2005-12

Triangle Grouping and Structure Recovery for 3D Building Modelling and Visualization

Joe Hegarty
Technological University Dublin, joe@tudublin.ie

James Carswell
Technological University Dublin, james.carswell@tudublin.ie

Recommended Citation

Hegarty, J. &Carswell, J. (2005) Triangle grouping and structure recovery for 3D building modelling and visualization. *5th International Workshop on Web and Wireless GIS (W2GIS2005),* Springer-Verlag LNCS; Lausanne, Switzerland. December.

2005-12-01

Triangle grouping and structure recovery for 3D building modelling and visualization

Joe Hegarty
Dublin Institute of Technology, joe@dmc.dit.ie

James D. Carswell
Dublin Institute of Technology, jcarswell@dit.ie

SAMATS - Triangle Grouping and Structure Recovery for 3D Building Modeling and Visualization

Joe Hegarty and James D. Carswell

Digital Media Centre, Dublin Institute of Technology, Aungier St., Dublin 2, Ireland
{joe@dmc.dit.ie, jcarswell@dit.ie}

Abstract. Location based and spatial technologies research for the web has endless application for mobile/position content delivery (m-commerce or p-commerce). By exploiting the inherent location-based intelligence of the underling spatial component, relevant examples can include geometrically accurate and photo realistic virtual representations for: property assessments; land/marine information systems; routing information; on-line shopping; cultural heritage/tourist information/sites; etc. A major challenge for this technology is its reliance on professional developers when creating the virtual worlds used for web-based navigation of these services. This paper describes SAMATS, a Semi-Automated Modeling And Texturing System, which has the capability of producing geometrically accurate and photorealistic VR building models for web-based p-commerce applications from a set of geo-referenced terrestrial images. This paper describes the second of three main components that comprise the full functionality of the complete SAMATS implementation. It focuses on the triangle grouping and structure recovery steps, while providing an overview of SAMATS' other components.

1 Introduction

2D and 3D information visualization using VR modeling is becoming an important area of e-commerce research for today's web-based location based services (LBS) applications. Examples of exploiting VR navigation for both cultural heritage and environmental applications can be found in [1,2,5]. However, producing visually convincing VR models for these LBS applications requires expert VR knowledge on the part of the system developers. This research investigates building reconstruction technology for creating geometrically accurate, photorealistic 3D models from terrestrial digital photography for use in LBS applications that non-expert VR developers can exploit. It is envisioned that the resulting 3D model output from this work be web-enabled and made available to subsequent LBS research endeavors (e.g. for archaeologists, town planners, tourism, e-Government, etc.). Being able to produce 3D VR building models using terrestrial imagery allows all users to exploit the future commercialization potential of web-based LBS.

In the literature, it can be seen that many previous and contemporary modeling systems require manual correspondences to be made across the image set in order to accurately determine the models 3D structure. For example, Ullman (1976) was among the first to investigate the principle of modeling structure from motion and along with Taylor and Kriegman (1995) require manual correspondences to be made. [12,13] Debevec et al (1996) approached the problem differently by creating a modeling and rendering system that allows the user to create models using a set of block primitives and by setting constraints on these primitives.

More automated modeling approaches are seen to involve the modeling of roofs from aerial imagery. However, models produced in this way fail to capture building façades accurately. Countering this, Lee et al [8,9,10] have looked into the merging of façade textures from ground based imagery with models produced from aerial imagery. Results closer to our approach can be found in [3] where a large set of 3D building models is constructed by using spherical mosaics produced from accurately calibrated ground view cameras fitted with a GPS device. Although highly automated, this system was limited to modeling simple shaped buildings by simply identifying the rooflines and extruding walls downwards. [14] Still closer is an example of extracting building and window edges which, like SAMATS, determines correspondences automatically, although a rough model of the structure being modeled is required in order for this system to work. This approach differs from SAMATS as we do not require such a model to be available a-priori.

SAMATS uses a novel approach to creating building models without the need for manual correspondences to be made. The ability of SAMATS to remove the manual correspondence step found in most modeling approaches is achieved by having all images geo-referenced in the same reference frame. However, the acquisition of geo-referenced terrestrial images is still a bottleneck that does not have a straightforward solution. It is a process that requires knowing both the X,Y,Z ground coordinates of the camera station plus the orientation of its field of view. SAMATS does not solve the difficulties in acquiring geo-referenced imagery - it only investigates the usefulness of such imagery in the overall modeling process.

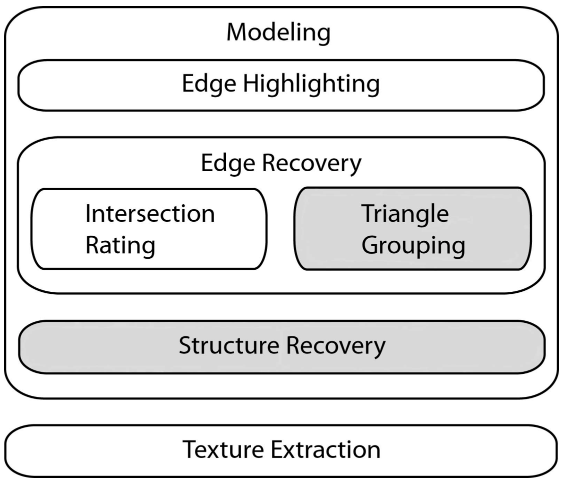

Fig. 1. SAMATS system diagram. The highlighted steps are the focus of this paper.

