früh and A. Zakhor, "3D Model Generation of Cities Using Aerial Photographs and Ground Level Laser Scans," *Proc. Computer Vision and Pattern Recognition*, IEEE CS Press, vol. 2.2, 2001, p. II-31-8.
8. H. Zhao and R. Shibasaki, "Reconstructing a Textured CAD Model of an Urban Environment Using Vehicle-Borne Laser Range Scanners and Line Cameras," *Machine Vision and Applications*, vol. 14, no. 1, 2003, pp. 35-41
9. C. Früh and A. Zakhor, "Data Processing Algorithms for Generating Textured 3D Building Façade Meshes From Laser Scans and Camera Images," *Proc. 3D Processing, Visualization and Transmission*, IEEE CS Press, 2002, pp. 834-847.
10. M.A. Fischler and R.C. Bolles, "Random Sample Consensus: A Paradigm for Model Fitting with Application to Image Analysis and Automated Cartography," *Communication Assoc. and Computing Machine*, vol. 24, no. 6, 1981, pp. 381-395.
11. M. Garland and P. Heckbert, "Surface Simplification Using Quadric Error Metrics," *Proc. Siggraph 97*, ACM Press, 1997, pp. 209-216.
12. S. Thrun et al., "Robust Monte Carlo Localization for Mobile Robots," *Artificial Intelligence*, vol. 128, nos. 1-2, 2001, pp.99-141.
13. G. Turk and M. Levoy, "Zippered Polygon Meshes from Range Images," *Proc. Siggraph 94*, ACM, 1994, pp. 311-318.

Christian Früh *is a postdoctoral researcher at the Video and Image Processing Lab at the University of California, Berkeley. His research interests include mobile robots, laser scanning, and automated 3D modeling. Früh has a PhD in electrical engineering and information technology from the University of Karlsruhe, Germany.*

Avideh Zakhor *is a professor in the department of electrical engineering and computer science at the University of California, Berkeley. Her research interests include image and video processing, compression, and communication. Zakhor received a PhD in electrical engineering from MIT.*

Readers may contact the authors at Univ. of Calif., Berkeley; 507 Cory Hall; Berkeley, CA 94720; {frueh, avz}@eecs.berkeley.edu.