30 Days to Build a City: Rapidly Extruding Realistic Virtual Worlds

Shaun Bangay
Computer Science Department
Rhodes University
Grahamstown
6140
South Africa
Email: cssb@cs.ru.ac.za

Abstract

Models of existing buildings are finding increased application for walkthroughs, visualization and promotional material. This paper describes the modelling of an entire city, including creating realistic models of individual buildings, accurately positioning each structure and adding the surrounding terrain and roads. Each building is matched to the original floor plan, provided with a realistic roof geometry, and painted with colours and textures corresponding to the original structures.

Mechanisms for providing drive-arounds, route finding and attaching multimedia content to the resulting model are investigated.

The techniques described emphasize efficient and realistic model construction. The concepts are applied to a case study: the creation of the virtual Grahamstown model. The process of creating this model from initial conception, to finished multimedia presentation took 30 days.

1 Creating a World

The city of Grahamstown, South Africa is the venue for an annual National Arts festival, an event which gives rise to a dramatic increase in population, and during which many of the local halls and auditoria are commandeered for use as venues for performances and art exhibitions.

A mockup of the city has been created in a virtual reality system for use as an exhibit during the festival. Each venue is associated with relevant multimedia items; these are linked to the virtual representation of that particular building. Exploration of the model assists in providing direction and information to visitors unfamiliar with the city. This paper describes the process of creating this virtual world. The discussion concentrates on the manner in which realistic imagery is produced under tight time constraints, and deployed using a number of inexpensive, low-end workstations.

Two versions of the virtual world have been created: a VRML version for linking the multimedia content to the various venues, and a version which uses the virtual reality system RhoVeR [Ban96] to create a drive-through of the virtual city.

The process of building a city involves capturing building outlines from existing maps and plans, extrusion into 3 dimensions, applying appropriate colours and textures, placing the buildings correctly amid suitably generated terrain and allowing exploration of the complete model. The remainder of this paper describes the tools and techniques used to facilitate this process.

2 Building Cities

Similar digitization efforts have been undertaken in the past. There is an increasing tendency to provide VRML models of sites as part of the content of the WWW site at many institutions [Web97][Boy97]. Models of cities such as Tokyo [3DL97] and the inner city of Frankfurt [Goe96] are currently publically available, while other city datasets can be had at a price [Axi98]. Many tools exist for creating 3D models, some dedicated specifically to architectural design, and to linking multimedia content to portions of the building [Par97]. The use of architectural plans to extrude detailed building models is described in [Lew96].

The primary differences between these efforts and the tools described in this paper is that this software is limited specifically to creating shells of buildings, and placing them within the terrain. The controls are optimized for this purpose, allowing rapid digitization with minimal overhead. The design caters for the mass digitization of many buildings and their correct placement relative to one another and to the underlying terrain. Detail of the model must be limited to support interactive rendering speeds on low-end workstations, while still encouraging recognition of key buildings and landmarks.

The implementation of a custom tool has a number of advantages:

- The tool can be dynamically customized to the nature (and limitations) of the data being captured.
- Common processes, and repetitive tasks, specific to the application can be automated.
- The interface can be customized to overcome the limitations of the humans using it, at least one of whom is suffering from RSI induced by frequent HCI (with these latter issues being extremely relevant to the digitization effort).

3 Selecting Maps

The virtual world needs accurate representations of some of the buildings (the important venues), as well as appropriate representations for the less important structures. The model is used to provide information on the route between venues, so correspondence with the real world is a requirement. Details of the terrain layout, in terms of gross geographical features, provide additional realism to the model.

The terrain layout is derived from contour maps of the city, which include building outlines. Some smaller higher resolution maps provide detailed building outlines. A number of maps covering adjacent areas are used since no one map provides a complete view of Grahamstown in the required detail.

Details of the features on each building are created using texture maps. The source of these maps are images photographed directly from that building.

