Rapid Creation of Large-scale Photorealistic Virtual Environments

Charalambos Poullis* CGIT/IMSC, USC
Suya You† CGIT/IMSC, USC
Ulrich Neumann‡ CGIT/IMSC, USC

ABSTRACT

The rapid and efficient creation of virtual environments has become a crucial part of virtual reality applications. In particular, civil and defense applications often require and employ detailed models of operations areas for training, simulations of different scenarios, planning for natural or man-made events, monitoring, surveillance, games and films. A realistic representation of the large-scale environments is therefore imperative for the success of such applications since it increases the immersive experience of its users and helps reduce the difference between physical and virtual reality. However, the task of creating such large-scale virtual environments still remains a time-consuming and manual work.

In this work we propose a novel method for the rapid reconstruction of photorealistic large-scale virtual environments. First, a novel parameterized geometric primitive is presented for the automatic building detection, identification and reconstruction of building structures. In addition, buildings with complex roofs containing non-linear surfaces are reconstructed interactively using a non-linear primitive. Secondly, we present a rendering pipeline for the composition of photorealistic textures which unlike existing techniques it can recover missing or occluded texture information by integrating multiple information captured from different optical sensors (ground, aerial and satellite).

Index Terms: [Large-scale Modeling, Texturing, Multiple Sensory input, Photorealistic Virtual Environments]: —

1 INTRODUCTION

Virtual reality technologies are becoming increasingly popular and are being widely used in a range of different applications. In particular, civil and defense applications employ such technologies for the simulation of real world operations areas. In such cases the models of urban buildings are of significant value since they facilitate planning, response, and real-time situational awareness in highly-occluded urban settings [7, 10, 17]. The personnel that simulate, plan, monitor, and execute responses to natural or man-made events can gain insight and make better decisions if they have a comprehensive view of the structures and activity occurring at an operational scene. The models are essential components of such a view, helping people comprehend spatial and temporal relationships. In addition, a photorealistic appearance is essential for the enhancement of the visual richness of the models and the immersive experience of the users [16, 6].

While models are important assets, the creation of photorealistic large-scale models remains at best a difficult, time-consuming manual task. The science for rapidly sensing and modeling wide-area urban sites and events, in particular, pose difficult or unsolved problems. Over the years, a wealth of research, employing a variety of sensing and modeling technologies, has been conducted to deal with the complex modeling problem. Different types of techniques, ranging from computer vision, computer graphics, photogrammetry, and remote sensing, have been proposed and developed so far, each has unique strengths and weaknesses, and each performs well for a particular dataset but may fail under another.

Similarly, the generation of high-resolution photorealistic textures has been extensively investigated and many algorithms have been already proposed. However, the complexity of these algorithms increases considerably when dealing with large-scale virtual environments. In addition, these texturing techniques for large-scale environments are limited to using a single image per building which makes it impossible to recover textures for missing or occluded areas and requires even more time-consuming manual work. These problems impose a serious limitation in achieving a realistic appearance for the models and in extent diminishes the immersive experience of the users.

In this work we address the problem of rapid creation of photorealistic large-scale virtual environments. Firstly, we address the 3D model reconstruction problem and propose a novel parameterized geometric primitive for the automatic detection, identification and reconstruction of building models. We leverage the symmetry constraints found in man-made structures to reduce the number of unknown parameters needed during the model reconstruction, therefore considerably reducing the computational time required. In addition, complex buildings containing non-linear surfaces such as domes, stadiums, etc. are interactively reconstructed using a non-linear primitive.

Secondly, we address the problem of texturing, and propose a rendering pipeline for the composition of photorealistic textures. A significant advantage of this pipeline is that textures for missing or occluded areas in one image can be recovered from another image therefore eliminating the need for any manual editing work. Images captured from multiple sensors (ground, aerial, satellite) are integrated together to produce a set of view-independent, seamless textures.

We have extensively tested the proposed approach with a wide range of sensing data including satellite, aerial, ground photographs and LiDAR(Light Detection And Ranging) and present our results.

The paper is organized as follows. In the next section, we discuss previous work related to modeling and texturing. Section 3 provides an overview of the system. Section 4 describes the two main components for the reconstruction of large-scale models from LiDAR, namely the data pre-processing in section 4.1, and the modeling in section 4.2. In particular, section 4.2.2 introduces the novel parameterized geometric primitive used to automatically identify and reconstruct buildings with the most commonly occurring linear roof-types. Section 5 describes the texturing pipeline employed for the generation of photorealistic textures. The two main components of the texturing pipeline are the registration of imagery from various optical sensors described in section 5.1, and the texture composition described in section 5.2.

2 RELATED WORK

2.1 Modeling

A good survey on large-scale modeling techniques can be found in [5]. In [4] the authors present a method for reconstructing large-scale 3D city models by merging ground-based and airborne-based

*e-mail: charalambos@poullis.org
†e-mail:suyay@graphics.usc.edu
‡e-mail:uneumann@graphics.usc.edu

IEEE Virtual Reality 2008
8-12 March, Reno, Nevada, USA
978-1-4244-1971-5/08/$25.00 ©2008 IEEE

LiDAR data. The elevation measurements are used to recover the geometry of the roofs. Facade details are then incorporated by the high resolution capture of a ground based system which has the advantage of also capturing texture information. The textures aid in the creation of a realistic appearance of the model. However, at the cost of having detailed facades they neglect to deal with the complexities and wide variations of the buildings' roof types. The same authors later extended their method to incorporate texture information from oblique aerial images. Although they combine multiple aerial images to determine the model's textures, their method is restricted to traditional texture mapping rather than combining all available texture information to generate a composite texture i.e. blending. Therefore, a significant color difference between images will cause visible and non-smooth transitions between neighbouring polygons of different texture images.

