ELSEVIER

ISPRS Journal of Photogrammetry & Remote Sensing 53 (1998) 379–391

PHOTOGRAMMETRY
& REMOTE SENSING

A multi-sensor approach to creating accurate virtual environments [1]

Sabry F. El-Hakim [a,*], Claus Brenner [b], Gerhard Roth [a]

[a] *Visual Information Technology Group, IIT, National Research Council, Ottawa, Ontario, K1A 0R6, Canada*
[b] *Institute for Photogrammetry, University of Stuttgart, Geschwister-Scholl-Str. 24(D), 70174 Stuttgart, Germany*

Received 1 July 1998; accepted 8 September 1998

Abstract

Creating virtual environment models often requires geometric data from range sensors as well as photometric data from CCD cameras. The model must be geometrically correct, visually realistic, and small enough in size to allow real-time rendering. We present an approach based on 3D range sensor data, multiple CCD cameras, and a colour high-resolution digital still camera. The multiple CCD cameras provide images for a photogrammetric bundle adjustment with constraints. The results of the bundle adjustments are used to register the 3D images from the range sensor in one coordinate system. The images from the high-resolution still camera provide the texture for the final model. The paper describes the system, the techniques for the registration of the 3D images, the building of the efficient geometric model, and the registration and integration of the texture with a simplified geometric model.

Keywords: virtual environments; geometric modeling; range images; texture mapping; registration; integration

1. Introduction

Virtual Environment (VE) systems generate a computer simulation of scene data, place the user within the simulation, and allow the user to manipulate objects within the environment. The technology allows people to discover new ways to experience real-world and simulated phenomena and will change the nature of computing (Sheehan, 1997). Recent advances in three-dimensional displays, real-time texturing and computer graphics hardware, and the increasing availability of modeling, and rendering software tools, have resulted in an increased demand for the contents of 3D virtual worlds. In particular, the demand for real-world-based contents, rather than synthetically generated contents, is rapidly increasing. This is because real-world data have the potential to generate realistically looking models in a more automatic and faster manner than the labor-intensive, time-consuming graphic-based detailed contents. However, in most applications of virtual environments, large and complex 3D models are required. Even with the increasing capabilities of computer hardware, it is not possible to render all of the geometry of these arbitrarily complex scenes at highly interactive rates of at least 20 frames per second even with high-end computer graphics systems. Keeping in mind the restrictions on data size, the challenge is to create environments that are not only geometrically correct, but also visually realistic.

[1] Revised version of a paper presented at the ISPRS Commission V Symposium, June 2–5, 1998, Hakodate, Japan.

* Corresponding author. Tel.: +1 613 991 6381; Fax: +1 613 952 0215; E-mail: sabry.el-hakim@iit.nrc.ca

0924-2716/98/$ – see front matter
PII: S0924-2716(98)00021-5

1.1. The process of creating virtual environments

Depending on the application and the complexity of the environment, achieving geometric correctness and realism may require a large number of images from various types of sensors, such as range sensors and scanners, video cameras, and digital color cameras. Therefore, to generate a large complex virtual environment from real-world data, the following issues must be addressed (Fig. 1): (1) data collection from various types of sensors.; (2) registration of all the data in a single reference system; (3) representation, or modeling, of the virtual environment that is geometrically correct, visually realistic, and can be rendered in real-time.

To achieve geometric correctness, proper registration and integration of the data from the various sensors must be carried out. Assuming this has been achieved, to correctly cover all the details, a geometric model containing a large number of polygons is produced. Needless to say, this results in a virtual environment that is often too large for real-time interaction or even visualization and walk-through. Several approaches can be used to reduce this problem. They can be summarized as follows.

(1) Applying an efficient polygon simplification method that simplifies the polygonal geometry of objects depending on their size or distance to the viewer with minimal loss of visual contents. This process is known as level of detail (LOD) hierarchy. Most methods may be divided into those based on decimation, or removing polygons and re-triangulating the resulting hole (e.g., Soucy and Laurendeau, 1996), and those that merge, or collapse, several vertices together into one vertex (e.g. Hoppe, 1996). The criteria, or constraints, used by each method vary depending on the desired balance between accuracy and speed and whether the topology is to be preserved. Heckbert and Garland (1997) and Luebke (1997) provide good surveys of existing methods.

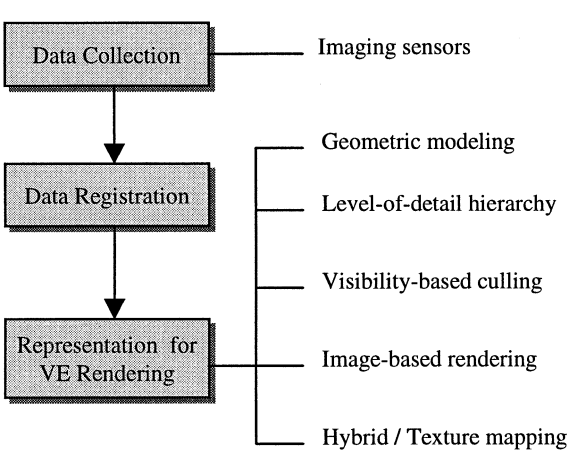

Fig. 1. The virtual-environments creation process.

(2) Applying an efficient and precise visibility-computation technique that determines the visibility of all parts of the model from a given observer's point of view. This is used for culling away polygons or surfaces on the back face of objects (Kumar et al., 1996), and those occluded by other objects or outside the viewing frustum (Teller and Sequin, 1991; Zhang et al., 1997).

(3) Applying image-based rendering (IBR), where images are used directly to generate photo-realistic views for rendering without a geometric model (Chen and Williams, 1993; McMillan and Bishop, 1995; Szeliski, 1996; Kang, 1997). The technique relies on automatic stereo matching which, in the absence of geometric data, requires a large number of closely spaced images to work properly.