Modeling and rendering in SAMATA is a 2 stage process. The first stage is broken into 3 steps – namely: building edge highlighting; building edge recovery; and building reconstruction (i.e. structure recovery). This paper focuses on the triangle grouping and structure recovery steps in the modeling stage of SAMATS, but for completeness gives an overview of the other components. For a detailed description of the edge highlighting component and the intersection rating component refer to [6]. For all other components refer to [7]. Figure 1 shows a systems overview of SAMATS.

2 Modeling

This section describes the process used to model the geometry of a building from a set of geo-referenced images using only simple edge highlighting by the user. The basic concept behind the modeling process is as follows; if one has two images of a scene taken from different locations, and the exact position and orientation of the camera is known for each image (i.e. the exterior orientation parameters $X_o,Y_o,Z_o,\Omega,\Phi,K$) then the exact location of any point visible in both images can be determined. This configuration is illustrated in figure 2. The modeling process outlined in this section extends this idea by using planar triangle intersections to find edges rather than line intersections to find points. The modeling process can be split into three main steps; Edge Highlighting, Edge Recovery and Structure Recovery.

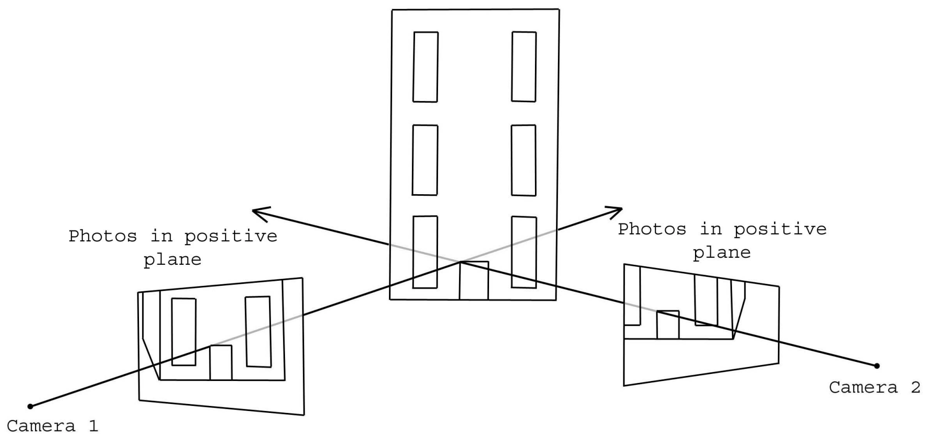

Fig. 2. Line projection used to determine a point in 3-space.

2.1 Edge Highlighting

Edge highlighting is the only manual step performed by the user in the SAMATS modeling process. Primary lines and secondary lines are used to highlight edges in the images. Primary lines are used to recover the position of building edges directly, determining the core structure of the model. They are responsible for the creation of every vertex in the final model. A secondary line is used to connect these primary lines together and must have each of its endpoints connected to one or more primary lines.

The reason the entire model is not defined by primary lines is because it is difficult to recover some edges given the input data. Primary lines are well suited to recovering the position of vertical edges because it is possible to create arbitrarily large angles of intersection about the vertical edge axis. However, for horizontal edges near camera level it is not possible to create arbitrarily large intersection angles, making it difficult to recover the horizontal edges accurately since slight inaccuracies in the camera's interior or exterior orientation parameters results in large errors in estimated edge location.