In [20] You et al, present an interactive primitive-based modeling system for the reconstruction of building models from LiDAR data. Using the user input, their system automatically segments the building boundary, performs model refinement, and assembles the complete building model. However, user input is required for the detection as well as the identification of buildings and their rooftypes.

In [15] they develop an interactive system for reconstructing geometry using non sequential views from uncalibrated cameras. The calculation of all 3D points and camera positions is performed simultaneously as a solution of a set of linear equations by exploiting the strong constraints obtained by modeling a map as a single affine view. However, due to the considerable user interaction required by the system, its application to large-scale areas is very limited.

In [13] the proposed system can deal with uncalibrated image sequences acquired with a hand-held camera. Based on tracked or matched features the relations between multiple views are computed. From this both the structure of the scene and the motion of the camera are retrieved. Although the reconstructed 3D models are visually impressive they consist of complex geometry -as opposed to simple polygonal models- which requires further processing and limits their applications.

In [12] Nevatia et al, propose a user-assisted system for the extraction of 3D polygonal models of buildings from aerial images. Low level image features are initially used to build high level descriptions of the objects. Using a hypothesize and verify paradigm they are able to extract impressive models from a small set of aerial images. The authors later extended their work in [8] to automatically estimate camera pose parameters from two or three vanishing points and three 3D to 2D correspondences.

In [2] a ground-based LiDAR scanner is used to record a rather complex ancient structure of significant cultural heritage importance. Multiple scans were aligned and merged together using a semi-automatic process and a complete 3D model was created of the outdoor structure. The reconstructed model is shown to contain high-level of details however the complexity of the geometry (90 million polygons for one building) limits this approach to the reconstruction of single buildings rather than large-scale.

In a different approach, [3] proposed an interactive system which can reconstruct buildings using ground imagery and a minimal set of geometric primitives. More recently [14] extended this system to incorporate pointcloud support as part of the reconstruction however the required user interaction increases considerably for large-scale areas. Moreover, the user interaction depends on the desired level of detail of the reconstructed models which may vary considerably according to the application.

2.2 Texturing

One of the most popular and successful techniques in this area is the one introduced by [3] which uses a small set of images to reconstruct a 3D model of the scene. View-dependent texture mapping (VDTM) is then performed for the computation of the texture maps of the model. By interpolating the pixel color information from different images new renderings of the scene can be produced. The contributions of each image to a pixel's color is weighted based on the angle difference between the camera's direction and the novel view-point's direction. The authors then extended their work and showed how VDTM can be efficiently implemented using projective texture mapping, a feature available in most computer graphics hardware. Although this technique is sufficient to create realistic renderings of the scene from novel view-points its computation is still too expensive for real-time applications, like games or virtual reality.

In [11] they order geometry into optimized visibility layers for each photograph. The layers are subsequently used to create standard 2D image-editing layers which become the input to a layered projective texture rendering algorithm. However, this approach chooses the best image for each surface rather than combining the contributions of all the images to minimize information loss.

In [2] reflectance properties of the scene were measured and lighting conditions were recorded for each image taken. An inverse global illumination technique was then used to determine the true colors of the model's surfaces which could then be used to relight the scene. The generated textures are photorealistic, however for large-scale areas it requires considerable amount of manual work for the capturing and the processing of the data.

A method for creating renders from novel view-points without the use of geometric information is presented in [9], where densely regularly sampled images are blended together. The image capture for small objects is very easily achieved, however this task is close to impossible to perform for large-scale areas.

A slightly different approach for texture generation and extraction is proposed in [18]. Given a texture sample in the form of an image, they create a similar texture over an irregular mesh hierarchy that has been placed on a given surface, however this approach cannot capture the "actual" appearance of the models.

3 System Overview

The system's overview is summarized in Figure 1. Firstly, the airborne LiDAR data is preprocessed as described in section 4.1. This step involves resampling (section 4.1.1) the data into a regular grid structure, filtering i.e. hole filling and smoothing (section 4.1.2), and segmenting the points into vegetation or ground (section 4.1.3). The preprocessing plays an essential role in the success of modeling since it reduces the noise and inconsistencies from the data while ensuring that important features such as discontinuities are preserved.

Secondly, the roof boundaries are extracted from the preprocessed data (section 4.2.1) and 3D models are generated based on the automatic identification of the roof-types (section 4.2.2). A major advantage of this approach is that it can automatically identify the roof-types which can be a very difficult task even for a human operator. In addition, an interactive non-linear primitive is employed for the reconstruction of complex non-linear roof-types (section 4.2.3).

Finally, the camera poses for the ground, aerial and satellite images are calculated from registered correspondences to the reconstructed 3D models (section 5.1), and are used by the rendering pipeline to generate the composite photorealistic textures (section 5.2). A key aspect of this pipeline is the integration of multiple information for the efficient and effective handling of textures for missing or occluded areas.

4 Large-scale Model Reconstruction

4.1 Pointcloud Data Pre-processing

The preprocessing of the raw 3D pointcloud data is performed in three steps: resampling, filtering and segmentation. The result is a