Use of aerial photographs was considered for obtaining building outlines, as used in [Sti96], but not attempted in this case due to the ready availability of the contour maps. An initial attempt was made to use aerial photographs as the source of textures on the buildings. Ultimately higher resolution textures captured from photographs of individual buildings are necessary to generate the required realism.

4 Creating Buildings

The maps available contain plan outlines of the walls of the various buildings. These walls must be raised to the appropriate levels, and a roof added. Suitable textures are then specified for the surfaces of the walls and roof.

Once the maps have been scanned in, the floor plans of the buildings are used as a template for specifying the arrangement of the walls. An optical recognition system was considered for tracing the lines representing building outlines, but this proved infeasible for a number of reasons. The

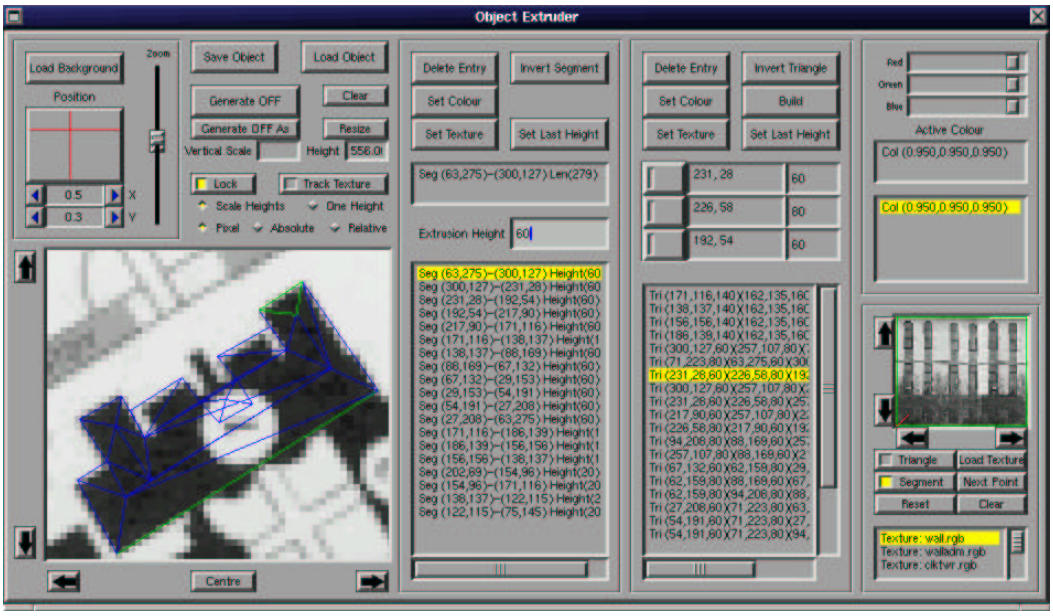


Figure 1: User interface presented by the extrusion tool.

quality of the maps is less than perfect - they contain lines representing contours and items of vegetation which would confuse any line tracing algorithms. The resolution of the maps after they have been scanned is also quite low, with some features of the building being of size comparable with individual pixels in the image. Higher resolution scans are possible, but these place additional demands on storage and processing resources already limited by the size and number of maps required.

In many cases the judgement of a human operator is required to identify which features of a building are significant. Since the models are used in a virtual world where interaction speeds are important, there is a tradeoff between the level of detail on the structures and the final rendering speed. Usually only a simplified outline for the building is appropriate, and kinks introduced by small projections in the shape can be smoothed over. Only features that are particularly characteristic of the building are retained.

The extrusion process is controlled by a specialized tool created specifically for the purpose. The features of this tool are described in the following sections. A screenshot of the user interface for the tool is shown in Figure 1.