(4) Replacing some of the geometry with texture maps, or applying a combination of methods (1) and (3) above. This is suited for more applications than either image-based or geometry-based approaches alone. The image-based rendering is usually applied to surfaces at a large distance or those that the user is not interacting with. Aliaga and Lastra (1997) used this approach by applying image-based rendering to views seen through openings such as doors and windows. Debevec et al. (1996) applied photogrammetry to generate a basic 3D model and provide constraints for the stereo matching required for IBR. Therefore, their approach requires only a small number of overlapping images compared to IBR methods where no geometric data is used.

Most of the above techniques will work well for relatively small and simple environments, but will have varying degrees of success on large and complex environments. Research is continuing in all the four categories to accommodate the ever-increasing demand for real-time interaction with large complex environments. For these environments, we believe that a combination of all these techniques is the best solution.

1.2. Paper overview

Our VE research addresses all the three issues of creating virtual environments: the data collection, the registration, and the VE representation. The goal is to develop a complete and flexible system that enters a site, uses the appropriate sensors to image it, and accurately, easily, and rapidly generate a complete virtual representation of that site. In earlier work (El-Hakim et al., 1997), we focused on the data collection and registration components of our system. In this paper, more emphasis is given to the VE representation, particularly the efficient polygon simplification and replacing of geometry with texture maps. In Section 2, a short overview of the system used for data collection in a complex environment will be given. The system is designed for complete geometric and texture acquisition. The overall procedure, from data collection to VE model creation, is summarized in Section 3. We then describe our approach for producing fully registered 3D and texture images in Section 4. In a 3D image, each pixel has X, Y, and Z coordinates while in a texture image each pixel has a gray or color value. An algorithm for building an efficient non-redundant triangular mesh-model from a large number of registered, overlapped, 3D images is presented in Section 5. A texture correction and mapping approach that generate a seamless uniform texture map from numerous images, and accurately place it and warp it on the triangulated model is described in Section 6. Test results and analysis, followed by concluding remarks and future work are then presented.

2. The data collection and registration (DCR) system

The design objective of the DCR system is to acquire geometric and photometric data from relatively large indoor environments and output registered 3D and 2D images. The system must be: (1) flexible or easily configurable to various applications; (2) portable and light weight, but also rugged and stable.

To satisfy the flexibility requirement, different types of sensor, mainly laser scanners and analogue and/or digital CCD cameras, either monochrome or color or both, will be necessary. Other systems that combine these types of sensor for object modeling have been covered in the literature (e.g. Sato et al., 1997). However, for indoor site modeling, systems with multiple types of sensor are seldom-reported (Ng et al., 1998). Most existing approaches are based on one type of sensor, such as range finders (Johnson et al., 1997; Miyatsuka et al., 1998) or 2D intensity images, such as the image-based rendering approach mentioned in Section 1.1. Since no one type of sensor is suitable for all environments and objects, our design combines various 3D and 2D imaging technologies. The 3D data can be obtained by either the scanning range finder resulting in 3D points organized as an image, or by photogrammetry from overlapped 2D images resulting in sparse unorganized points, or both. The texture may be obtained directly from the range sensor, from the photogrammetric images, or from a high-resolution digital color camera. The latter provides the highest degree of realism. Using a system cart with a range of easy to mount accessories satisfied the portability and ruggedness requirements. The top of the cart is fitted with a breadboard to which strong lightweight beams can be easily attached and configured to satisfy the application requirements. All sensors are mounted on these beams with compatible mounting devices.

The current system consists of the following (Fig. 2).

(1) A Rackmount Pentium-II industrial PC and LCD touch-screen. Up to 12 GB of data can be stored on the PC. An external storage medium using 1 GB cartridges is used for data overflow.

(2) Twelve standard analogue, monochrome, CCD cameras.

(3) One Biris laser range sensor (Blais et al., 1991, 1996) mounted on a computer-controlled pan-tilt unit. This version of Biris has been modified to increase its range to 3.2 m. The Biris sensor produces the X and Z coordinates of a line of pixels in 1/60 s. The pan-tilt unit scans this line in the Y direction, thus generating a raster of 3D points, or a 3D image.

(4) One digital still color camera, with CCD resolution of 1160×872 pixels, mounted on a second computer-controlled pan-tilt unit.

(5) Cart, beams of various lengths, mounting devices, and power supplies.

The total weight of all the above components is about 100 kg, and two persons can handle it.

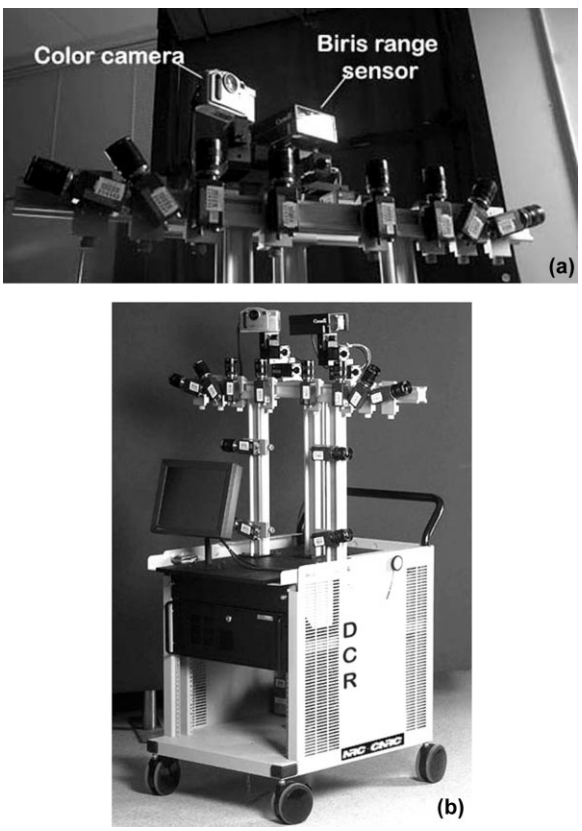

Fig. 2. The DCR System: (a) the sensors, (b) the complete system.