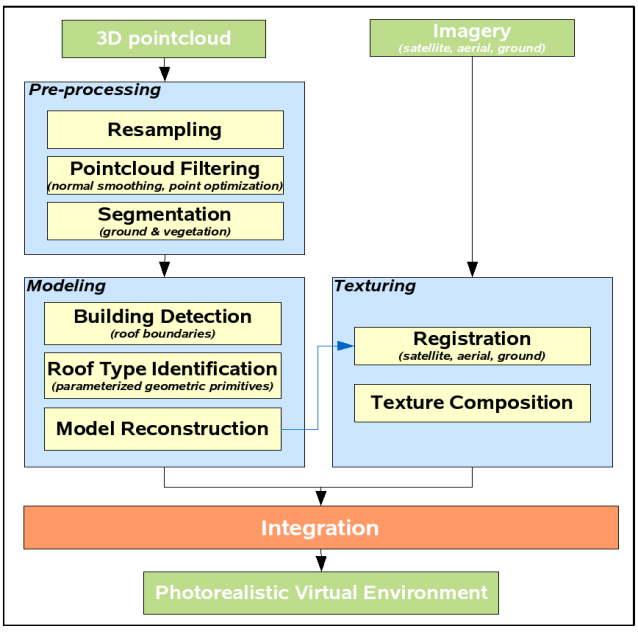
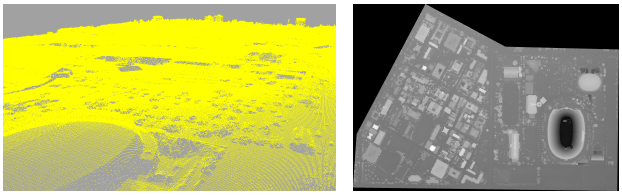


Figure 1: System overview. Three main components: preprocessing, modeling and texturing.

classification of the refined points into ground and vegetation.

4.1.1 Resampling

The unstructured 3D pointcloud (Figure 2(a)) is initially resampled into a 2D regular grid (map shown in Figure 2(b)). To overcome the problem of information loss of the samples stored in the grid, the dimensions of the grid are determined based on the sampling rate which is automatically calculated either by specifying a desired resolution or by specifying a maximum allowed error tolerance between the samples. For the results shown in this paper we have used a maximum allowed error tolerance $\tau = 0.00001$.



(a) (b)

Figure 2: Resampling of the 3D pointcloud data into a regular grid. (a) Unstructured 3D pointcloud. (b) 2D Regular grid. The image coordinates represent the X,Y of each point and the intensity represents the elevation Z.

4.1.2 Filtering

Noise in the measurements of the original data and inconsistencies introduced by the resampling process are removed while ensuring that important features such as discontinuities are preserved. To achieve this, we perform a normal optimization based on graph-cuts to smooth the normals, followed by a point optimization using gradient-descent to smooth the points.

A set of uniformly distributed normals (Figure 3(c)) computed from a half-sphere (Figure 3(a),3(b)) are used to re-label the noisy normals computed from the original points ([19]). This results in

(a) (b) (c) .

Figure 3: Computation of smooth normal labels from a half-sphere. (a) 2D point map of a half-sphere. (b) 3D (top)-view of (a). (c) Smooth normal labels computed from (a).

smoothing the normals while preserving important features such as discontinuities [1]. A gradient-descent optimization then refines the original points such that their computed normals are identical to the smooth re-labeled normals.

For example, the initial normals in Figure 4(b) are computed from the original points in Figure 4(a). A graph-cut optimization performed using these initial normals and the smooth normal labels from Figure 3(c) results in the re-labeled normals shown in Figure 4(c). Finally, a gradient-descent optimization iteratively refines the original points such that the computed normal at each point matches the re-labeled normal (Figure 4(d)).

A closeup of a mesh produced by triangulating the refined points is shown in Figure 4(h),4(i). Different number of smooth normal labels $\|L\|$ (computed by varying the radius of the half-sphere) and different values for the smoothness weight λ used by graph-cuts, are shown. The results shown in this paper were generated with $\|L\| = 104, \lambda = 0.5$.

4.1.3 Segmentation

Buildings are man-made structures with the characteristic of having roofs with uniform elevation and uniform orientation. On the other hand, the elevation and orientation of trees and other vegetation rapidly changes between neighbouring points. Thus, the goal of the segmentation is to determine neighbouring points with elevation and orientation difference less than a predefined threshold ($\Delta_z = 0.01, \Delta_\theta = 15^\circ$) and to group them into regions representing building candidates using a region-growing algorithm. In addition, regions consisting of a number of points less than a minimum are discarded ($\tau_{area} < 50px$).

4.2 Modeling

The modeling of the buildings from the segmented candidate points is automatically performed in three steps: building detection, roof-type identification and model reconstruction.

4.2.1 Building Detection

One of the main limitations of airborne LiDAR scanners is the accurate measurements near discontinuities due to the rapid change in elevation which often leads to boundaries with a zig-zag shape. To overcome this problem, the convex hulls enclosing the points of each region detected using a region-growing algorithm, are linearized using Douglas-Peucker polygonal approximation ($\Delta_\varepsilon = 1px$). For example, Figure 5(a) shows the detected points after the application of a region-growing algorithm using an elevation threshold $\Delta_z = 0.01$ and an orientation threshold $\Delta_\theta = 10^\circ$. The starting point for the region growing is initialized as the point with the highest elevation which is indicated with a red color. Figure 5(b) shows the resulting approximated convex hull of the detected points in Figure 5(a). The length of the boundaries are used to determine the width W and height H of the building structure. Finally, a series of subtractive polygonal boolean operations are applied to the regions' boundaries to resolve any inconsistencies produced by

