Automating the 3D Modeling Pipeline

Peter Allen   Alejandro Troccoli   Paul Blaer
Department of Computer Science, Columbia University, New York, NY 10027

Abstract

We describe advances in automating the 3D modeling pipeline to create rich 3D textured models. Our work is aimed at large scale site modeling, where much manual effort is often needed to create complete models. We present i) an automatic 2D-3D registration method for texture using cast shadows as a cue to refine the registration parameters, ii) methods for change detection in an acquired model, and iii) a new mobile robot that can be used to automatically acquire data for modeling.

1. Introduction

At the Columbia Robotics Laboratory, we have been involved in a number of 3D modeling projects including reverse engineering of industrial objects [15], urban site modeling [18, 19], modeling of endangered Gothic Cathedrals [4], and reconstruction of archaeolgical sites [2]. Our experience has taught us the importance of automating as much of the modeling process as possible. Building 3D models from range scans and images is time-consuming and difficult, usually involving much manual effort. In a typical 3D modeling pipeline, geometry is acquired in the form of range scans that need to be registered together in a common coordinate system. Additionally, images provide texture information. To build a texture-mapped model, the images and the geometry have to be registered (i.e., a mapping from model space to image space has to computed). Part of our research focuses on developing new methods for reducing model-building time and improving accuracy [4]. In particular, texture mapping can pose difficult problems in providing complete and accurate coverage of a complicated model. Our method uses knowledge of shadows and sun position during the image acquisition process to correctly register the imagery with the constructed 3D model.

Much of our work is involved in historic and archaeological site modeling. Accordingly, we also want to record and keep track of the changes to the site as the researchers excavate the site, documenting the process. This problem also poses interesting technical challenges. Scanning the entire site every day is not desirable or practical. Instead we would like to acquire enough information to track changes, so as to be able to build a 3D model that represents a snapshot of the site at any given point in time.

Another way in which the pipeline can be automated is in the data acquisition phase. Rather than manually move sensors and plan viewpoints [16], we have built a mobile robot base with a 2D and 3D sensor suite that can locate and navigate itself to automatically acquire scans.

2. Previous Work

Over the past few years, a number of research teams have been addressing the use of range scans and images to develop 3D models for the virtual preservation of historic sites. Some notable projects include the modeling of Michelangelo's David and other statues by Levoy et al. [13], the IBM Pieta project of Bernardini et al. [6], the Great Buddha project of Ikeuchi et al. [12], and the virtualization of a Byzantine Crypt by Beraldin et al. [5]. Our goals overlap with the work of these researchers, but also differ in several ways. First, we are interested in recording an archaeological excavation in progress. Second, we want to keep track of changes as the excavation proceeds. And finally, we require that our models serve as a complement to the archaeologists' documentation, which means sharing the same reference coordinate system. The work of Acevedo et al. [1] is similar to ours with respect to these latter goals. However, they use photogrammetry, instead of range data from laser scans, to create their 3D model; hence, the technical challenges we each face are quite different. Image-based reconstruction for archaeology has also been addressed by Pollefeys et al. [14].

3. The 3D Modeling Pipeline

In the following sections, we present the 3D modeling pipeline and visualization facilities that we have developed. We illustrate them with examples based on data that we acquired on-site at the excavation of a 6th-4th century BC archaeological excavation at Monte Polizzo in western Sicily [2].

Geometry, in the form of point clouds, is acquired by a laser range finder, and texture, in the form of photographs, is obtained with a digital camera. Details for each stage are