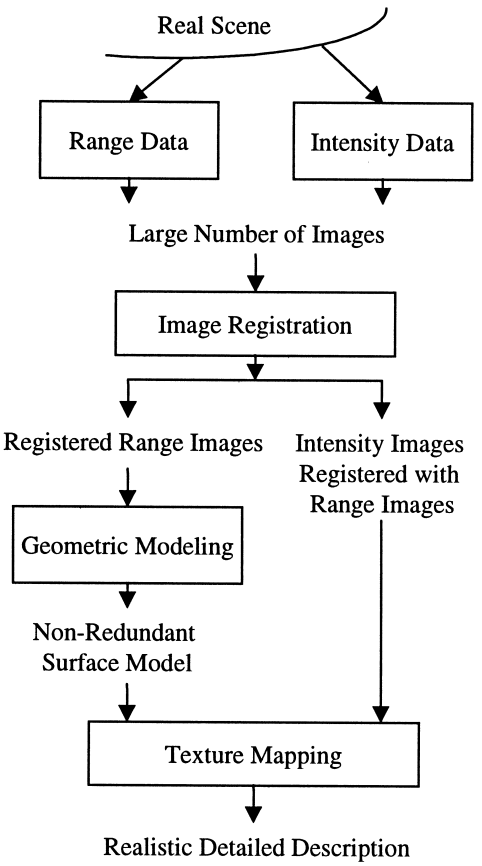

Fig. 3. Imaging and modeling procedure.

3. Summary of the overall procedure

Fig. 3 summarizes our overall procedure for data acquisition, registration, and modeling:

(1) All the sensors are positioned and configured on the mounts to completely cover a section of the site. The same part of the scene should be approximately covered by images from the range sensor, the analogue CCD cameras, and the color still-camera. The twelve analogue CCD cameras are positioned first to cover the scene with about 60% overlap between images. The range sensor and the color camera are then positioned, using the computer-controlled pan-tilt units, to cover approximately the same scene. The overlap between the range sensor images and between the color images is usually 20%.

(2) The cart moves to the next part of the site, and the image acquisition is repeated so that the new images overlap with the images at the previous position. This overlap between the analogue CCD images in two cart positions should be 20–60%, depending on the required accuracy and complexity of the scene.

(3) Once the site is completely covered, the registration procedure is applied. The images from the twelve analogue CCD cameras play a major role here. The result of the registration procedure is that all the intensity images and the 3D images from the range sensor are positioned and oriented in the same coordinate system. This procedure is described in Section 4.

(4) The registered 3D images are used to generate the geometric model: a non-redundant triangulated mesh. Section 5 describes this procedure.

(5) The registered intensity images from the digital color camera are mapped on the geometric model

using an advanced texture mapping procedure that gives the model the realistic look and adds surface details with a minimum number of triangles. In some applications where color is unimportant, such as some industrial applications and mining automation, the monochrome images are used for texture to reduce the size of data. Section 6 describes this procedure.

4. The image registration technique

The analogue CCD cameras within one cart position are mounted so that they produce a strip of images. These images are pre-calibrated, i.e. their parameters relative to each other are known. On the other hand, relative location parameters of images in different strips (different cart positions) are unknown. These parameters are computed from bundle adjustment, using camera locations in one cart position as constraints. Additional constraints from the known relationships between points, as determined from range sensor data, are utilized to strengthen the bundle solution. The procedure (Fig. 4) was described in details in an earlier publication (El-Hakim et al., 1997) and is only summarized here.

(1) The relationships between the images of the CCD cameras are accurately pre-determined by the calibration procedure. The calibration procedure may be carried out at any cart position once the cameras are securely mounted in the appropriate positions. No need for calibration points with known absolute coordinates since there is sufficient overlap between the images and there are plenty of relative 3D coordinates from the range sensor. This can be thought of as a self-calibration procedure. The relationships between all sensors are also determined here.

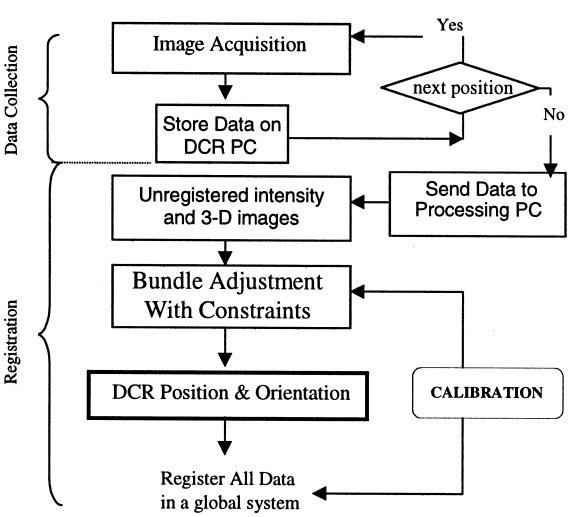

Fig. 4. The image registration procedure.

(2) To add geometric strength to the triangulation, data from the range sensor are used. It adds constraints in the form of geometric relationships between points, such as distances.

(3) Sets of equations representing the above two types of constraints are combined with the bundle adjustment basic equations to strengthen the solution and minimize error propagation, particularly since no external control points are used.

(4) As a result of the bundle adjustment, all the analogue CCD images are now in the same coordinate system. Since the relationship between the range-sensor images, the color images, and the analogue CCD images are known, it is now possible to register all range images in the same coordinate system.

5. The surface geometric modeling approach

In order to make practical use of the registered 3D data it is necessary to construct a geometric model from this data. If the 3D data are presented as a set of images it is trivial to create a triangular mesh by simply triangulating each image. However, since there is often considerable overlap between the 3D images from different views, a mesh created in this fashion will have many redundant faces. It is desirable to create a non-redundant mesh, in which there are no overlapping faces.

We have created a voxel-based mesh creation algorithm, which has the following characteristics (Roth and Wibowo, 1995, 1997).

(1) It uses a simple voxel data structure, which is very efficient in both space and time.

(2) It is able to process 3D data in raster, profile and point cloud format.