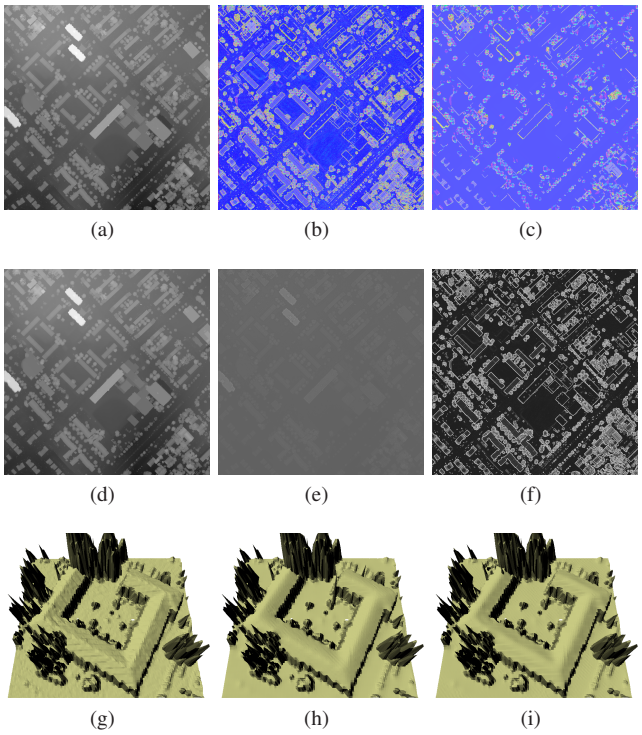


Figure 4: The filtering is performed in two steps: normal-based optimization using graph-cuts for smoothing the normals, and point-based gradient-descent optimization for smoothing the points using the smooth normals: (a) Initial noisy points $P_{initial}$. (b) Initial noisy normals $N_{initial}$. (c) Smooth normals N_{smooth}. (d) Smooth points P_{smooth}. (e) Point difference ($|P_{initial} - P_{smooth}|$). (f) Normal difference ($1 - |N_{initial} . N_{smooth}|$). Graph-cut optimization control parameters: (g) Original data. (h) $\|L\| = 104, \lambda = 0.5$. (i) $\|L\| = 492, \lambda = 1.0$.

overlaps due to structures on the roofs such as chimneys, elevator engines, air-conditioners, etc.

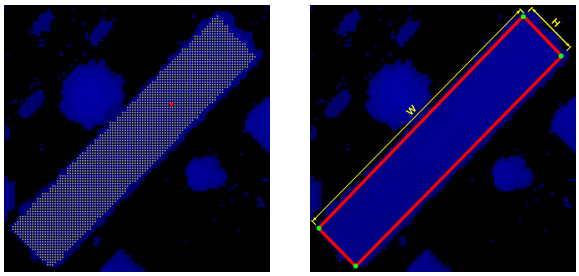


Figure 5: Building detection. (a) Detected roof points using region-growing. The red point indicates the starting point i.e. with highest elevation. (b) Resulting building boundaries. (Note: the radius of the red point and the width of the red lines were manually increased for clarity)

4.2.2 Linear Roof-type Identification and Reconstruction

The identification of the roof-types is an essential part of the modeling process. In many cases especially for complex buildings and low-resolution data, this is still a very difficult task even for a human operator. We address this problem and present a novel parameterized geometric primitive for the automatic identification of the most commonly occurring roof-types.

We leverage the symmetry constraints found in man-made structures and parameterize a geometric primitive using only two variables. Figure 6(a) shows the geometric structure of the parameterized primitive. By specifying a local coordinate system the control points (external points V_{0-3} and internal points P_{0-3}) are expressed as functions of two variables α, β such as,

$$V_0 = [-\frac{W}{2}, -\frac{H}{2}], \qquad V_1 = [\frac{W}{2}, -\frac{H}{2}] \tag{1}$$

$$V_2 = [\frac{W}{2}, \frac{H}{2}], \qquad V_3 = [-\frac{W}{2}, \frac{H}{2}] \tag{2}$$

$$P_0 = [-\frac{\alpha W}{2}, -\frac{\beta H}{2}], \qquad P_1 = [\frac{\alpha W}{2}, -\frac{\beta H}{2}] \tag{3}$$

$$P_2 = [\frac{\alpha W}{2}, \frac{\beta H}{2}], \qquad P_3 = [-\frac{\alpha W}{2}, \frac{\beta H}{2}] \tag{4}$$

where W is the width, and H is the height of the primitive which are determined by the length of the detected boundaries as explained in section 4.2.1. This parameterization allows for the symmetry constraints to be enforced since a change of a single point will affect all other points. In order to ensure that all the internal points P_{0-3} are always within the exterior points V_{0-3} the parameters are bound in the range $0 \leq \alpha, \beta \leq 1$. By varying the values of the two parameters α, β all commonly occurring roof-types can be produced by a *single* primitive as demonstrated in Figure 6(b). This is a major advantage since it significantly reduces the number of primitives required to model a scene and allows a *single* primitive to be used for the automatic identification of *multiple* roof-types. Another significant advantage is that the optimization involves only two unknowns which are bound-constraint due to the condition that $0 \leq \alpha, \beta \leq 1$, thus making the convergence to a solution extremely fast.

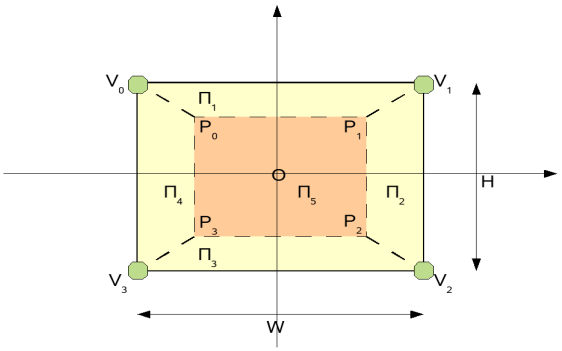
A quasi-Newton minimization for bound-constrained problems, is then performed to find the optimal values of the two variables α, β such that E_{error} in equation 5 i.e. the sum of the least-square fitting error of the five planar surfaces, is minimum,

$$E_{error} = \sum_{n=0}^{N} \sum_{\forall p \in P} (\psi_p - f(\chi_p, \alpha, \beta))^2 \tag{5}$$

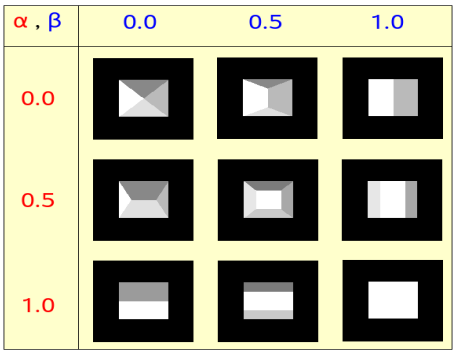
where N is the number of planes (5), and $f(...)$ is a fitting function.