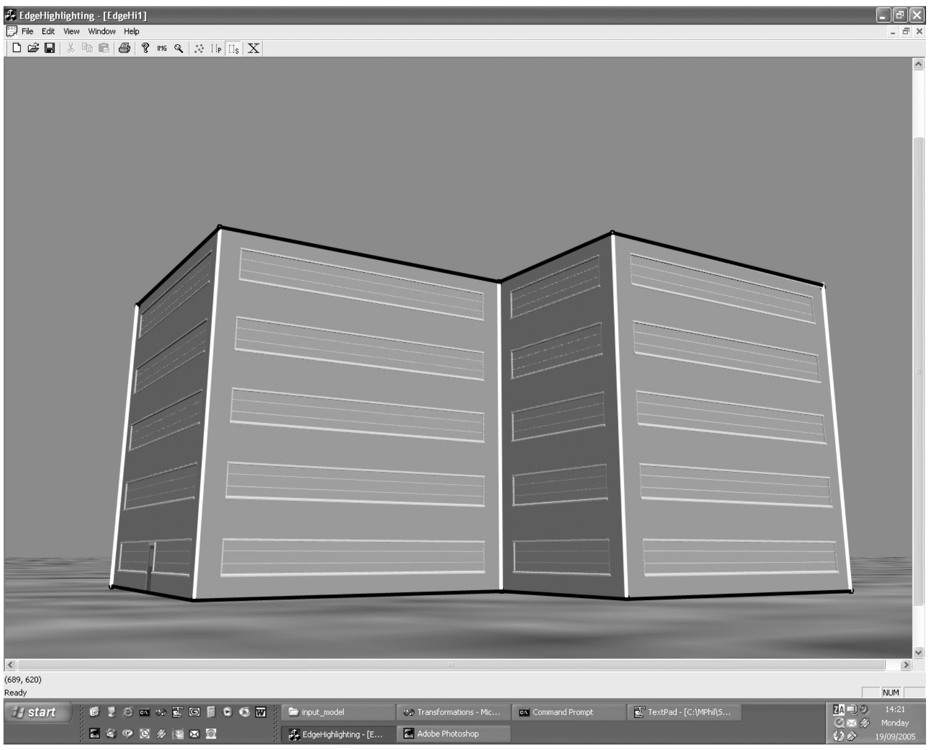

Fig. 3. Screenshot of the edge highlighting application. Note that the vertical edges are highlighted using white primary lines while the horizontal roof tops and building footprints are highlighted using black secondary lines.

Secondary lines work by connecting primary lines, where the use of a primary line would be prohibitive due to insufficient intersection angle between the triangle planes. Since primary lines will generally be used to recover the vertical edges of buildings, secondary lines should then be used to highlight the horizontal wall bases (building footprints) and roof tops, which indicates to the system that these edges should be connected without invoking the same recovery technique used for the primary edges.

Primary edge must be highlighted in at least three images, this is a requirement of the automated correspondence algorithm. It can be advantageous to define a primary edge in more than three images when trying to recover edges that are poor primary edge candidates. Secondary edges need only be defined in a single image. Figure 3 shows a screenshot of the edge highlighting application.

2.2 Edge Recovery

After the primary edges have been manually highlighted, six automated steps are performed to recover the final edges; Line Projection, Triangle Intersection, Correspondence Recovery, Edge Averaging, Vertex Merging, and Secondary Edge Recovery. Each of these steps is described next.

2.2.1 Line Projection

The first step in determining the positions of the primary edges is to project the 2D primary lines to form 3D triangles. The interior and exterior orientation parameters of the camera are used to project the primary lines from the cameras position out to infinity. This is performed for every primary line in each image.

2.2.2 Triangle Intersection

Once every 2D primary line has been transformed to a 3D triangle, the next step is to determine the intersections between the triangles. Every triangle stores a list of the triangles it intersects.

2.2.3 Correspondence Recovery

Generally each triangle intersects many other triangles even though only a small number of the triangle intersections have both their primary lines highlighting the same edge. Most 3D modeling systems resolve this problem by performing manual correspondences between the lines so that lines which highlight the same building edge are grouped together. Once the lines are converted to triangles the only valid intersections are between members of the same group. This can be a very time consuming process. SAMATS improves on contemporary techniques by performing this correspondence automatically in three steps; Intersection Rating, Triangle Grouping and Group Merging.

2.2.3.1 Intersection Rating

The process of intersection rating requires every triangle to rate each of the triangles it intersects to determine which of the intersecting triangles represent the same primary edge as itself. This automated rating process exploits the condition that there must be at least three primary lines, and hence triangles, for each primary edge. Each intersecting triangle is not rated on the coverage of the intersection line it makes, but rather on the similarity of its intersection line with others.

At the end of the intersection rating step, the list of intersecting triangles for each triangle will have a rating. Also, since the rating system is based on comparing intersection lines, a reference to the triangle responsible for the rating is also stored. For example, triangles $\mathbf{t}_i$ $\mathbf{t}_j$ and $\mathbf{t}_k$ all intersect each other. If $\mathbf{t}_j$ is the best rated intersecting triangle of $\mathbf{t}_i$, and it was a comparison between the intersection lines $\mathbf{l}_{ij}$, $\mathbf{l}_{ik}$, and $\mathbf{l}_{jk}$ which were responsible for this rating, then a reference to $\mathbf{t}_k$ will be stored along with this rating for $\mathbf{t}_j$ in $\mathbf{t}_i$'s intersecting triangles list.

2.2.3.2 Triangle Grouping

After the intersection rating step, for every triangle $\mathbf{t}_i$, every intersecting triangle $\mathbf{t}_j$ will have a rating assigned to it. Also, the $\mathbf{t}_k$ responsible for each $\mathbf{t}_j$ rating will be stored along with the rating. This information can then be used to group triangles together, with each group representing a primary edge.

Essentially, the grouping process is performed in two steps. Firstly, the GSS (Group Scope Set) of each triangle is determined. The GSS for a triangle $\mathbf{t}_i$ is a list of triangles which contains the triangle itself (in this case $\mathbf{t}_i$), the GSS for $\underline{\mathbf{t}}_j$ (the best rated $\mathbf{t}_j$) and the GSS for $\underline{\mathbf{t}}_k$ (the $\mathbf{t}_k$ for $\underline{\mathbf{t}}_j$). The GSS can only hold a single instance of any triangle. This ensures that the recursive triangle grouping algorithm terminates. Not every triangle will have the same size GSS. The size of these sets will vary depending on the number of triangles used to represent each primary edge as well as the relationship between their intersection lines.