4.1 The Extrusion Tool: Construction primitives

The process of creating a building consists of raising the two dimensional floor outline obtained from the map, to create a three dimensional volume which represents the portion of the building enclosed by the walls. This can be viewed as extruding the building through the template provided by the floor plan. The resulting shape is represented as the set of polygons which make up the walls of the virtual building. This must then be capped by a roof, also represented as a collection of polygons. The polygon primitive is supported by both virtual reality systems for which the buildings are destined.

The two primitives used to create buildings are vertical rectangles, used to form most of the walls, and triangles which are used to construct the roof, and any other specialized feature. The use of these primitives is described in the following sections.

4.1.1 Walls

Use of the extrusion tool requires an image file containing the map with the desired building outline. This image file is displayed as the background to the drawing area, over which the outline of the building is captured. The background image can be translated and scaled to allow the building outline to fill as much of the drawing area as possible. As a result capture of the outline is faster, and less stressful, since cursor positioning need not be particularly precise.

The outline of the building is captured as a sequence of line segments. Normally end points of successive segments will overlap, to produce a closed polygon. Points within a small distance from each other are automatically identified (used to represent a single vertex). Again this simplifies the process of capture, and ensures that the end points of overlapping segments are not duplicated.

Each line segment in the outline forms a rectangular wall in the final virtual building. An additional parameter must be specified representing the height of the wall (relative to the length of the base segment) with ground level representing the origin of that dimension. The position of ground level (relative to sea-level or any other convenient reference point) can also be specified as a base offset for the entire structure.

4.1.2 Roofs

Initially consideration was given to the use of a standard template for roofs which could be selected and automatically placed on top of the buildings. The complexity and range of possible roof designs make this infeasible. Automatic generation of a pitched roof [Aic95] is complicated by the need to conform to the version applied in reality. Instead provision is made to support the lowest common denominator of the common triangular pitched roof, the triangle.

The horizontal projection of the triangle is captured in a manner similar to the line segments for the walls; by sketching the vertices on the background image. Unlike the walls, the roof details are not contained in the image and so additional artistic skill, and images/memories of the building are useful for filling in the missing details.

The height of each vertex of each triangle may be independently specified, with the system attempting to provide the most suitable default values. Initial vertex heights default (in order, depending on availability) to:

1. the height of the vertex from another triangle sharing that vertex,
2. the height of an overlapping wall segment ending at the point,
3. the last height specified for triangle vertex or line segment.

The triangle formulation is extremely versatile. A conventional pitched roof with two triangular endfaces, and two trapezoidal sides can be quickly built using six triangles. More complex roofs can be easily decomposed into triangles, even by novice users of the system with only a few minutes training. Triangles can be used to construct vertical faces of the building, although in this case the visual representation can only be shown as a line on the drawing area. Vertex coordinates must be manipulated in numerical form only in this case.

4.2 The Extrusion Tool: Adding Realism

Once the shell of the building has been constructed, the addition of appropriate colours and textures improves the resemblance between the model and the original structure. Colour is most frequently used in cases where the building is relatively insignificant, or where the texture of an area would be relatively plain. Using a single colour on a polygon, as opposed to a texture, improves the speed of rendering that polygon on non-accelerated graphics hardware. The use of textures with important landmarks is almost mandatory, and the resulting realism is extremely high.

4.2.1 Colours

A single colour can be associated with each wall or triangle in the building. Colours may be mixed directly from their constituent primaries. Once a colour has been chosen for one component of the building it is added to the list of colours for that model which simplifies matching should it be needed again (which is extremely likely).

4.2.2 Textures

The textures for the building surfaces are available as rectangular image files, containing the features for the surface of the buildings such as shape of doors and windows and material details such as stone or brick.