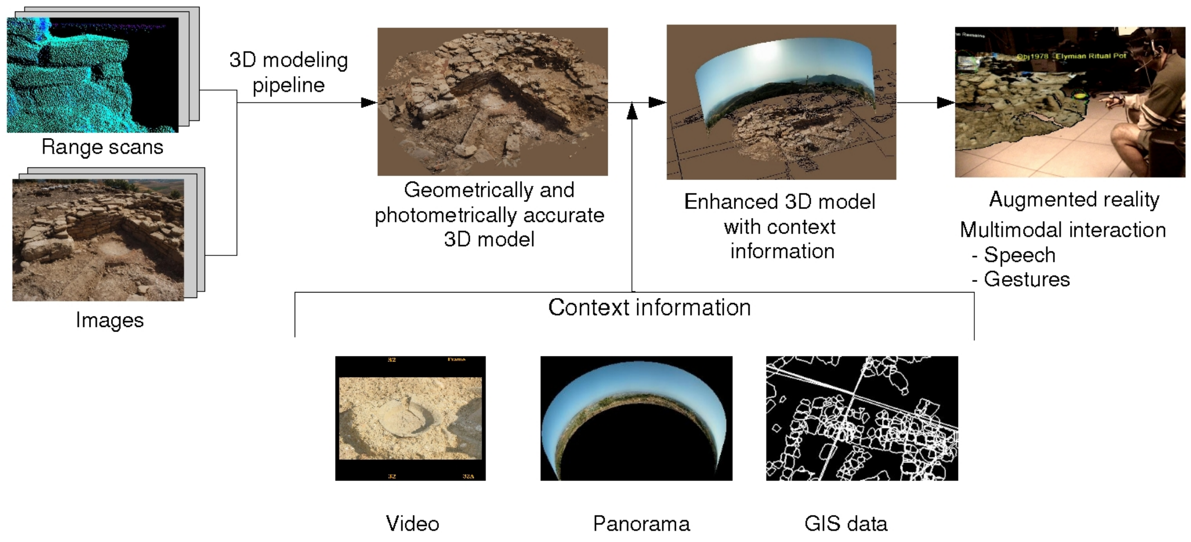

Figure 1. *Our 3D modeling and visualization pipeline. We start by building a textured 3D model using range scans and images, which we enhance with contextual information in the form of panoramic images, video, and GIS data. This context-rich model is then used as input to our multimodal augmented reality application.*

given below. A video of our model can be downloaded from www.cs.columbia.edu/∼allen/sicily.avi.

Scan acquisition. To model the acropolis at Monte Polizzo we used a time-of-flight laser scanner (Cyrax 2500) to measure the distance to points on the site. Data from the scanner comprises point clouds, with each point consisting of three coordinates (*x*,*y*,*z*) and a value representing the amplitude of the laser light reflected back to the scanner.

Scan registration. Multiple scans are required to completely acquire a site such as the Monte Polizzo acropolis. The resulting point clouds need to be registered together. Typically, the coordinate system of one of the point clouds is chosen as the coordinate system for the model. In archaeology, however, a global site coordinate system is set up from GPS data. A set of control points are accurately measured using (preferably differential) GPS, and are then used to initialize a total station (e.g., a Leica TCR 705 total station. A total station is a theodolite with an electronic distance meter that is used to measure points of interest, such as the location of findings, rocks, or the terrain contour. To register each point cloud with respect to the site's coordinate system. we use a set of targets that the scanner can automatically recognize, shown in Figure 2. Before taking a scan, we place the targets on the area we plan to cover, and use the total station to measure their positions in the site's coordinate system. Afterwards, we scan the scene at a low resolution to identify and acquire the targets' positions in the scanner's coordinate system, and so solve the 3D-to-3D registration problem. The targets are then removed and a full-resolution scan is acquired.

This technique allowed us to accurately register each individual point cloud with the site's coordinate system. For our purposes, it proved advantageous over pairwise or multi-scan registration using the iterative closest point algorithm (ICP) for several reasons. First of all, it required no scan overlap. This allowed us to take fewer scans, with greater freedom to choose the scanner position by eliminating the traditional overlap requirement of ICP. We did, however, acquire overlapping scans to minimize unseen regions (holes). In addition, as soon as we finished a scanning day, our model was completely registered. And finally, it allowed us to record changes easily by scanning only the affected areas. We would not have been able to track changes robustly if we had relied on ICP for scan registration, because ICP relies on point-to-point correspondences over an overlapping set of point clouds. If the point clouds represent different states of the site, these correspondences may not exist.

Surface generation. From sets of registered point clouds that represent the state of the site at the same point in time, we generated a triangular-mesh surface, using the VripPack software developed by Curless and Levoy [9]. VripPack outputs the best mesh that fits the point cloud data, smoothed to account for registration errors.