(3) It has a number of different ways of handling noisy and spurious 3D data points.

(4) It can fill holes in the triangulation to close the mesh and create a true volumetric model.

(5) It can report the accuracy of the triangular mesh relative to the original 3D data.

(6) It can handle 3D data that have an associated intensity or color value.

The basic data structure we use is a voxel grid of fixed dimensions in x, y and z. This voxel grid will contain the original data points, along with the mesh triangles. We have computed the percentage of occupied voxels for a large number of different objects and voxel grid sizes. In general, we find that between 1% and 6% of the total number of possible voxels are occupied. Since there may be many points in a voxel, the number of 3D data points is often much larger than the number of voxels. Therefore, storing only the occupied voxels enables our approach to handle very large 3D data sets. With this voxel grid as the underlying data structure, the following sequence of operations is executed to create the triangular mesh:

(1) Set the voxel size automatically or manually.

(2) Add each data point to the appropriate voxel.

(3) Eliminate spurious data points.

(4) Compute the local normal for each data point.

(5) Smooth the normals with a relaxation algorithm.

(6) Run the marching cubes algorithm to get the surface triangulation.

(7) Close any small holes that exist.

(8) Remove small isolated triangle regions (indicate erroneous data).

(9) Find the mesh accuracy relative to the 3D data.

The marching cubes algorithm (Lorensen and Cline, 1987) is used to generate the triangles for each voxel. Marching cubes is an iso-surface algorithm that extracts the zero set of a signed distance function. In this application, the signed distance function must be created from the 3D data points and their normals. For each voxel vertex this signed distance, which we call the field value, is computed by taking the weighted average of the signed distances of every point in the eight neighbouring voxels. Once the field value at each voxel vertex is known, then a linear interpolation process finds the intersection of the underlying surface with each edge of the voxel. Each of these intersection points is a vertex of the final triangulation. The triangles that approximate this surface in the voxel are found using a lookup table. The lookup table is made of all the possible cases of vertex signs. Since we have eight vertices of which each can be either positive or negative, we have 256 cases. However, because of similarities, only fourteen unique cases exist.

Spurious data points can occur because of errors in the acquisition process. Since spurious points often produce small and isolated triangle regions simply removing these triangles also removes such data points. However, spurious points that are very close to a valid surface can not be removed in this fashion. In such cases we use a visibility heuristic, which states that a voxel that is pierced many times by rays joining the range sensor origin to other voxels is likely to be invalid, and therefore contains spurious data points. These two heuristics are quite successful in removing spurious data points in the majority of cases.

The goal in mesh creation is usually to achieve specified mesh accuracy relative to the original data. Typically, this required accuracy is in the range of 1/10 mm to 2 mm. Note that when we speak of modeling accuracy we are talking about the faithfulness of the final triangulation relative to the 3D data. That is not the same as the accuracy of the original 3D data relative to the true object geometry.

The accuracy of the created mesh relative to the 3D data depends on the voxel size. It is possible to increase the mesh accuracy by simply reducing the voxel size. However, the voxel grid size must be two to three times greater than the sampling density of the 3D data. For this reason, all voxel algorithms by their nature tend to smooth the original data. This does blur small details, but also reduces noise. This is a limitation of all voxel approaches to mesh creation. Since 3D data are usually oversampled, a mesh of the desired accuracy can usually be obtained.

It is essential to cope with holes in the data coverage. We are able to close small holes in the final triangulation; however, large regions of the object surface that have no 3D data can not be closed properly. In such cases, the best solution is to obtain more 3D data by rescanning. We have taken 3D data in both cloud and image formats from various sensors and created a number of mesh models. In general, the results validate our claim that our method is an order of magnitude faster than others in the literature, and that it can deal with spurious, noisy and missing data points.

6. Texture corrections and texture mapping

While the generated geometric model is useful for managing the 3D construction of the VE, many details can only be viewed from mapping light-intensity data, or texture, on the model. Texture mapping is also an efficient way to achieve realism with only a low resolution, faster to render, geometric model. Traditional texture mapping techniques have focused on generating artificial shading, shadows, and other computer-generated effects. More recently, the interest has shifted to techniques that map real-scene images onto the geometric model, also known as image perspective techniques (IPT). High-resolution gray-scale or color images can be precisely mapped into the geometric model provided that the camera position and orientation are known in the coordinate system of the geometric model. In our system, these data are available since the pose parameters for each intensity image are computed in the registration procedure. Given the 3D coordinates of the vertices of a polygon, the corresponding projections of these vertices in an intensity image can be located. The light intensity values within the area defined by these projected vertices are stretched, rotated, and warped to fit into its counterpart 3D polygon. For reviews of the various texture mapping techniques, see Lansdale (1991), Haeberli and Segal (1993) and Weinhaus and Devarjan (1997).

In principle, the following algorithm could be used for texture mapping. For each 3D triangle t the following should be done.

(1) Select one image i from the set of images taken from the scene in which triangle t appears. Usually the image where triangle t appears largest is selected.

(2) Using exterior orientation, determine the correspondence between 3D triangle vertex coordinates in space and 2D coordinates in image i.

(3) Specify 3D and texture coordinates in a suitable modeling language such as VRML.

(4) View the scene using a standard viewer.

However, due to the following considerations, this simple approach is not feasible in most cases:

- The correct mapping between the plane triangle in which t lies, and the image plane of image i is given by a projective transform. Since viewers do not use this transform, simply specifying texture coordinates will lead to a wrong mapping with distortions visible at triangle edges (even at edges between triangles mapped from the same image).
- When standard lenses are used for the cameras, lens distortion parameters have to be applied, else distortions will be visible at common edges of adjacent triangles mapped from different images.
- Usually, it is desirable to have a constant texel-size on the object. This results in a more uniform appearance of the rendered model, since differences in detail due to varying camera distances are minimized. It also makes it possible to control file size and rendering speed more precisely, since the amount of texture data required is approximately given by model surface area divided by texel-size.