The rectangular nature of the textures ensures an excellent correspondence with the walls of the buildings, but is less suitable for mapping onto the surface of one of the triangular facets. The extrusion tool allows the specification of the area of the texture map which is to be mapped onto the surface. This area is triangular in the case of a triangle, or a quadrilateral in the case of a wall. The texture is laid out in a window, and the outline of the selected area for each polygon is mapped onto it. In addition, the texture may be tiled both horizontally and vertically, allowing areas with repeated features to be efficiently generated from only a single instance of a texture of that feature. This has efficiency implications for the final rendering. Explicit storage of repeated features imposes additional memory overheads on a system that already makes substantial demands on that resource.

Default values are provided for the initial texture coordinates. The walls initially encompass the entire rectangular texture, while the triangular roof segments are mapped to an area corresponding to their projection onto the map. This latter choice ensures that adjacent roof triangles are seamlessly textured. Texture coordinates from other primitives can be shown on the map, allowing matching of textures on different sides of the building and seamless connection between adjacent polygons.

The ability to select active areas of the texture, even for the walls of the building, simplifies reuse of the texture maps for cases where alternative portions of the texture are required. A texture used for an entrance arch with a window above, could equally well be used for a wall containing just the window. This again gives additional savings in memory usage during rendering.

4.2.3 Buildings

The translation and scaling factors of the background image are combined with the segment coordinates to determine the coordinates of the building relative to the image file from which it is captured. Additional information about the position and dimensions of the image file in terms of the coordinate system of the original map allow the all objects to be transformed into a common coordinate system. In this way, buildings captured from different maps can automatically be combined into the same virtual world with the correct relative positions and dimensions.

5 Capturing Textures

The source of surface textures for each building is the building itself. This is an aspect in which this application differs from most architectural CAD applications, where the physical existence of the structure is less likely. It is a relatively simple matter to film the surfaces of the buildings and capture these for use as texture maps. As with many applications, however, there some difficulties in the process which can only be appreciated when the process is attempted in practice.

The major difficulty is the growth of the city. A substantial amount of greenery (trees, shrubs and other plants) has been planted close to the walls of buildings, and has grown to obscure surface details. A number of devious strategies are required to obtain all the details of the building's surface.

A simple solution is to film only those portions of the surface that are clearly visible, and tile those features over the rest of the texture map. This works reasonably well, and results in small texture maps, but assumes that sufficient surface area is visible to obtain the sample, and that the surface does contain repeated features.

A more complex solution is to film from a point between the plant layer and the wall. In most cases, there is not sufficient room to get a wide angle shot, and the pictures need to be taken looking vertically up the wall, often with a number of views for different heights. Pictures also need to be taken at intervals along the wall and later recombined. Images obtained in this way suffer from severe perspective distortion as height increases. This creates problems when attempting to recombine the images. A simple program to perform image warping proves sufficient to reverse this distortion. The warp is controlled by two lines in the image which are placed along the edges of vertical features. Similar transformations provided by commercial image manipulation packages have been used to achieve the same effect in other applications [Goo95].

Lighting of the surface being filmed also complicates the process of texture capture. Obviously surfaces with shadows across them will retain these when converted to texture maps. The best time to capture textures is in overcast weather, or failing that, around noon, when shadows are minimal. Lighting conditions change between the different samples at various heights (because the viewing angle changes), and between textures on different sides of the building. These prove to be particularly noticeable in the final texture, especially since they generally appear as horizontal/vertical lines, and are exaggerated by Mach banding effects. These can be reduced by touching up the image. The most effective solution is manipulation in the colour space of the image. The saturation of the individual images is reduced until they are all of comparable shades of grey. The images are recombined, and the saturation is restored, colouring the overall image toward the dominant colour for that texture. Some colour information is lost in the process, but this does not appear to have significant effect on the realism of the final rendered building.

The texture generation strategies applied here are specific instances of a general texture generation problem, as described in [Ofe97]. The problems of generating textures in the presence of perspective distortion and directional lighting effects can be comprehensively addressed by making use of correlation between- and transformation of- sequences of images, and by combining and averaging the lighting effects.