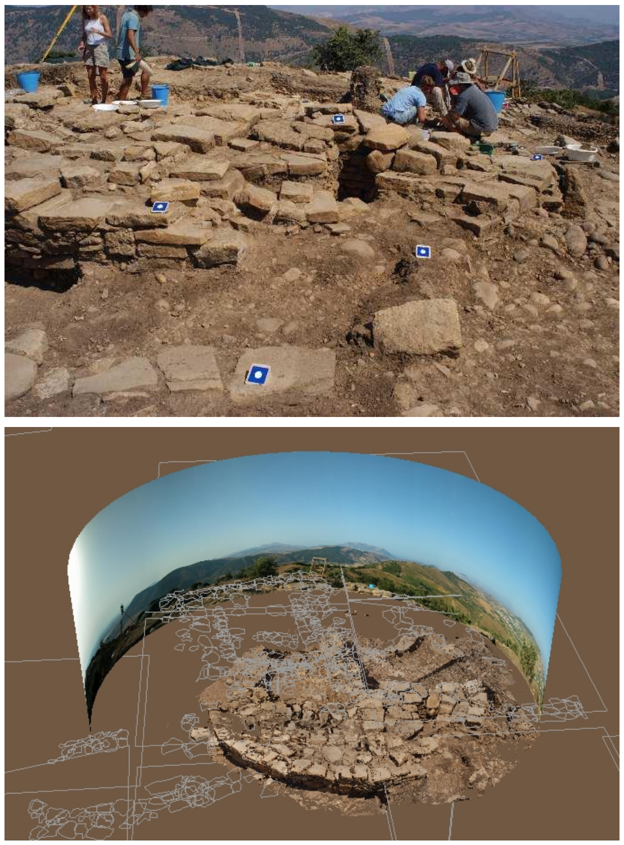

Figure 2. *Modeling the acropolis at Monte Polizzo. Top: Targets have been placed in the area to be scanned. Bottom: Final textured model with panoramic image as background and GIS data.*

3.1. Texture mapping

Texture acquisition. In addition to the scanner, we used a Nikon D100 digital camera, mounted on the scanner's case, to acquire texture information. For each scan we acquired, we took a photograph.

Local texture registration. Prior to our trip, we performed a simple calibration to estimate the camera's external and internal parameters. We determined the camera calibration by scanning a flat wall with the room lights off and the camera's shutter open for the eight-second duration of the scan. This provided us with an image of the grid pattern described by the laser as it sampled the wall and the 3D coordinates of each sample. We scanned again from a different distance and angle to acquire more samples. We then segmented the images to obtain the centroid of each grid point, and solved the calibration problem using the 2D-to-3D correspondences just obtained.

Global texture registration. While the local texture calibration procedure provided us with a good estimate of the camera's parameters, we found that our images were slightly misaligned with respect to the complete model. One reason for this is that our calibration was local to the scanner's coordinate system. To texture-map the final model, this local registration had to be transformed to the site's coordinates. Hence, any errors in scan-to-scan registration will also affect the texture registration. In addition, our initial calibration was accurate at the depths at which calibration points had been measured, but not as accurate at other ranges. To solve these misalignments, we developed a new method based on the shadows cast by the sun. Our method performs a global texture registration; it registers the texture with respect to the model's coordinate system, as opposed to the scanner's coordinate system. Since we have the latitude and longitude of the site and the time at which each photograph was taken, we can compute the location of the sun and find portions of the 3D model that should be in shadow. By matching these with the shadows in the image we solve the 2D to 3D registration problem.

Assuming the internal parameters of a camera are known, we find the camera position c with respect to the 3D model: $\mathbf{c} = (\phi_x, \phi_y, \phi_z, t_x, t_y, t_z)$. This is a six-parameter rigid body transform that maps a point X_w in world coordinates into its corresponding point X_c in the camera reference frame. The first three parameters (Euler angles) represent the angles of rotation about each of the coordinate axes and form a rotation matrix, $\boldsymbol{R}(\phi_x, \phi_y, \phi_z) = \boldsymbol{R_x}(\phi_x)\boldsymbol{R_y}(\phi_y)\boldsymbol{R_z}(\phi_z)$. The remaining three parameters are the components of a translation vector $\mathbf{t}$. Together, they satisfy the following relationship:

$$X_c = \boldsymbol{R}(\phi_x, \phi_y, \phi_z)X_w + \mathbf{t}.$$

If we knew the correct set of external camera parameters $(\phi_{x_f}, \phi_{y_f}, \phi_{z_f}, t_{x_f}, t_{y_f}, t_{z_f})$, then an orthographic view of a textured version of the model with the eye looking in the direction of the sun's rays should show no texture representing shadows. However, if the texture is misaligned, such a rendering will exhibit a number of shadow pixels. Our method exploits this idea by searching the parameter space for a point that minimizes the number of pixels representing shadows in the rendered image of the model.

The problem is properly stated as follows. If we let I denote the image to be registered and M the model, then f, our cost function, is defined as

$$f(I_r) = \sum_{x,y \in I_r} shadow(I_r, x, y),$$

where I_r stands for a rendered image of M as seen from the direction of the sun and textured with I using a texture camera with external parameters set to $\mathbf{c}$, and $shadow(I_r, x, y)$ is 1 if pixel(x,y) of I_r is in shadow, otherwise 0.