The simplest case arises when a primary edge is represented by three triangles. In this configuration each triangle $\mathbf{t}_i$ refers to the other two as either its $\underline{\mathbf{t}}_j$ triangle or as its $\underline{\mathbf{t}}_k$ triangle. In such a situation all three triangles have identical GSS containing the three triangles, see figure 4.

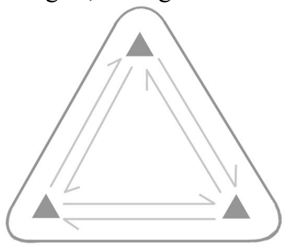
Fig. 4. Three triangles, all with the same GSS.

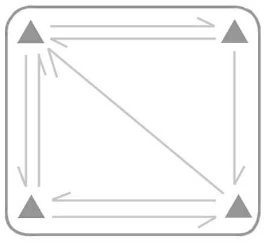
Fig. 5. Four triangles, all with the same GSS.

If there are more than three triangles representing a primary edge there can be three broad types of set configuration. One configuration involves four or more triangles that represent the same primary edge with every triangle having identical GSSs, see figure 5. Another configuration involves four or more triangles that represent the same primary edge but with only a subset of triangles having identical GSSs, while the other triangle(s) have GSSs containing the subset of triangles plus additional triangles. This results in the real group consisting of four or more triangles although the GSSs of some of the triangles will only have a subset of these triangles, see figure 6. The final configuration involves six or more triangles that represent the same primary edge but with each triangle having one of two or more GSSs. In this configuration each group is solved independently and then the groups are merged as a post-process, see figure 7. Any combination of the above configurations can also occur together.

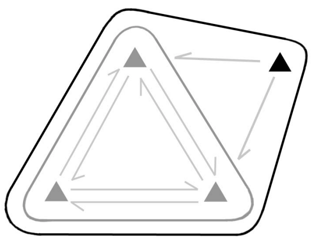
Fig. 6. Four triangles, three of which have the same GSS. The unreferenced triangle has a GSS containing all four triangles.

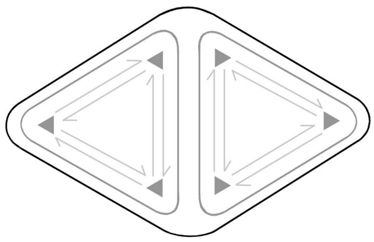
Fig. 7. Six triangles forming two separate GSSs. The black line represents the group after merging.

The second step in the grouping process is to use the GSSs to group the triangles into groups. The grouping algorithm runs in two phases. In phase one only triangles that have three triangles in their GSSs are processed. Each triangle as well as its GSS members are assigned a new group. The first phase solves either fully or partially the configurations shown in figures 4, 6, and 7. At the end of this phase the majority of triangles will have been assigned a group. Only triangles which have a configuration similar to that shown in figure 5 or unreferenced triangles like those shown in figure 6 remain. In phase two these remaining triangles are assigned a group. If a triangle refers to triangles in an existing group (figure 6), it is added to that group provided that its rating in this group is within some minimum threshold. If a triangle's GSS has triangles which have not yet been grouped, a new group will be created for these triangles (figure 5). It may not be possible to assign a group to every triangle for a number of reasons. For example, the user may not have used three primary lines to highlight a particular primary edge. Also there may be too great an error to group some primary lines together either due to an error in the camera's interior and/or exterior orientation parameters or an error in primary line placement by the user. In such cases these triangles are marked as invalid.

2.2.3.3 Group Merging

The final step in the grouping process is group merging. This is required because sometimes a primary edge may be represented by 6 or more triangles, which form 2 or more self-contained groups with no inter-group referencing (figure 7). If the groups were left the way they were, there would be 2 primary edges representing the same building edge instead of just one. The merging step simply compares each group to each other group by first comparing the highest ranked members of each group to each other. If it is found that the ranking between these triangles is within some threshold, the algorithm goes on to test every combination of group members together to guarantee that they; a) all intersect, and b) the lowest ranking observed is within some minimum threshold. If these two criteria are met, the two groups are merged.

2.2.4 Edge Averaging
Once all triangles have been assigned a group the primary edges must be determined for each group. This is simply the weighted average of all the intersection lines between all group members.

2.2.5 Vertex Merging
During the edge averaging step, each primary edge will be created totally independently from all other primary edges. In most cases this is acceptable since the majority of primary edges are not connected to any other primary edge. Sometimes however primary edges are connected. This is indicated in the edge highlighting step by having two or more primary lines share the same endpoint.

All primary edges that are connected need to have their connected endpoints coincident. This is achieved by creating a mapping between every primary line and every primary edge, and also between every primary line endpoint and every primary edge vertex. Once the mappings have been made, we can see if any of the primary lines share the same endpoints, which maps to primary edges sharing the same vertex. Once the vertices are identified they are set to the average of their positions.