6 Providing the Background

Adding various geographical features such as roads, rivers and railway lines and terrain to represent hills and valleys is accomplished by using existing features of the extrusion tool. In particular the triangle primitive can be used to capture these features since they all have prominent horizontal projections.

The width of features such as roads and rivers are usually on the same scale as the dimensions of a building. The height field for the triangles is specified directly as the height of the contour line upon which a particular vertex lies.

There are some limitations in capturing terrain is this way. Occasional errors, the amount of effort involved, and the faceted nature of the resulting surface limit the realism of the resulting objects. These issues will be addressed in future development (see section 10).

7 Exploring the City

The models produced by the extrusion tool can be imported by VRML-aware systems, and by RhoVeR. Each of these systems allows viewing of the complete model. Additional context specific information is required to provide an opportunity for more useful interaction with the virtual city.

Applications of the model include drive-arounds, route finding and providing venue related information. These all require additional refinements to the model.

Figure 2: Digitized road map of Grahamstown, showing position of the maps used for digitization.

7.1 Producing a drive-through

The drive-around of the virtual city shows the view from within a vehicle travelling the streets of Grahamstown. To simplify the interface to the users, the car must be capable of following the roads automatically. User control is limited to regulating speed, and making direction decisions at intersections. In some cases, equipment limitations may require that the drive-through be demonstrated as a pre-rendered animation sequence, with even the routing decisions being made automatically.

These facilities require an ability to identify and follow the roads. A suitable random wander algorithm must provide an interesting tour of the city, in the absence of user input.

7.1.1 Road tracing

The roads are defined, for purposes of terrain rendering, as a collection of triangles. Rather than redigitize the roads in a format suitable for driving we decided instead to utilize the terrain data and convert it to a collection of paths. This would save effort in recapturing the data, and ensure consistency between the visible roads and the routes traversed. The nature of the captured road data introduces some complications to this process. The road data originally captured is shown in Figure 2. The roads cover a number of maps, and those roads crossing boundaries must be reconnected when converting to a path description. Intersections were originally captured by overlaying the two sets of polygons; no common vertices and segments are present in these cases. Errors in the capture process also result in gaps between supposedly adjacent triangles. It is thus clear that any path finding algorithm must have a degree of tolerance to flaws in the road data.

Two different algorithms are employed as alternative methods of generating paths from the road descriptions. Each consists of a core algorithm which would work for perfect data, and a number of correction routines usually driven heuristically which reduce the amount of error to an acceptable level. The result is a directed graph which is used to drive the driving simulator.

Centre Following An approximation to the path down the centre of the road can be obtained by connecting the centres of adjacent triangles. Adjacent triangles share a common edge. In "real" data, adjacent triangles may only share a portion of an overlapping edge, or some common area, or only a single common vertex. A heuristic approach for determining adjacency between triangles involves connecting those that have vertices (or mid-point) within a certain distance from the vertices (or mid-point) of another triangle. Naive application of this principle connects all triangles within the given radius, so the heuristic is restricted to joining the neighbouring triangles that are not already connected in a sequence of adjacent triangles. The path length of

the sequence when this connectivity is checked must be limited, so as to allow complete connection around small loops such as islands in the road and traffic circles.

Examples of the paths produced using these algorithms are illustrated in Figure 3. Paths between nodes in the resulting graph are present in duplicate, with one in each direction. The motive for using a directed graph will be clear after considering the results produced when using the edge tracing approach.

Edge Tracing In reality, a driver would hopefully follow the edge of the road, rather than meandering down the centre. In South Africa, the convention is to drive on the left hand side of the road, hence the left hand boundary must be identified. Positioning the car so that the left hand side of the car traces the left hand boundary of the road is sufficient to place the car in the correct lane.

The boundary line segments are identified from the digitized road triangles by eliminating interior line segments, which are common to two triangles. At the same time, the boundary segment can be identified as either a left or right hand road boundary. If the digitized data were completely accurate this information could be extracted from the order of the vertices. However, some triangles are inverted. The fact that the roads are all (almost) horizontal allows these triangles to be detected and reoriented.