Considering this list of error sources, none of which is handled by viewers during scene rendering, it is clear that image warping has to be done in advance, i.e. all triangle texture maps have to be computed independently of the viewer texture mapping routines. Even when correct modeling of exterior, interior and additional camera parameters is used, there are still problems in practice that may lead to geometric and radiometric discontinuities which can easily disturb the impression of looking at a ‘real’ scene. In particular, radiometric differences between the images from different cameras lead to radiometric differences along triangle edges. Also, too large deviations of the underlying triangle mesh from the true object surface give rise to geometric errors, for example parts of the object’s surface appear in more than one triangle texture. Table 1 summarizes some error sources and the techniques we adopted to minimize their visual impact. We address each of these problems in the following sections.

6.1. Proper geometric fit

As discussed above, image warping is more appropriate if executed independently of the transformation applied by the viewer. To that end, the employed method defines a local texel coordinate system for each 3D triangle. The texel size, in object coordinates, can be set to the desired resolution. Each texel is then computed using exterior and interior orientation, including lens distortion parameters obtained from camera calibration. As seen in

Table 1
Error sources for visual discontinuities in mapped scenes and techniques used to minimize their visual impact

Error source	Visible at triangle edges	Type	Technique used
Wrong mapping (viewer)	Always	Geometric	Warping according to collinearity equations
Lens distortion	When mapped from different images	Geometric	Application of additional parameters
Radiometric differences between cameras	When mapped from different images	Radiometric	Global gray-value adaptation, blending
Non-uniform radiometry across single camera images	Always	Radiometric	Local gray-value adaptation, blending
Large deviations of triangle mesh from true surface	When mapped from different images	Geometric	Local triangle re-assignment, blending

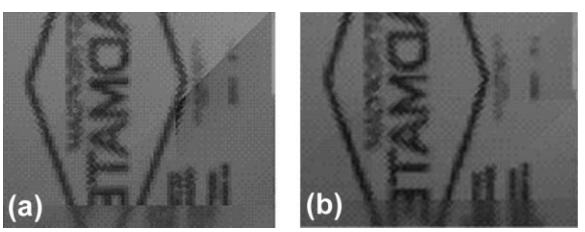

Fig. 5. Ensuring geometric fit by using distortion parameters: (a) without distortion parameters, (b) with distortion parameters.

Fig. 5, there is a clearly discernible difference between triangles mapped with and without distortion parameters.

6.2. Radiometric differences

Usually, radiometric discontinuities result along common edges of adjacent triangles mapped from different images, as shown in Fig. 7a. The main reasons for this are: (1) radiometric differences between cameras; (2) non-uniform response of each camera across the image plane; (3) different sensed brightness due to different camera positions.

The first item above may result from different aperture settings. However, since in our case video cameras with automatic gain control are used, the radiometric differences have to be modeled on a per-image basis rather than per camera. We address this problem by a method termed 'global gray-value adaptation'. The second item is most often caused by a brightness decrease from the image center to image borders. The third item is due to different orientation relative to surface normal vector. Both the second and third items can be tackled by a radiometric correction on a per-triangle basis, termed 'local gray-value adaptation'.

The global gray-value adaptation estimates gray-value offsets between images. The gray-value differences along the border of adjacent regions (a region is a set of triangles from one image) are minimized by least-squares adjustment (Fig. 6). The adjustment is much like in the case of a geodetic height network, where the observed height differences correspond to gray-value differences along region borders in our case. The adjustment is carried out simultaneously on all the regions to ensure one gray-value offset per region. The gray-value offsets determined at the borders are then applied to all the triangles in each corresponding region. Fig. 7 shows the result of this operation.

The local gray-value adaptation modifies the gray-values of each triangle to ensure smooth transitions to all adjacent triangles. However, this is not straight-

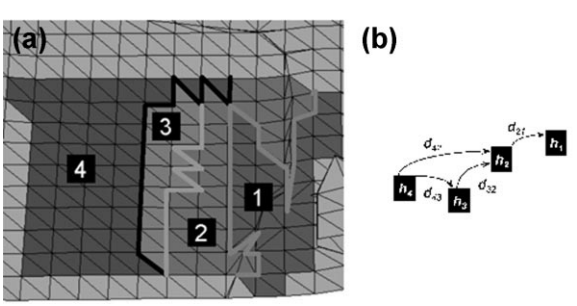

Fig. 6. Global gray-value adaptation: (a) regions and borders formed by triangles mapped from the same image, (b) corresponding observations d_{ij} and unknowns h_i (i and j represent regions).

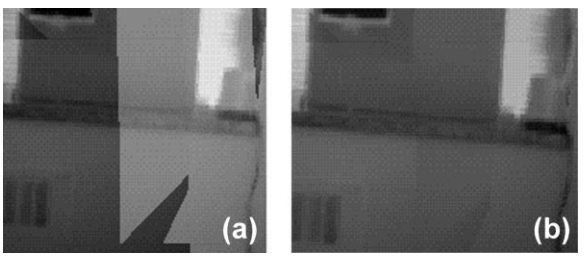

Fig. 7. Global gray-value adaptation: (a) without gray-value adaptation, (b) with global adaptation.

forward since if we observe offset o_1 along triangle edge e_1 and o_2 along e_2 it is unclear how to correct the gray-values in the vicinity of the triangle vertex where e_1 and e_2 intersect. Thus, we have adopted a technique that relies on iterative least-squares estimation. In order to force a gradual change to gray-values within a triangle, we fit a plane to the gray-value offsets observed at the triangle borders. The plane parameters are determined by a least-squares adjustment that minimizes these differences. After correcting the gray-values according to the plane parameters, this process is iterated several times. Usually, there are no discernible changes after a few iterations. Fig. 8 shows the effect of both global and local adaptation.