Given the initial estimate of the camera position $\mathbf{c}_0$ that we obtained off-line with our pre-calibration, the problem

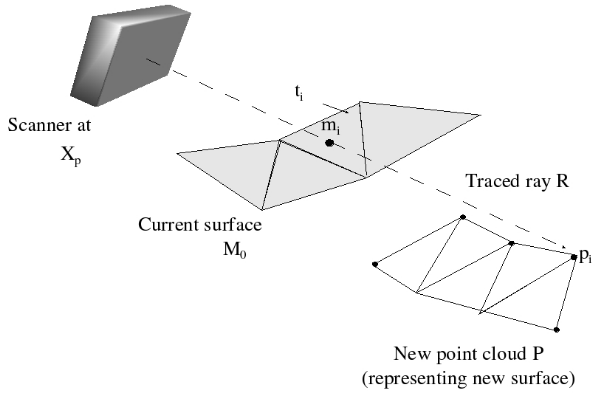

Figure 3. *Incremental changes in the model are detected by casting a ray from the scanner position to the model through point p_i, and finding its intersection point m_i.*

is to find a point $\mathbf{c}_f$ that minimizes f.

The complete shadow registration process consists of two stages: a preprocessing stage and a minimization stage. In the preprocessing stage, the shadows in the image are found using thresholding, and masked out with a given color. In the minimization stage, simulated annealing is used to search for a global minimum of f, starting from the initial estimate. We applied the algorithm to ten of the texture images we used to create our model. Figure 7 shows screenshots of the final textured model. A detailed analysis and quantitative results can be found in [20].

Texture-map generation. To create the final texture-mapped model, we assigned each mesh triangle one of the available textures. For each vertex v, we first find its valid image set I_v. An image i_k belongs to the valid image set of a vertex if the following three conditions are met: first, the projection of v must be within the boundaries of i_k; second, the normal n_v of v must be pointing towards the camera of i_k (i.e., the angle between the optical axis of of the camera and n_v must be less tan $\pi/2$); and, finally, there must be no other point in the mesh M in the line connecting the projection of v in i_k and v (these conditions are mentioned in [17]). We perform the last test using a ray-tracing operation accelerated with an octree. We then compute for every triangle t its valid image set I_t. An image i_k is in I_t if it belongs to the valid image set of each of the triangle's vertices. Finally, from the valid image set of each triangle, we choose the image i_k with the best pixel/area resolution. The final textured model is rendered using hardware-supported projective texture mapping.

3.2. Model change detection

We have also developed a method to track changes to the excavation site. Because of the large size of the Monte Polizzo site, we limited ourselves to a single structure, approximately 10 by 10 meters in area. We first acquired the scans necessary to build a full initial model of the structure. The archaeologists then proceeded to remove some stones and we acquired new scans. We did not scan the entire structure again; instead we scanned only those areas where stones had been removed. Using our change tracking method, we can recreate the site with and without the removed stones, simulating the archaeological process itself. Figure 4 shows renderings of the site model at different stages, where the numbers labels the stones that were removed. The first image shows the site in its initial state, the middle one shows the site after the stones were removed, and the right image shows the detected changes in gray.

During post-processing, we created an initial model M_0 of the structure by registering and merging the initial point clouds together. We then incrementally incorporated the changes to the model. From the initial model M_0 and a point cloud P that was acquired after the stones were removed, we detect the geometry corresponding to the removed stones, delete it, and add the geometry corresponding to the newly exposed surfaces to obtain a new instance M_1 of the model, in the following manner:

1. Let X_p be the position of the scanner when P was taken.
2. For every point p_i in P do
   (a) Trace a ray R from X_p through p_i (see Figure 3).
   (b) Find the triangle t_i of M_0 that R intersects.
   (c) Find m_i, the intersection point of R and t_i.
   (d) Compute the distance d_i between p_i and m_i.
   (e) If d_i is greater than a given threshold, and R passes first through m_i and then through p_i, remove all edges of triangle t_i from M_0.
3. Remove all non-connected vertices from M_0. Call this the intermediate model M_{im}.
4. Create a mesh M_p from P and use a package such as VripPack to merge M_{im} with M_p to obtain M_1.

These steps are repeated for all point clouds acquired after the stones were removed. The result is a new model that represents the updated state of the site.