Degenerate cases can occur when any of the interior points P_{0-3} collapse on the same point. For example when $\alpha = \beta = 0$ all the interior points P_{0-3} collapse on the same point, therefore producing a degenerate plane Π_5 (Figure 6(a)). Since fitting a plane requires a minimum of 3 distinct points, we specifically handle degenerate cases and return a plane fitting error of 0. In this example, the plane fitting error for the degenerate plane Π_5 will be 0 and thus the total error E_{error} will be equal to the plane fitting error of the planes Π_{1-4}. Our experiments have shown that even in degenerate cases such as this, the minimization correctly converges to the optimal values for the parameters α, β. A special case occurs with flat (or flat-sloped) roofs since any value for α, β is an optimal value. In this case, the reconstructed model may consist of multiple co-planar surfaces.

During this minimization, a gaussian mixture model (GMM) is used to model the elevation distribution of all points $p \in P$ inside the planar surface boundaries. A GMM consists of a superposition of gaussian densities which can better separate the outlier points produced by objects such as elevator engines, air-conditioners and chimneys commonly found on the roofs, thus removing the otherwise significant bias of those outliers from the distribution of the true surface points lying on the roof. An expectation maximization algorithm fits a GMM of 3 components to the elevation samples and

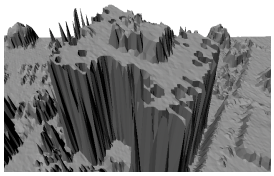


(a)

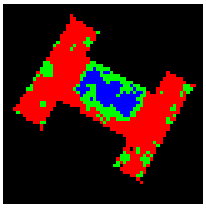


(b)

Figure 6: The parameterized geometric primitive. (a) Parameterized primitive. (b) The different roof types generated by varying the parameters α, β.



(a)



(b)

GMM comp.	Mixing coeff.(κ)	Mean(μ)	Variance(σ^2)
G_0	0.684208	0.526408	0.0000000479
G_1	0.225278	0.538507	0.0000405911
G_2	0.0905144	0.493301	0.000859696

(c)

Figure 7: Example classification of points using a gaussian mixture model consisting of 3 components. (a) Roof cluttered with objects. (b) Point classification based on the gaussian component which maximizes the probability of each point (G0-red, G1-green, G2-blue). (c) The values of the GMM for 7(a).

determines the mixing coefficient κ_i, the mean μ_i and the variance σ_i^2 of each gaussian density G_i where $i = 0, 1, 2$.

Figure 7(a) shows a building with several objects of different elevation located on the roof and the parameters for the GMM components are shown in Table 7(c). We define the dominant component of the GMM, G_{dom}, to be the component with the highest mixture coefficient κ_{dom} in the GMM e.g. G_0 in Table 7(c). The dominant component G_{dom} is then used to classify all the points $p \in P$ whose propability is maximized in G_{dom} (equation 6),

$$P_{inliers} = \forall p \in P \forall i \in \{0,1,2\}(Pr(p|G_{dom}) \geq Pr(p|G_i)) \quad (6)$$

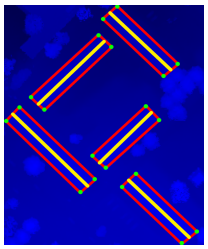
, as the true surface points $P_{inliers} \in P$, and everything else as outliers $P_{outliers} = P - P_{inliers}$. Figure 7(b) shows the points inside the detected boundaries being color-coded based on the distribution that maximizes their probability. Red points have the highest probability in G_0-which is the most dominant component G_{dom}-, green points in G_1 and blue points in G_2. Using only the inlier points equation 5 now becomes,

$$E_{error} = \sum_{n=0}^{5} \sum_{\forall p \in P_{inliers}} (\psi_p - f(\chi_p, \alpha, \beta))^2 \quad (7)$$

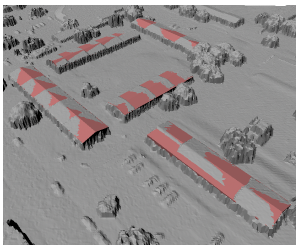
and the fitting is performed only on the points $P_{inliers}$ corresponding to the dominant component i.e. red points.

The classification using the GMMs has the effect of making the plane fitting more robust and less susceptive to the presence of outliers. This greatly improves the performance of the optimization as well as the accuracy of the reconstructed surfaces.

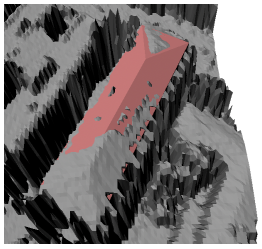
An example is shown in Figure 8(a) where the detected boundaries are shown in red and the automatically recovered interior boundaries produced by the optimization are shown in yellow. Note that even for a human operator it is hard to identify the roof-type of these buildings from Figure 8(a). The reconstructed buildings are shown in Figure 8(b), overlaid to the original data. Figure 8(c),8(d) shows different roof-types successfully identified by the primitive.



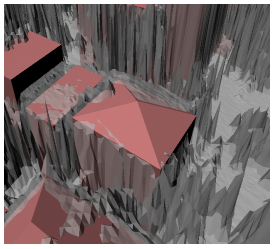
(a)



(b)



(c)



(d)

Figure 8: Automatic roof-type identification using the parameterized geometric primitive. (a) Detected boundaries (red lines) and recovered interior boundaries (yellow lines). (b) Reconstructed buildings from (a) overlaid on original data (Gabled roof-types). (c) Hipped roof-types. (d) Pyramidal roof-types.