2.2.6 Secondary Edge Recovery
Secondary edges are determined using the same mapping information obtained during the vertex merging step. First the secondary lines endpoints are determined. Then the corresponding vertices are determined for these endpoints and a new group is created for each secondary line using these vertices as the secondary edges endpoints. The outline of the model should be complete. See figure 8 for a screenshot of the recovered primary and secondary lines of the building shown in figure 3.

2.3 Structure Recovery

Even though the outline of the model has been determined there is still no surface data associated with the model. The model is only defined in terms of vertices and lines, and not in terms of surfaces and the triangles that make up each surface. Recovering this structural information is broken into three steps; Surface Determination, Surface Aligning, and Surface Triangulation.

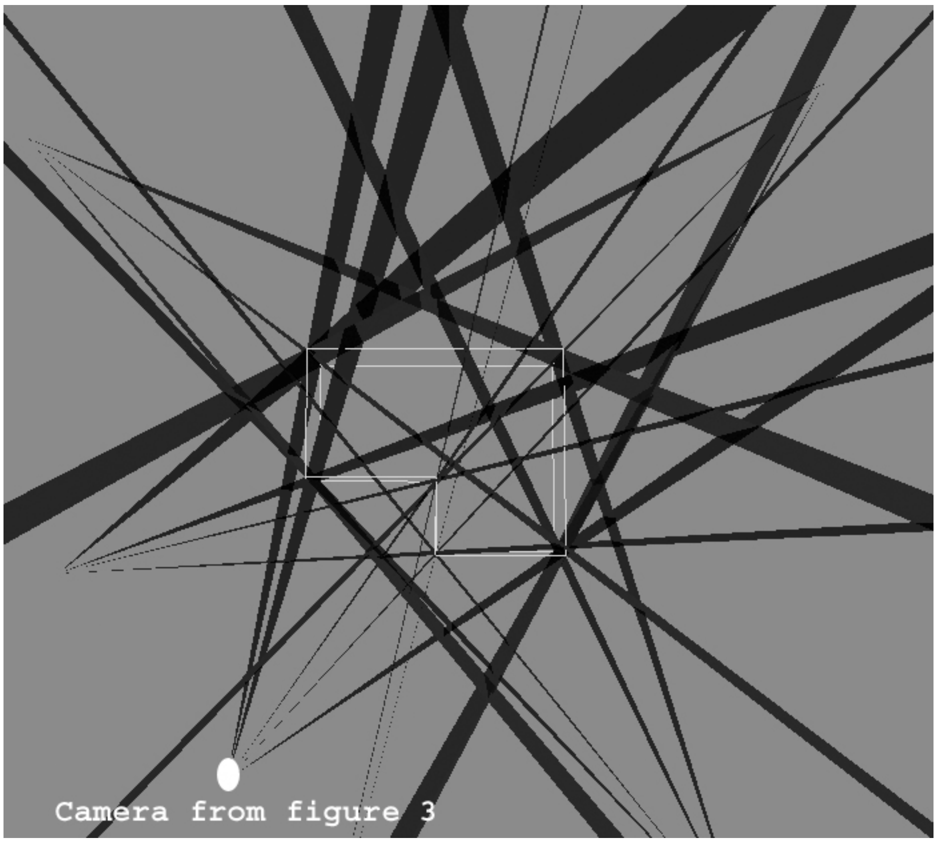

Fig. 8. Screenshot of the recovered building from figure 3. Note that the location of the camera from figure 3 is highlighted. The projection of the 5 primary lines are clearly shown.

2.3.1 Surface Determination

Surfaces are determined by treating the model as a graph, with the models vertices representing nodes in the graph and the primary and secondary edges representing the edges in the graph. Each surface corresponds to a cycle in the graph, but not every cycle in the graph corresponds to a surface, as illustrated in figure 9.

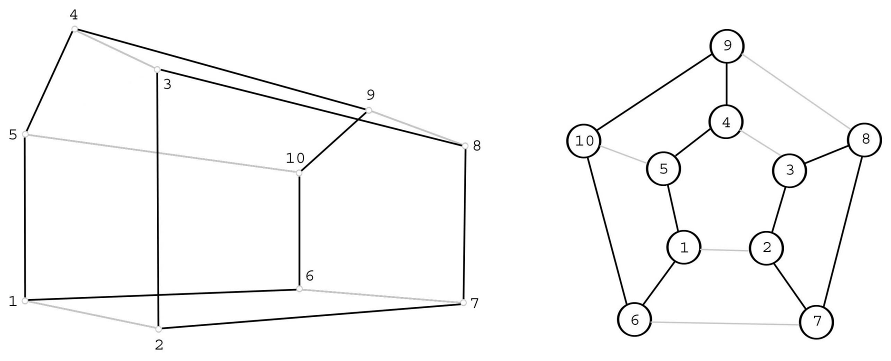

Fig. 9. The black outlines represent cycles in the graph. One of the cycles represents a surface (2-3-8-7), while the other does not.

There are two main assumptions made in order to determine the surfaces from the vertices and edges; the model must be closed and the number of surfaces associated with each vertex is equal to the number of edges connected to it. Surfaces are then determined by finding the shortest cycles in the graph where all the vertices are co-planar.