The points of intersection of every pair of boundary segments are identified, and those that occur within both segments (within a small error margin) are linked, provided that line segments exist between them. The direction in which the link is made is determined by the handedness of the boundary. This process identifies road edges, and corrects for cases where intersecting roads do not have common vertices, but merely overlap.

This graph is not sufficient for a full drive-through, since no connection exists across road intersections. This is solved through the addition of an additional pass, which links vertices that fall within a small radius of each other. Provided this radius is approximately the width of a road then suitable links across intersections are made. Some additional heuristics are applied to this algorithm to restrict its operation to intersections, and to ignore joins in otherwise straight roads.

A link from node A to node B, where the two nodes are within the cutoff radius, can be created provided they conform to the following restrictions:

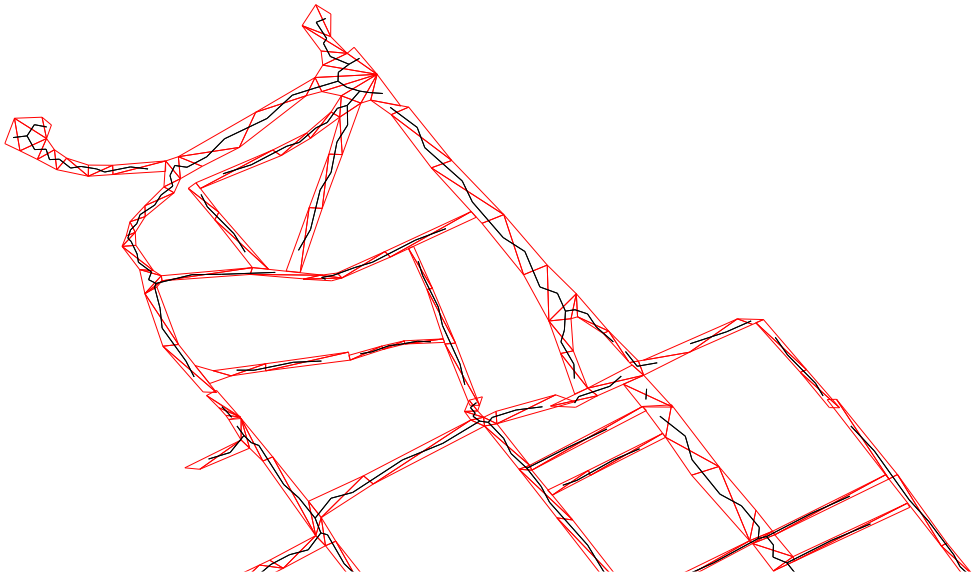
- There must exist an edge v_a originating at A and an edge v_b originating from B which do not have opposite directions. In practice, the angle between the two vectors must not be in a range around 180°. This eliminates links across the middle of straight roads.
- The edge from A to B must not be in the opposite direction to any edge originating from B. This prevents the creation of sink nodes from which no driver can escape.

This approach performs adequately, outlining straight sets of roads, and adding additional paths across intersections. The number of paths it can create across intersections can be substantial in some cases, providing a maze of paths in a small area which make implementing a realistic driving algorithm difficult. In addition, a constant cutoff radius large enough for wide roads can create problems for narrow intersections. Adjusting the algorithm to dynamically adjust the cutoff radius to the width of the road, and limiting the maximum number of edges per node in the graph (4 edges proves to be a suitable limit) creates a more suitable representation of the road system. The results of this algorithm are shown in Figure 4.