3.3. Adding context information

3D modeling from range scans is a powerful tool, but a 3D model by itself lacks important context information. We

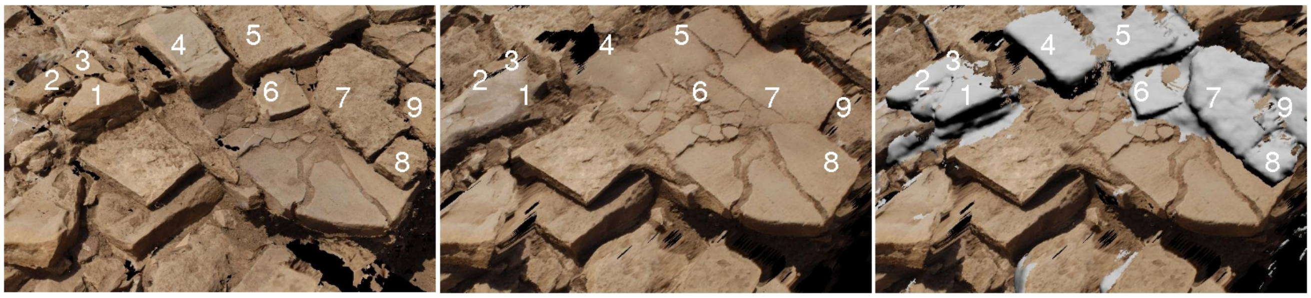

Figure 4. *Tracking changes. Left: Initial model. The stones to be removed are numbered. Middle: Model after stones have been removed. Right: Detected changes, in light gray and numbered. The careful observer will notice some small gray patches that do not represent removed stones. These are due to small misalignments and smoothing errors.*

can obtain this context by combining data from different sensors. In our pipeline, we combine our 3D model with surveying data, panoramic images, and digital video.

In current archaeological recording methods, a total station is used to lay out a site grid and record the 3D position of finds. The logged data is then used to make a site plan, typically using GIS software. We can incorporate this data in our modeling pipeline to add important and meaningful information. For example, by displaying the site grid, archaeologists that were not at the site can relate written reports to the 3D model. The GIS data and the 3D model are, by design, in the same coordinate system and no special registration is required.

Additional context information is provided by panoramic images. In the field, we acquired a complete panorama from a set of thirty eight images using a Kaiden QuickPan III spherical panoramic head. The final cylindrical panorama was created with PhotoStitch, a photo-stitching package. For registration purposes, we recorded the position of the camera using the total station. (We did not record the camera's orientation, which we had to find manually.) Since the camera was leveled, finding the correct rotation is a one–degree-of-freedom search (rotation about the camera's y axis), which we performed manually by rotating a textured-mapped cylinder around the model.

Finally, we used a color digital video camera to capture moving imagery of the excavation in progress, recording the position of the video camera with the total station.

4. Mobile Site Modeling Robot

Our goal is to eventually automate the entire modeling process, including the data acquisition. To address this issue, we have designed a mobile robot platform and a software system architecture that controls the robot to perform human-assisted or fully autonomous data acquisition tasks. The *AVENUE* (Autonomous Vehicle for Exploration and Navigation of Urban Environments) system plans a path to a desired viewpoint, navigates the mobile robot to that viewpoint, acquires images and three-dimensional range scans of the building(s), and then plans for the next viewpoint.

4.1. Platform Hardware

Our mobile robot, AVENUE [3, 11], has its base unit the ATRV-2 model (see Fig. 5). The base unit has an on-board computer, odometry from wheel encoders, and a set of sonar units located around the perimeter of the robot. In addition to these base features, we have added additional sensors including a differential GPS unit, a laser range scanner, a camera mounted on a pan-tilt unit, an omnidirectional camera, a digital compass, and two 802.11b wireless network cards. Figure 5 also shows a sample laser scan taken with this system on the Columbia University campus.

4.1.1. Localization and Navigation

The navigation portion of the AVENUE system [10] currently localizes the robot through a combination of three different sensor inputs. It makes use of the robot's built-in odometry, a differential GPS system, and a vision system. Odometry and GPS are the primary localization tools. The vision input is a useful supplement for localization when some preliminary data are available about the region and its structures. The vision system matches edges on nearby buildings with a stored model of those buildings in order to compute the robot's exact location. Initially the model is rough and approximate. It becomes more refined and precise as the actual model construction process progresses. However, to pick the correct building model for comparison, the robot needs to know its approximate location. Odometry can be problematic because of slippage. In urban environments with tall buildings, GPS performance can fail when not enough satellites can be seen. To alleviate these problems, we use our two-level, coarse-to-fine vision scheme that can supplement GPS and odometry for robot