2.3.2 Surface Aligning

Once all the models surfaces have been determined, the normal vector for each surface must be determined. The first step is to determine the adjacency of the surfaces, i.e. which surfaces are adjacent to each other. This is performed because surfaces are aligned in pairs. Once the surface adjacencies have been determined one of the surfaces is flagged as the master surface, while all other surfaces are flagged as slave surfaces. First all slave surfaces that are adjacent to the master are aligned, becoming themselves masters in the process, then all slave surfaces adjacent to these new master surfaces are aligned, becoming masters themselves. The process continues recursively until all surfaces have been flagged as masters. The aligning step uses the fact that adjacent surface pairs are attached along one of their edges. This edge can act like a hinge between the two surfaces making it possible to rotate one of the surfaces about this hinge so that the two surfaces are co-planar. If then the surfaces are transformed so that they are perpendicular with the z-axis with the hinge between them aligned with the x-axis, we notice that the interior of one surface is above the hinge while the interior of the other surface is below the hinge.

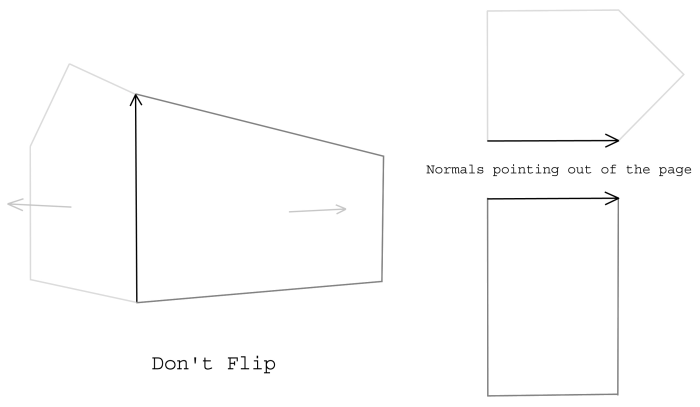

Fig. 10. The surfaces are on opposite side of the edge vector. Therefore the surfaces are correctly aligned.

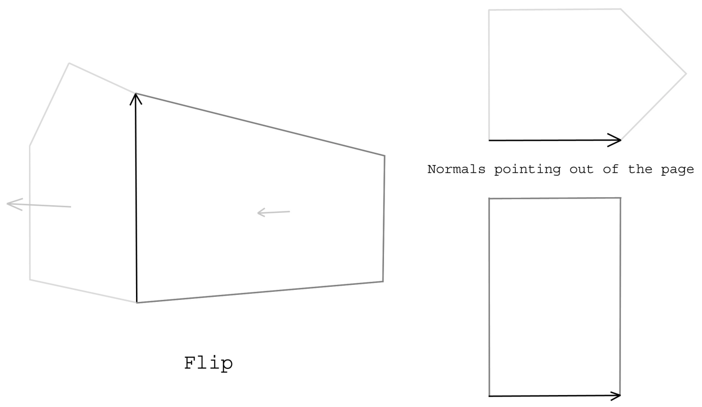

Fig. 11. The surfaces are on the same side of the edge vector. Therefore the normal of the slave surface needs to be inverted.

Using this fact each surface pair is aligned by transforming both the master surface and the slave surface so that their surface normals are aligned with the z-axis and the edge vector between them is aligned with the x-axis. Then each surface is checked to see if its interior is above or below the hinge edge. If both surfaces are on the same side of the hinge edge they are misaligned so the normal of the slave surface is flipped. If the two surfaces are on opposite sides, the two surfaces are already aligned, see figures 10 and 11.

Even though the models surfaces have been determined at this stage there maybe a serious problem with the models normals, they may all be pointing inwards instead of outwards. This is due to the fact that a random surface was chosen as the master surface at the beginning of the surface aligning step but it was not determined whether or not this normal points inwards or outwards. Luckily this is not a serious problem since all we have to do to rectify the situation is flip all the surface normals.

2.3.3 Surface Triangulation

Once each surface has been determined and aligned, each surface must be decomposed into triangles. The surfaces in the model can be either convex or concave although the surfaces should not contain holes. There are many factors that can be used to determine how a surface should be decomposed; minimize the number of triangles created, try to keep all triangles equilateral, try to keep all triangles close to equal area. The algorithm used to triangulate each surface can be found in [11]. This algorithm does not take any of these factors into consideration however. First each surface is orientated so that it is perpendicular with the z-axis. The z-coordinate is then ignored and the triangulation process treats the surface as if it was a 2D surface.

3 Texture Extraction

Coming into this section, we have a geometrically accurate model of the building. However, there exists data contained in the image set that has not yet been used to increase the models realism, the buildings façades. The SAMATS texture extraction process takes the façades from the images and applies them to the model. An overview of this component is presented next. For a more detailed explanation of the Texture Extraction component refer to [7].