4.2.3 Non-linear Roof-type Reconstruction

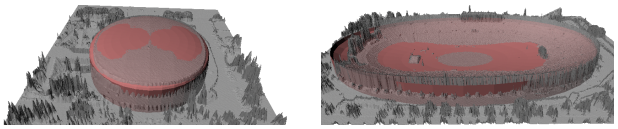
The reconstruction of 3D models is performed based on the automatically identified roof-types as previously explained to produce a water-tight, light-weight polygonal geometric model of the scene. Despite the fact that the parameterized primitive can model most of the buildings found in an urban area, it is limited to handling linear surfaces and cannot be used for the identification of complex

buildings containing non-linear surfaces. Therefore, an interactive approach is employed for the identification and reconstruction of such complex buildings.

An ellipsoidal primitive is designed to handle all types of non-linear surfaces either dome-like or stadium-like (hollow) by fitting an ellipsoid to the data. The primitive is initialized with the centroid and two perpendicular points on the surface's perimeter used to determine α and β in equation 8. A non-linear optimization is then performed to recover the optimal value for the single unknown variable γ such that the equation 8 is minimized.

$$\frac{x^2}{\alpha^2}+\frac{y^2}{\beta^2}+\frac{z^2}{\gamma^2}=0 \tag{8}$$

Figure 9(a) shows an arena with a dome-like non-linear roof reconstructed using this primitive. Similarly, Figure 9(b) shows a stadium-like structure being overlaid on the original data. The system automatically determines what type of surface is being reconstructed and automatically chooses the orientation of the surface type i.e. dome- or stadium-like.



(a) (b)

Figure 9: Reconstruction of non-linear structures. (a) Dome-like structure. (b) Stadium-like structure.

5 Photorealistic Texture Generation

The generation of photorealistic textures is performed in two steps: registration of imagery to the reconstructed model and texture composition.

5.1 Registration

The goal of the registration is to recover the camera projection matrix which is expressed in terms of the intrinsic and extrinsic parameters of the camera,

$$C=\underbrace{\begin{bmatrix}\alpha & -\alpha cot(\theta) & u_0\\ 0 & \frac{\beta}{sin(\theta)} & v_0\\ 0 & 0 & 1\end{bmatrix}}_{intrinsic}\underbrace{\begin{bmatrix}r_{11} & r_{12} & r_{13} & t_1\\ r_{21} & r_{22} & r_{23} & t_2\\ r_{31} & r_{32} & r_{33} & t_3\end{bmatrix}}_{extrinsic} \tag{9}$$

where $\alpha = kf_x$, $\beta = kf_y$, (f_x, f_y) is the focal length on the x and y axis respectively, θ is the skew angle, u_0, v_0 is the principal point on the x and y axis respectively and r_{1-3}, t_{1-3} determine the camera's rotation and translation relative to the world.

Various methods have already been proposed for the estimation of these parameters. However, when dealing with images from multiple sensors it is often required to use various methods for the recovery of the matrix C depending on the type and characteristics of the images.

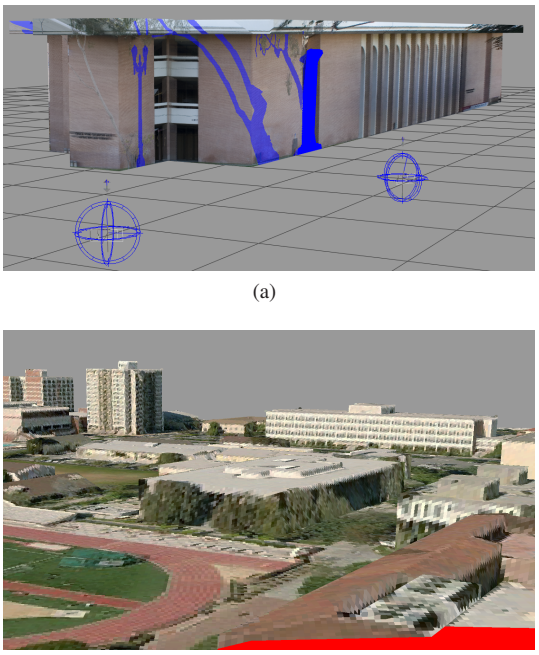
5.1.1 Registration of Ground Images

The camera model of equation 9 is used to recover the camera pose for ground images. A minimum of six correspondences between the images and the reconstructed model are required and the camera parameters are calculated using a linear least-squares method. In cases where there are not enough correspondences between the image and the model, a non-linear optimization is used to recover the parameters. Alternatively, vanishing points are used for estimating the parameters as in [15]. A bundle adjustment optimization finally refines the camera parameters thus improving the overall accuracy of the alignment. Figure 10(a) shows ground images being aligned to a reconstructed model of a building. The blue marks are areas of unwanted textures (trees and light-post) which are discussed later.

5.1.2 Registration of Satellite and Aerial Images

Aerial images are taken from oblique angles and therefore the camera pose can be recovered with the same process explained previously for ground images. In cases where an initial estimate of the camera pose is available, either through manual interaction or GPS coordinates, we use a non-linear optimization to determine the parameters of equation 9. Figure 10(b) shows the camera pose recovery of an aerial image. The image is then projected on the reconstructed models for visual verification.

Satellite images on the other hand have weak-perspective and are often produced by stitching together several perspective images. The camera model of equation 9 cannot be used to deal with multi-perspective images and will most definitely fail. However, this type of images can be rectified by using vanishing points to recover the extrinsic parameters of the camera (rotation and translation) [20]. Additional correspondences can then be used to compute a homography between the rectified image and the model.