6.3. Handling approximated surface geometry

Since the triangulated mesh used in our application only approximates the actual surface, there will always be geometric errors in the mapped texture. Those errors arise at common edges of adjacent triangles mapped from different images. They will be small if (1) the triangulated mesh is a good approximation of the true surface, and (2) the standpoints

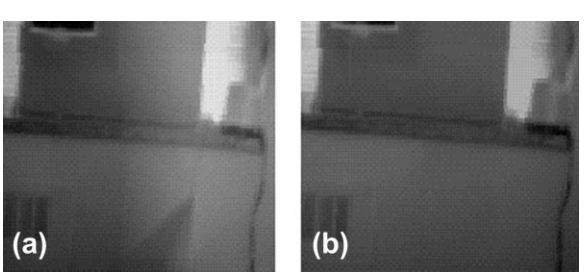

Fig. 8. Local and combined global with local adaptation: (a) only local adaptation, (b) global and local adaptation.

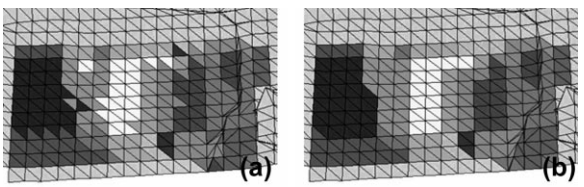

Fig. 9. Part of the triangulated mesh. Triangle shades correspond to image numbers from which the triangle texture is obtained: (a) before local re-assignment, (b) after local re-assignment.

for the two images are close together. However, even if those conditions are not satisfied, there are techniques to diminish the visual impact of the approximated mesh geometry.

One method is to locally re-assign triangle patches to images. With sufficient image overlap, the texture of a triangle can be obtained from a number of different images. A reasonable choice is to select the image in which the triangle appears largest. In a second step, this initial assignment is changed based on the image assignment of adjacent triangles and the image area covered by the triangle in alternative images. In effect, local re-assignment generates larger regions of triangles mapped from the same image and eliminates isolated triangle mappings. Thus, the number of triangle edges where adjacent triangles are mapped from different images is reduced (Figs. 9 and 10).

Another method to reduce texture discontinuities is to use texture blending. When blending is selected, the mapping algorithm does not try to find the best image for each triangle but rather computes the texture from all images the triangle appears in by forming a weighted average. While blending is an algorithmically simple approach that diminishes geometric as well as radiometric discontinuities, it must be noted that it usually introduces a detectable blurring effect (Fig. 11). It has to be decided from

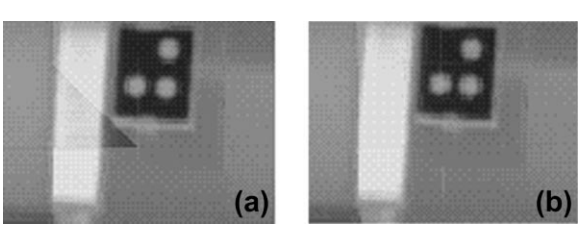

Fig. 10. Effect of locally adapting image-to-triangle selection: (a) without local re-assignment, (b) with local re-assignment.

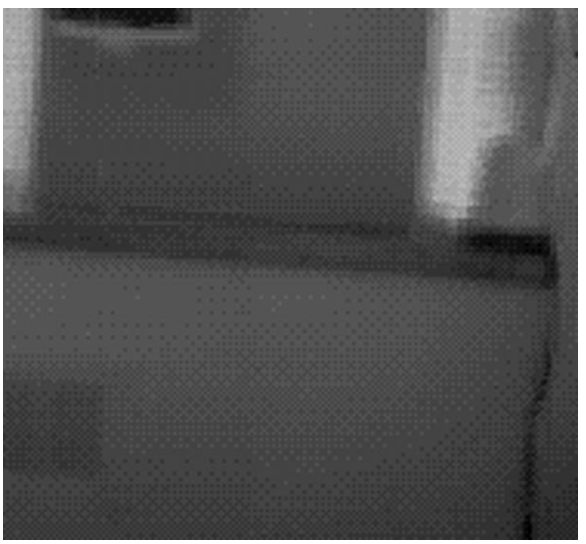

Fig. 11. Effect of blending (on the image in Fig. 7a).

case to case if global and local gray-value adaptation, which produce sharper texture maps but might show geometric artifacts at adjacent triangles, or texture blending, which reduces artifacts at adjacent triangles but tends to blur the textures, is the better choice.

7. Experimental results and discussion

Our approach to virtual environment creation has been applied to an indoor site measuring 12 m (L) $\times$ 5 m (W) $\times$ 3 m (H). The site includes a number of reference targets placed on stable surfaces where their positions are known in the global coordinate system. These targets are used to evaluate the geometric accuracy of the model. Some other natural features, such as corners, were also utilized. The site includes different types of object geometry and texture to evaluate the ability of the various sensors and the geometric and texture modeling approaches to recover as many details as possible. Some of the test results are summarized here.

In the geometric accuracy evaluation, the final error is given by $e = e_d + e_m$, where e_d is the digitized surface error, which is a combination of the range sensor errors and registration errors, and e_m is the modeling error resulting from replacing the digitized points with a triangulated mesh. The simplification process for the LOD hierarchy, which is required for real-time rendering, further increases this modeling error.

7.1. Accuracy of the digitized surfaces

The tests were performed under normal laboratory conditions (for example, no special illumination was used). First, the registration accuracy was evaluated using the difference between reference-target coordinates computed with photogrammetric bundle adjustment and their known coordinates. The RMS of the differences was 0.6 mm or 1:20,000. The second accuracy evaluation test was on the 3D coordinates obtained by the range sensor after registration. In this test, spherical targets of known positions were used. We fitted spheres to the range sensor data on the spherical targets and their centers were computed and compared to the known centers. The average difference was 2.75 mm. This is the combined error of the range sensor 3D data and the bundle adjustment registration and it represents the accuracy of the digitized surfaces before the modeling process.

7.2. Accuracy of the geometric modeling

This is the difference between the triangulated mesh model and the digitized surface data. In our modeling method, this is usually half the voxel size. In this experiment, the voxel size was set to 33 mm for the full-resolution model containing 155,494 triangles. Several models at decreasing resolution, down to 2810 triangles, or voxel size of 277 mm, were also created for the LOD hierarchy required for real-time rendering.