3.1 Overview

The aim of the texture extraction process is to produce a 3D model with photorealistic textures. The texture extraction process can be broken into a number of steps. Firstly, the number of images that will contribute to each triangle is determined using back-face culling. There can be any number of contributing images, with each image's contribution first being stored in a temporary texture before they are all blended together per-pixel based on the camera-surface distance and orientation. Occlusion maps are used to prevent incorrect façade data being stored with each triangle. All triangles are then packed into a single large texture retaining the relative size of each triangle, thus creating an authalic texture map. The texture coordinates for each triangle are then set to sample the correct region of the texture map, with the texture then being assigned to the model. Figure 12 shows the final packed texture for the example scene and figure 13 shows a screenshot of the final model created.

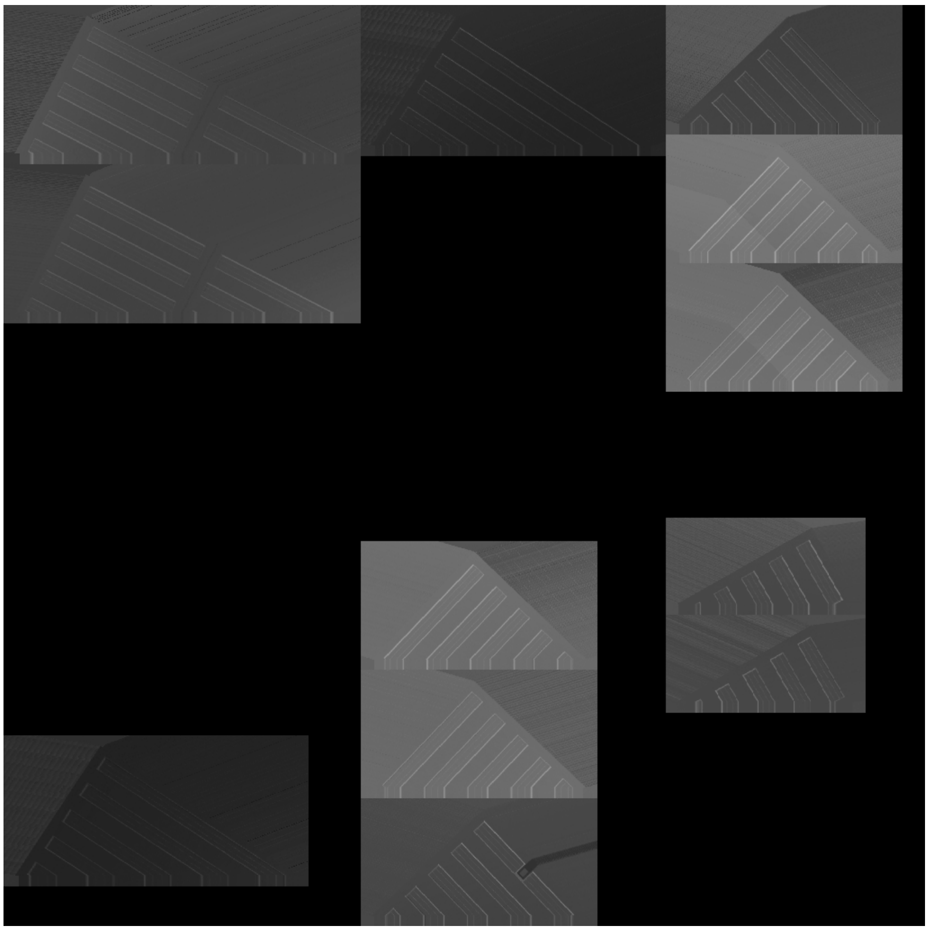

Fig. 12. Final packed texture of the example scene. Textures packed large to small from top to bottom, left to right. The black gaps in the middle are for the roof and floor triangles which have no texture information from the image set. Also note the color clamping from the border of each triangle, most noticeable for the door at bottom row middle column.

4 Conclusions

This research shows that given sufficient geo-referenced terrestrial imagery, user input to the modeling process can be reduced significantly. In SAMATS, user input is required for the edge highlighting step but since no correspondence is required this step could potentially be automated using edge detection and a set of heuristics to guide the choice between using primary lines or secondary lines.