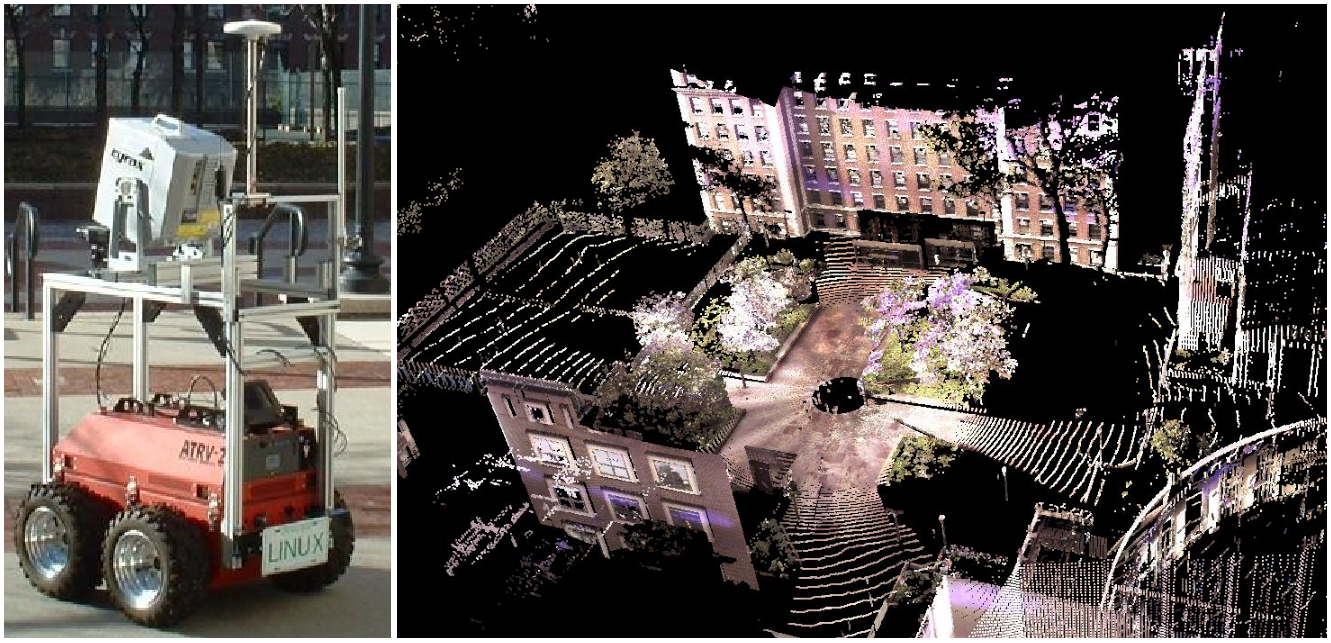

Figure 5. The ATRV-2 Based AVENUE Mobile Robot (left). A sample laser scan taken with the AVENUE system on the Columbia University campus (right). The hole at the center of the scan is where the scanner was positioned.

localization. First we topologically locate the robot [7] with a coarse position estimate and then we use this estimate as the initial approximation for the precise localization which matches building edges and models.

Our coarse localization method involves building up a database of reference omnidirectional images (such as Fig. 6) taken throughout the various known regions that the robot will be exploring at a later time. Each reference image is then reduced to three histograms, using the Red, Green, and Blue color bands. Omnidirectional images are used because they are rotation invariant in histogram space. When the robot is exploring those same regions at a later time; it will take an image, convert that to a set of three histograms, and attempt to match the histograms against the existing database. The database itself is divided into a set of characteristic regions. The goal is to determine in which specific physical region the robot is currently located. This method was improved by looking at the histograms of each image at multiple resolutions rather than just at the image's original resolution.

This method still suffers somewhat from sensitivity to outdoor lighting changes. It also can have some difficulty distinguishing between very similar looking topological regions. Therefore, we have developed an additional system [8] to be used as a secondary discriminator. We have chosen to utilize information from wireless ethernet networks, which are becoming very common in urban environments. A profile of the signal strengths of nearby access points is constructed and then used for matching with an existing database.

The combination of these two systems allows us to topologically localize the robot with good accuracy. To test the system, we took readings in 13 different regions throughout the northern half of the Columbia Campus and used each localization method alone and then the combination of both methods. On average, the correct region was identified by our combined method 89% of the time.

5. Summary and Conclusions

We have described an integrated 3D modeling and visualization pipeline for archaeology, which we have applied to digitally recording an excavation in progress at Monte Polizzo, Sicily. The area of 3D modeling for cultural heritage preservation is evolving rapidly and we believe that new tools such as these will be an important resource for future archaeological research. Our work is unique in a number of ways, because it 1) integrates a variety of different data sources: range scans, images, GIS data, and video, 2) develops a new technique for image–to–model-base registration based on the shadows cast by the sun, 3) includes a temporal change racking component, and 4) presents a novel mobile robot to help automate the data acquisition phase of site modeling.