7.3. Texture mapping

We applied the texture mapping approach described in Section 6, first to the full-resolution geometric model of the test site, then to geometric models with decreasing number of polygons. The addition of texture allowed the reduction of the number of triangles for the full resolution model from 155,494 to 3953 without visual detection of the loss of geometric details, even though the voxel size for the simplified model was 222 mm. Fig. 12 displays views from the final model.

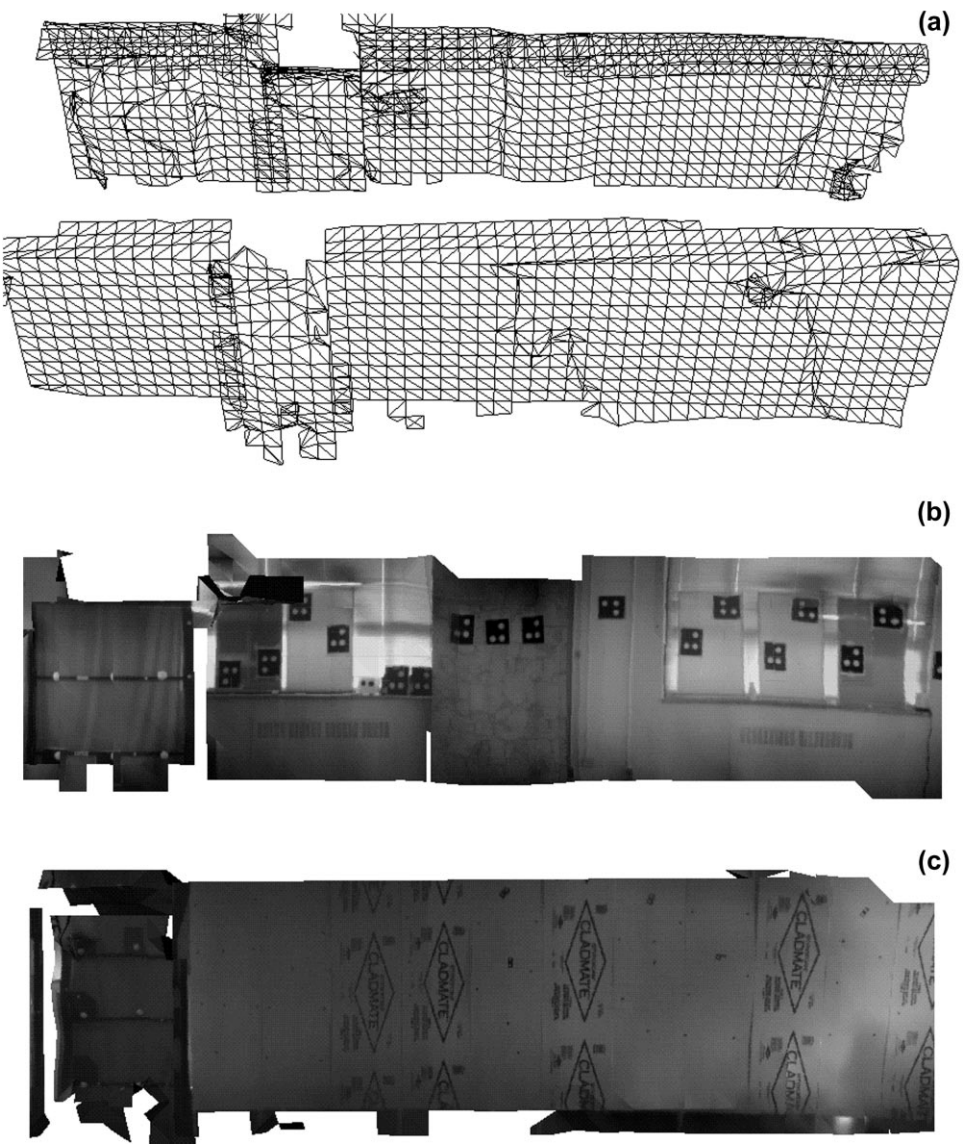

Fig. 12. The geometric model without and with texture: (a) the simplified geometric model — 3953 triangles; (b) part of side 1 with texture map; (c) part of side 2 with texture map.

Another test was to compare the use of intensity images from the standard analogue cameras, which were precisely calibrated and positioned with bundle adjustment, with the use of intensity images from the high-resolution digital still camera. The latter could only be calibrated and positioned using range data available from the range camera, as described in Section 4. Even though using the digital camera in this manner resulted in a less accurate registration, the resulting texture maps were more realistic looking.

8. Concluding remarks and future work

The creation of virtual environments from real data remains a challenging task particularly for large complex sites. It is not clear which approach, of the many that have been proposed in literature, will

work best. Several approaches have to be tried at various sites of different complexity and size before any conclusion is reached. In this paper, two claims are being made: (1) creating virtual environments for a wide range of applications requires data from multiple types of sensor; (2) modeling virtual environments requires a combination of techniques to ensure real-time rendering. We will elaborate further on each of these points.

In some applications, where only visualization and walk-through are required, it may be sufficient to create the virtual environment from intensity images alone. This may be done with image-based rendering or a hybrid image-based and model-based approach. However, for applications that require complete documentation of the environment or close interaction with its objects, data from range sensors are also required to properly cover surfaces that are either not textured or have complex geometric shapes and details.

For real-time rendering, particularly when dealing with large complex environments, the modeling approach must apply a combination of the four methods discussed in Section 1.1. For example, using the fourth method (replacing geometry with texture) will ensure that, even if the complete full-resolution model contains a large number of polygons, the model can be rendered in real time.

The results of the testing and demonstration of the system in the laboratory has shown that it is appropriate for mapping indoor environments of dimensions within the operating range of the 3D sensor. The simplified model with texture could be easily rendered in real time on today's graphics workstations and PC's with sufficient texture memory.