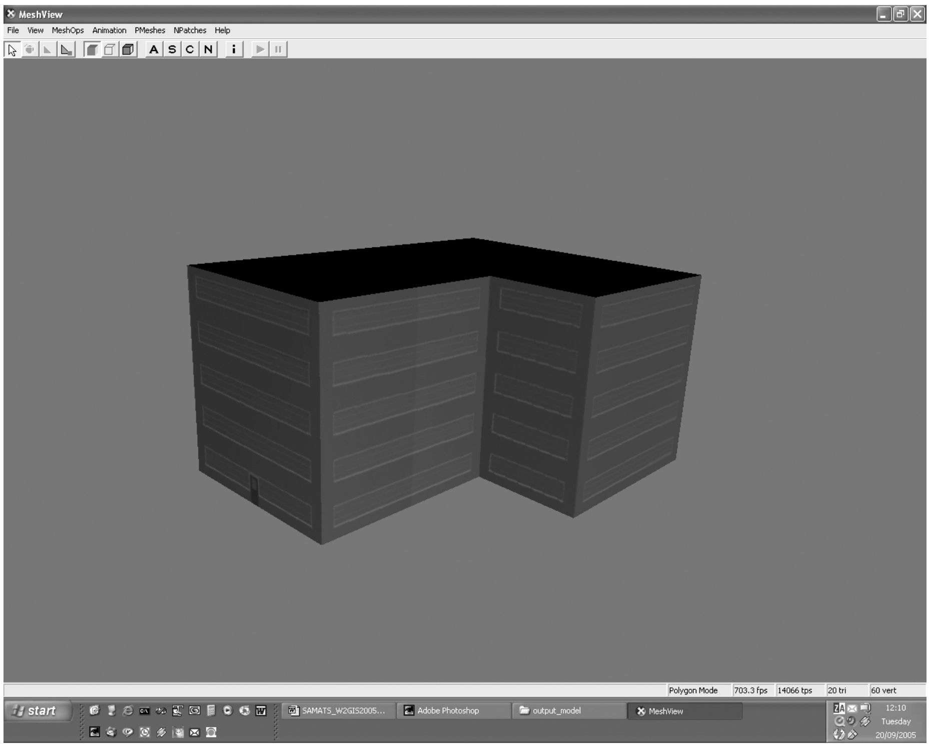

Fig. 13. Screenshot of the final model

To date, SAMATS has only been tested on synthetic images where the exact EO and IO parameters of the camera are known. Achieving such precision in the real world would prove difficult without specialized equipment. As such, new techniques for the non-expert will be required to facilitate the gathering of the geo-referenced images required by SAMATS in order for this system to be utilized effectively in the real world. As the user friendliness and functionality of today's GPS enabled digital imaging technology improves over time this constraint may no longer apply - making the acquisition of accurate geo-referenced imagery as easy as regular imagery.

SAMATS has shown the ability to model rectangular and triangular roofed structures very well; however SAMATS does have trouble modeling certain other structures. For example, SAMATS has no special ability to model curved surfaces accurately where cylindrical column must be replaced by rectangular columns. Another difficulty that can arise is SAMATS' inability to handle partially highlighted edges making it difficult to model buildings in tightly confined spaces. However, in many cases SAMATS is proving very effective as a 3D modeling and visualisation tool for the non-expert when developing applications of web-based VR LBS.

References

1. J.D. Carswell, A. Eustace, K. Gardiner, E. Kilfeather. An Environment for Mobile Context-Based Hypermedia Retrieval, in 13th International Conference on Database and Expert Systems Applications (DEXA2002); IEEE CS Press; Aix en Provence, France; September 2002
2. J.D. Carswell, M. Bertolotto, N. Mandrak. Applications of Mobile Computing for Fish Species at Risk Management: in Proceedings of International Conference on Environmental Informatics of International Society of Environmental Information Sciences (ISEIS2004); Regina, Canada; August 2004
3. S.R. Coorg. Pose Imagery and Automated Three-Dimensional Modeling of Urban Environments. PhD thesis, MIT Ph.D. Thesis, 1998.
4. P.E. Debevec, C.J. Taylor, and J.Malik. Modeling and Rendering Architecture from Photographs: A hybrid geometry- and image-based approach. In SIGGRAPH '96 Conference Proceedings, 11-20, 1996.
5. K. Gardiner and J.D. Carswell. Viewer-based Directional Querying for Mobile Applications: International Workshop on Web & Wireless Geographical Information Systems (W2GIS2003); IEEE CS Press; Rome, Italy; December 2003
6. J. Hegarty and J.D. Carswell. SAMATS – Edge Highlighting and Intersection Rating Explained. Proceedings of 2nd International Workshop on CoMoGIS, 314 – 323, 2005.
7. J. Hegarty. SAMATS – Semi-Automated Modeling And Texturing System. Masters Thesis, Dublin Institute of Technology.
8. S.C. Lee, S.K. Jung, and R. Nevatia. Integrating Ground and Aerial Views for Urban Site Modeling. In Proceedings of International Conference on Pattern Recognition, 2002.
9. S.C. Lee, S.K. Jung, and R. Nevatia. Automatic Integration of Façade Textures into 3D Building Models with a Projective Geometry Based Line Clustering. Computer Graphics Forum, 21(3):511-519, 2002.
10. S.C. Lee, S.K. Jung, and R. Nevatia. Automatic Pose Estimation of Complex 3D Building Models. Proceeding of the 6th IEEE Workshop on Applications of Computer Vision, 2002.
11. J. O'Rourke. Computational Geometry in C (Second Ed.). Cambridge University Press, 1998.
12. C.J. Taylor and D.J. Kriegman. Structure and Motion from Line Segments in Multiple Images. PAMI, 17(11):1021-1032, November 1995.
13. S.Ullman. The Interpretation of Structure from Motion. Proceedings of the Royal Society of London, 1976.
14. S. Zlatanova and F.A. van den Heuvel. Knowledge-based Automatic 3D Line Extraction from close range images. Web – http://www.gdmc.nl/zlatanova/thesis/html/refer/ps/SZ_FH_Corfu.pdf