(a)

(b)

Figure 10: Registration of ground, aerial and satellite. (a) Registration of 2 ground images. The blue areas indicate occluded areas which will be recovered by the texture composition. (b) Camera pose recovery of aerial image. Projective texture mapping is used to project the image on the reconstructed models for visual verification.

5.2 Texture Composition

The texture composition is performed by integrating the appearance information of all registered images to create view-independent, seamless textures. A key aspect of this integration is the efficient and effective recovery of texture information for occluded and missing areas. The generation of the composite textures involves three steps: scene pre-processing, computation of the texture map resolution and rendering and packing.

5.2.1 Scene Pre-processing

A view-dependent based subdivision is first applied to the model to ensure that all surfaces are *entirely* visible in all images. For each image, a visibility check is performed and the following actions are taken:

- a surface which is entirely visible in the image remains unchanged.
- a partially visible surface is clipped at the boundary of the projection of the image plane, and
- a non-visible (or backfacing) surface remains unchanged.

Figure 11(a) shows an example of the visibility clipping. The cube is clipped at the projected boundary of the first image plane and similarly, the cylinder is clipped at the projected boundary of the third image. This ensures that all surfaces are entirely visible in all images which is imperative for the correct computation of the texture map resolutions.

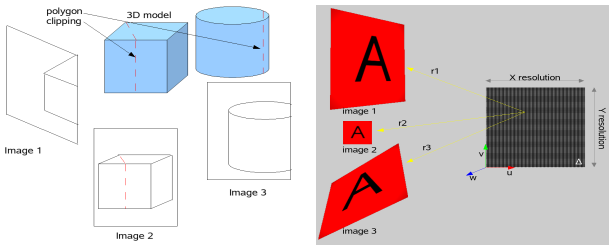


Figure 11: Visibility clipping and texture map resolution computation. (a) Visibility-based clipping. (b) Texture map resolution.

5.2.2 Texture Map Resolution

A surface may appear in multiple images with different resolutions. To ensure minimal information loss and sharp textures, the resolution of a texture map is chosen to be the size of the projected polygon with the largest area in image space. Figure 11(b) shows each surface (red polygons) projected into three image planes. The dimensions of the polygon with the largest area are chosen as the resolution of the texture map for the surface. Surfaces which are backfacing or are not visible in any of the images are automatically assigned a default color.

5.2.3 Rendering and Texture Packing

The composite textures are then rendered using ray-tracing. For each point on the surface, a ray is cast to the image planes. A point on the surface, is visible in an image if there is no other surface intersecting the ray casted from the point to the image plane. If the point is visible in an image then the corresponding pixel color is retrieved and is weighted based on the following criteria:

1. the distance of the pixel to the image boundary. Pixels which are close to the edges receive a lower weight than pixels located in the middle. This is required in order to hide any stitching effects between images, and reduce the artifacts introduced when blending multiple images. In our examples the weight of this criterion is $w_{d1} = 0.3$.
2. the angle between the camera's direction and the surface's normal. Images taken from oblique angles are down-weighted, since they have a higher degree of perspective distortion. In our examples the weight of this criterion is $w_\theta = 0.3$.
3. the distance of the pixel from the principal point. Pixels which are further away from the principal point exhibit higher radial distortion. Although the radial distortion coefficients can be closely approximated and used to undistort the image, we have found that it is more likely to have misalignments between images at points further away from the principal point. In our examples the weight of this criterion is $w_{d2} = 0.2$.
4. the resolution of the image. High-resolution images capture higher-level of detail than low-resolution images thus they are preferred. In our examples the weight of this criterion is $w_r = 0.2$.

The composite texture maps are finally sorted based on their resolution and are packed into a texture atlas. The process transforms the texture space of each map therefore requiring the recalculation of the texture coordinates for the models in the scene.

As previously mentioned, the rendering pipeline has the significant advantage that multiple information is integrated for the generation of the textures thus allowing the efficient and correct handling of occluded and missing areas. An example was shown Figure 10(a) where the occluded areas were indicated with a blue color. The composite textures produced for that building are shown in Figure 12. The blue masked areas which indicated occluded areas due to the presence of a tree and a light-post are successfully recovered.

Figure 12: Occluded areas and areas which are not visible by any camera and therefore no texture information is available appear as black.

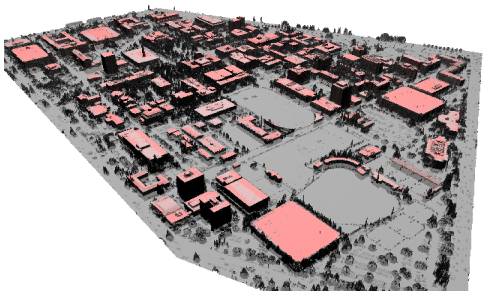
Another example of a textured building is shown in Figure 13(a). The composite textures were generated using a set of 3 ground images. In this case building details such as the window intrusions are not modeled therefore causing perspective distortion effects as shown in Figure 13(b). These effects become even more dramatic as the angle between the camera viewing direction and surface orientation increases. This is the main reason for the use of the second criterion during the weight computation.

Figure 13: Composite textures generated using 3 high-resolution ground images. (a) Novel view point render. (b) Perspective distortion effects due to unmodeled geometry.