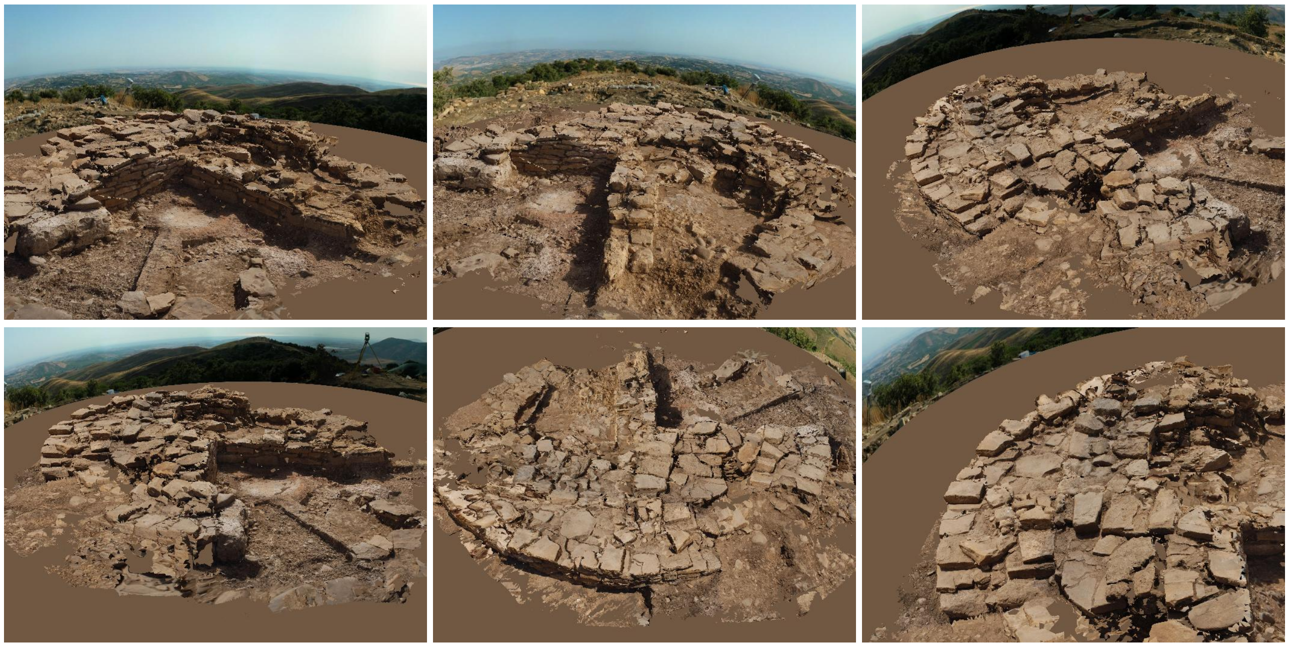

Figure 7. *Screenshots of our 3D model with the panorama in the background.*

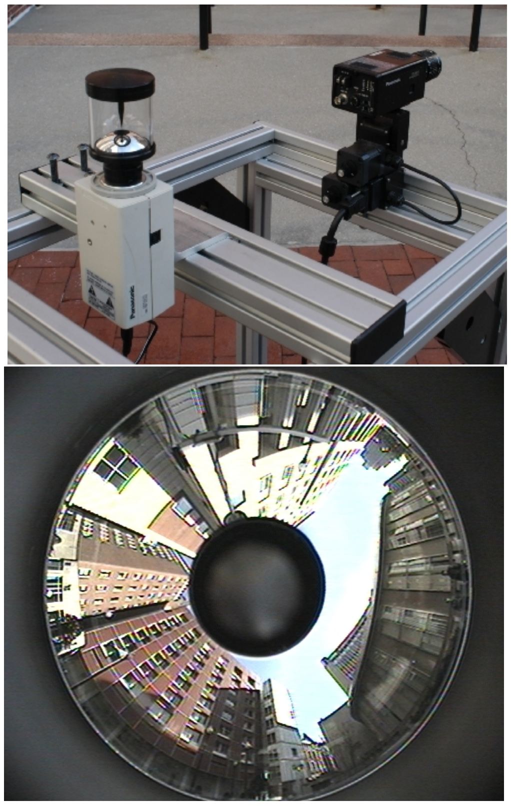

Figure 6. Our robot's omnicamera (left) and a typical image from that camera (right).

Acknowledgments

Steve Feiner, Ed Ishak, Hrvoje Benko, Ben Smith, Lynn Meskell, and James Conlon assisted with the field work at Monte Polizzo. Atanas Georgiev was instrumental in developing the software for the Avenue control system. Ian Morris and the Stanford Archeology Center graciously provided access to and images and data from Monte Polizzo. This research was funded in part by NSF grants IIS-0121239 and ANI-00-99184.