Several challenges, in both the data collection and modeling phases, remain and are the subject of our future work. Some of those are:

- On-site fast modeling to verify and ensure coverage of all parts of the site. The accuracy here is not important; however, the modeling process will require an automatic and fast registration procedure.
- Dealing with gaps which will inevitably exist in the geometric model of a complex environment, even when the previous item on this list is implemented.
- Development of a design, or a sensor configuration, strategy for automatic system adaptation to various environments. Since every site and application is different, the type of sensor, the parameters of each sensor, and the placement of these sensors will vary. It is desirable to have a system that automatically suggests the optimum configuration. Currently this process requires a solid background and experience in sensors, network design, image registration, and geometric modeling of various types of data.
- Automatic segmentation of complex scenes. This will assist in understanding the contents of the scene and provide a better way of model simplification. For example, a plane surface can be modeled with only corner vertices.

References

Aliaga, D., Lastra, A.A., 1997. Architectural walkthroughs using portal textures. Proc. IEEE Visualization '97, pp. 355–363.

Blais, F., Rioux, M., Domey, J., 1991. Optical range image acquisition for the navigation of a mobile robot. IEEE Conf. Robotics and Automation, Sacramento, Calif., Vol. 3, pp. 2574–2580.

Blais, F., Lecavalier, M., Bisson, J., 1996. Real-time processing and validation of optical ranging in a cluttered environment. ICSPAT, Boston, Mass., pp. 1066–1070.

Chen, S.E., Williams, L., 1993. View interpolation for image synthesis. Proc. SIGGRAPH '93, pp. 279–288.

Debevec, P.E., Taylor, C.J., Malik, J., 1996. Modeling and rendering architecture from photographs: a hybrid geometry and image-based approach. Proc. SIGGRAPH '96, pp. 11–20.

El-Hakim, S.F., Boulanger, P., Blais, F., Beraldin, J.-A., 1997. A system for indoor 3D mapping and virtual environments. SPIE Proc., Videometrics V, Vol. 3174, pp. 21–35.

Haeberli, P., Segal, M., 1993. Texture mapping as a fundamental drawing primitive. Proc. 4th Eurographics Workshop on Rendering, Paris, pp. 259–266.

Heckbert, P.S., Garland, M., 1997. Survey of polygonal surface simplification algorithms. Part of Multiresolution Surface Modeling Course #25, SIGGRAPH '97.

Hoppe, H., 1996. Progressive meshes. Proc. SIGGRAPH '96, pp. 99–108.

Johnson, A.E., Hoffman, R., Osborn, J., Hebert, M., 1997. A system for semi-automatic modeling of complex environments. Proc. Int. Conf. Recent Advances in 3-D Digital Imaging and Modeling, Ottawa, pp. 213–220.

Kang, S.B., 1997. A survey of image-based rendering techniques. Technical Report CRL 97/4, Cambridge Research Lab, Digital Equipment Corp.

Kumar, S., Manocha, D., Garrett, W., Lin, M., 1996. Hierarchical Back-Face Computation. Proc. 7th Eurographics Workshop on Rendering, pp. 231–240.

Lansdale, R.C., 1991. Texture Mapping and Resampling for

Computer Graphics. M.Sc. Thesis, Dept. of Electrical Engineering, University of Toronto.

Lorensen, W.E., Cline, H.E., 1987. Marching cubes: a high resolution 3D surface reconstruction algorithm. Computer Graphics: SIGGRAPH '87, 21 (4), July, pp. 163–169.

Luebke, D., 1997. A survey of polygonal simplification algorithms. Dept. Computer Science, University of North Carolina, Chapel Hill, Tech. Rep. TR97-045.

McMillan, L., Bishop, G., 1995. Plenoptic modeling: an image-based rendering system. Proc. SIGGRAPH '95, pp. 39–46.

Miyatsuka, Y., Chen, X., Takahashi, Y., 1998. Archaeological 3D GIS for virtual museum in Damascus. ISPRS Commission V Symposium, Real-Time Imaging and Dynamic Analysis. Int. Arch. Photogramm. Remote Sensing 32 (5), 348–351.

Ng, K., Sequeira, V., Butterfield, S., Hogg, D., Goncalves, J.G.M., 1998. An integrated multi-sensory system for photo-realistic 3D scene reconstruction. ISPRS Commission V Symposium, Real-Time Imaging and Dynamic Analysis. Int. Arch. Photogramm. Remote Sensing 32 (5), 356–363.

Roth, G., Wibowo, E., 1995. A Fast Algorithm for making mesh-models from multiple-view range data. Proc. 1995 DND/CSA Robotics and Knowledge Based Systems Workshop, pp. 349–355.

Roth, G., Wibowo, E., 1997. An Efficient volumetric method for building closed triangular meshes from 3-D image and point data. Proc. Conf. Graphics Interface 97, pp. 173–180.

Sato, Y., Wheeler, M.D., Ikeuchi, K., 1997. Object shape and reflectance modeling from observation. Proc. SIGGRAPH '97, pp. 379–388.

Sheehan, M., 1997. Technology Profile: Virtual Environments. SRI Consulting TechMonitoring Report, 65 pp.

Soucy, M., Laurendeau, D., 1996. Multiresolution surface modeling based on hierarchical triangulation. Comput. Vision Image Understanding 63 (1), 1–14.

Szeliski, R., 1996. Image mosaics for virtual environments. IEEE Comput. Graph. Appl. 16 (2), 22–30.

Teller, S.J., Sequin, C.H., 1991. Visibility preprocessing for interactive walkthroughs. Computer Graphics: SIGGRAPH '91, 25 (4), pp. 61–69.

Weinhaus, M., Devarjan, V., 1997. Texture mapping 3D models of real-world scenes. ACM Comput. Surv. 29 (4), 325–365.

Zhang, H., Manocha, D., Hudson, D., Hoff III, K.E., 1997. Visibility culling using hierarchical occlusion maps. Proc. SIGGRAPH '97, pp. 77–88.