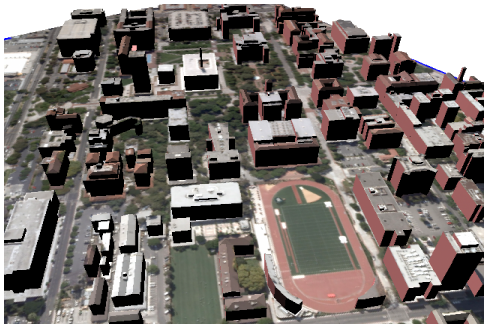
6 Results

Figure 14 shows a virtual environment created with the proposed approach. The models in Figure 14(a) are shown overlaid on the

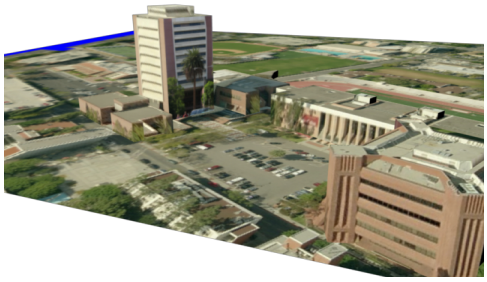
original LiDAR data. Figure 14(b) shows the model textured with high-resolution (4K) satellite imagery and Figure 14(c) shows an area in the scene with textures generated from satellite, aerial and ground imagery.



(a)



(b)



(c)

Figure 14: Reconstructed virtual environment. (a) Reconstructed models overlaid on original data. (b) Area textured with high-resolution (4K) satellite imagery. (c) Area textured with satellite, aerial and ground imagery.

7 Conclusion

We have presented a novel and complete approach for the rapid creation of photorealistic large-scale virtual environments.

Firstly, we resolved the 3D model reconstruction problem using a novel parameterized geometric primitive for the automatic detection, identification and reconstruction of building models. This primitive significantly reduces the number of user interaction and increases the computational speed of the reconstruction process by exploiting common symmetry constraints found in man-made structures. In addition, complex buildings containing non-linear surfaces such as domes, stadiums, etc. are interactively reconstructed using a non-linear primitive.

Secondly, we resolved the problem of texturing and presented a rendering pipeline for the composition of photorealistic textures which allows for the effective handling of missing or occluded areas. The result is a set of view-independent, seamless textures which are composited from images captured from multiple sensors (ground, aerial, satellite).

Finally, we have presented results which validate our approach.

References

[1] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. In *ICCV*, pages 377–384, 1999.

[2] P. Debevec, C. Tchou, A. Gardner, T. Hawkins, C. Poullis, J. Stumpfel, A. Jones, N. Yun, P. Einarsson, T. Lundgren, M. Fajardo, and P. Martinez. Estimating surface reflectance properties of a complex scene under captured natural illumination. Technical report, USC, ICT, 2004.

[3] P. E. Debevec, C. J. Taylor, and J. Malik. Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach. In *Proc. of the ACM Conf. on Computer Graphics*, pages 11–20, New York, Aug. 4–9 1996. ACM.

[4] C. Früh and A. Zakhor. Constructing 3D city models by merging ground-based and airborne views. In *CVPR*, pages 562–569. IEEE Computer Society, 2003.

[5] J. Hu, S. You, and U. Neumann. Approaches to large-scale urban modeling. *IEEE CGA*, 23(6):62–69, 2003.

[6] C.-R. Huang, C.-S. Chen, and P.-C. Chung. Tangible photorealistic virtual museum. *IEEE CGA*, 25(1):15–17, 2005.

[7] U. Laptaned. Situation awareness in virtual environments: A theoretical model and investigation with different interface designs. In *Computers and Advanced Technology in Education*, page 528, 2006.

[8] S. C. Lee, S. K. Jung, and R. Nevatia. Automatic pose estimation of complex 3D building models. In *WACV*, pages 148–152. IEEE Computer Society, 2002.

[9] M. Levoy and P. Hanrahan. Light Field Rendering. In *Computer Graphics Proceedings, Annual Conference Series, 1996 (ACM SIGGRAPH '96 Proceedings)*, pages 31–42, 1996.

[10] P. H. Martin Reznek and T. Krummel. Virtual reality and simulation: Training the future emergency physician. In *Acad Emerg Med Volume 9, Issue 1,*, pages 78–87, 2002.

[11] A. R. Martinez and G. Drettakis. View-dependent layered projective texture maps. In *Pacific Conference on Computer Graphics and Applications*, pages 492–496. IEEE Computer Society, 2003.

[12] R. Nevatia and K. E. Price. Automatic and interactive modeling of buildings in urban environments from aerial images. In *ICIP (3)*, pages 525–528, 2002.

[13] M. Pollefeys, L. J. V. Gool, M. Vergauwen, F. Verbiest, K. Cornelis, J. Tops, and R. Koch. Visual modeling with a hand-held camera. *International Journal of Computer Vision*, 59(3):207–232, 2004.

[14] C. Poullis, A. Gardner, and P. Debevec. Photogrammetric modeling and image-based rendering for rapid virtual environment creation, Proc. of Army Science Conf. 2004.

[15] D. P. Robertson and R. Cipolla. Building architectural models from many views using map constraints. *Lecture Notes in Computer Science*, 2351:155–163, 2002.

[16] M. Roussou and G. Drettakis. Photorealism and non-photorealism in virtual heritage representation. In D. Arnold, A. Chalmers, and F. Niccolucci, editors, *Int'l Symp. on Virtual Reality, Archaeology and Intelligent Cultural Heritage*, pages 51–60, Brighton, United Kingdom, 2003. Eurographics Association.

[17] D. M. Simpson. Virtual reality and urban simulation in planning: A literature review and topical bibliography. In *Journal of Planning Literature, Vol. 15, No. 3,*, pages 359–376, 2001.

[18] G. Turk. Texture synthesis on surfaces. In *SIGGRAPH*, pages 347–354, 2001.

[19] T. P. Wu and C. K. Tang. Dense photometric stereo using a mirror sphere and graph cut. In *CVPR*, pages I: 140–147, 2005.

[20] S. You, J. Hu, U. Neumann, and P. Fox. Urban site modeling from liDAR. In V. Kumar, M. L. Gavrilova, C. J. K. Tan, and P. L'Ecuyer, editors, *ICCSA (3)*, volume 2669 of *Lecture Notes in Computer Science*, pages 579–588. Springer, 2003.