References

[1] D. Acevedo, E. Vote, D. H. Laidlaw, and M. S. Joukowskyy. Archaeological data visualization in VR: Analysis of lamp finds at the Great Temple of Petra, a case study. In *Proc. IEEE Visualization '01*. IEEE Computer Society, 2001.

[2] P. Allen, S. Feiner, A. Troccoli, H. Benko, E. Ishak, and B. Smith. Seeing into the past: Creating a 3D modeling pipeline for archaeological visualization. In *Proc. 3D Data Processing, Visualization and Transmission Symposium*, Sep. 6-9 2004.

[3] P. K. Allen, I. Stamos, A. Gueorguiev, E. Gold, and P. Blaer. Avenue: Automated site modeling in urban environments. In *3DIM, Quebec City*, pages 357–364, May 2001.

[4] P. K. Allen, A. Troccoli, B. Smith, S. Murray, I. Stamos, and M. Leordeanu. New methods for digital modeling of historic sites. *IEEE Comp. Graph. and Applic.*, 23(6):32–41, 2003.

[5] J.-A. Berladin, M. Picard, S. El-Hakim, G. Godin, G. Valzano, A. Bandiera, and D. Latouche. Virtualizing a Byzantine crypt by combining high-resolution textures with

laser scanner 3D data. In *Proc. VMMS 2002*, pages 3–14, September 2002.

[6] F. Bernardini, H. Rushmeier, I. M. Martin, J. Mittleman, and G. Taubin. Building a digital model of Michelangelo's Florentine Pietà. *IEEE Comp. Graph. and Applic.*, 22(1):59–67, Jan/Feb 2002.

[7] P. Blaer and P. K. Allen. Topological mobile robot localization using fast vision techniques. In *IEEE ICRA*, pages 1031–1036, May 2002.

[8] P. Blaer and P. K. Allen. Topbot: automated network topology detection with a mobile robot. In *IEEE ICRA*, pages 1582–1587, September 2003.

[9] B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *Proc. SIGGRAPH 96*, pages 303–312. ACM Press, 1996.

[10] A. Georgiev and P. K. Allen. Vision for mobile robot localization in urban environments. In *Int. Conf. Intelligent Robots and Sytems (IROS 2002)*.

[11] A. Gueorguiev, P. K. Allen, E. Gold, and P. Blaer. Design, architecture and control of a mobile site modeling robot. In *IEEE ICRA*, pages 3266–3271, April 24-28 2000.

[12] K. Ikeuchi, A. Nakazawa, K. Nishino, and T. Oishi. Creating virtual buddha statues through observation. In *IEEE Workshop on Applics. of Comp. Vision in Architecture*, volume 1, 2003.

[13] M. Levoy, K. Pulli, B. Curless, S. Rusinkiewicz, D. Koller, L. Pereira, M. Ginzton, S. Anderson, J. Davis, J. Ginsberg, J. Shade, and D. Fulk. The digital Michelangelo project: 3D scanning of large statues. In *Proc. Siggraph 2000*, pages 131–144, 2000.

[14] M. Pollefeys, L. Van Gool, M. Vergauwen, K. Cornelis, F. Verbiest, and J. Tops. Image-based 3D acquisition of archaeological heritage and applications. In *Proc. 2001 Conf. on Virtual reality, Archeology, and Cultural Heritage*, pages 255–262. ACM Press, 2001.

[15] M. Reed and P. K. Allen. 3-D modeling from range imagery. *Image and Vision Computing*, 17(1):99–111, February 1999.

[16] M. K. Reed and P. K. Allen. Constraint based sensor planning for scene modeling. *IEEE Trans. on PAMI*, 22(12):1460–1467, 2000.

[17] C. Rocchini, P. Cignomi, C. Montani, and R. Scopigno. Multiple texture stitching and blending on 3D objects. In *Rendering Techniques '99*, Eurographics, pages 119–130. Springer-Verlag Wien New York, 1999.

[18] I. Stamos and P. K. Allen. Registration of 3D and 2D imagery in urban environments. Vancouver, Canada, 2001.

[19] I. Stamos and P. K. Allen. Geometry and texture recovery of scenes of large scale. *Computer Vision and Image Understanding(CVIU)*, 88(2):84–118, 2002.

[20] A. Troccoli and P. K. Allen. A shadow based method for image to model registration. In *IEEE Workshop on Image and Video Registration, Conf. on Comp. Vision and Pat. Recog. (CVPR)*, 2004.