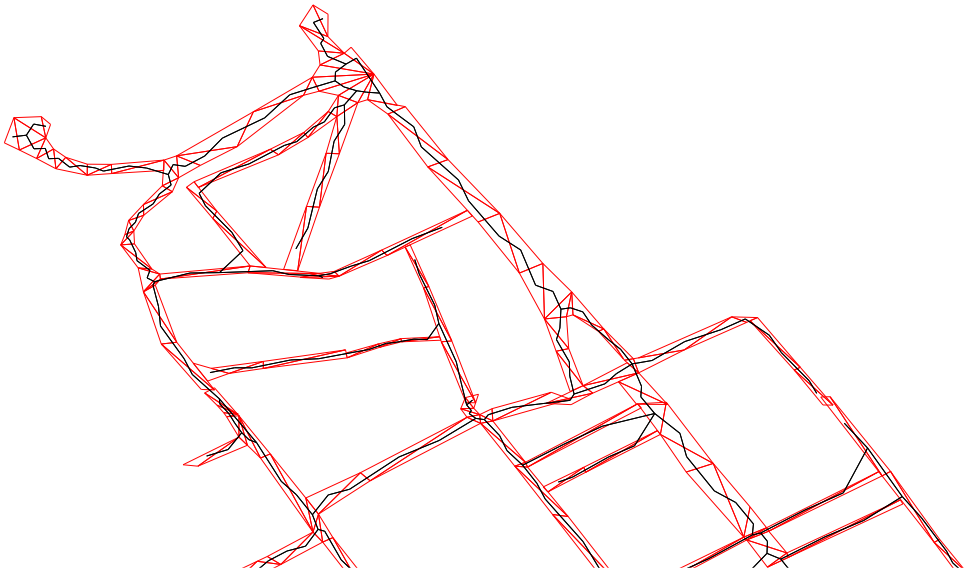
7.1.2 Touring the City

The car in the virtual world runs a control routine which reads in a graph produced from the digitized road data and uses this to navigate though the virtual world in a realistic manner.

The graph is traversed so as to explore the world as thoroughly as possible. Splines are used to control the motion of the vehicle, especially when following the jagged zig-zag path produced by the centre following algorithm. A sequence of points from the graph are used as control points for the spline. The distance between control points is used to normalize the step taken between frames,

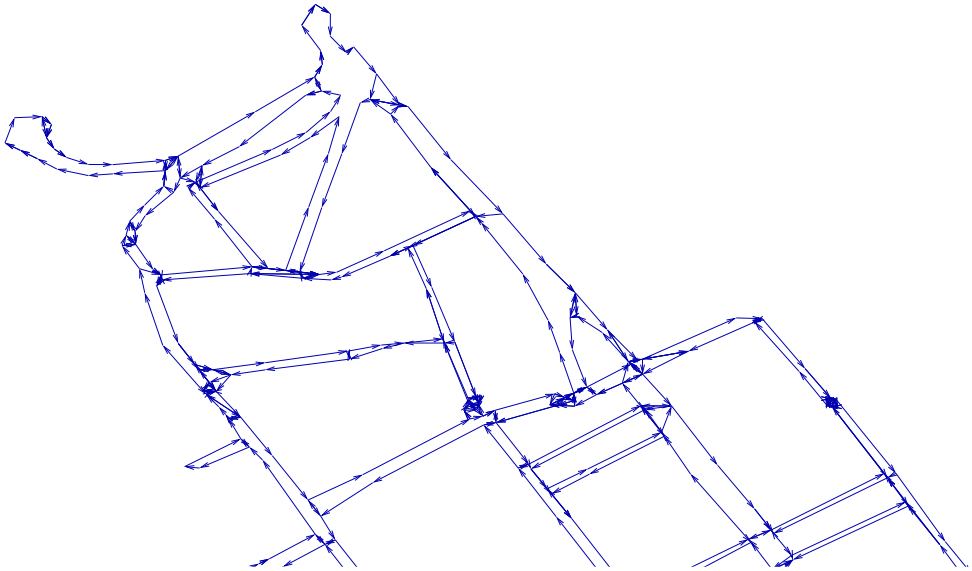


(a)

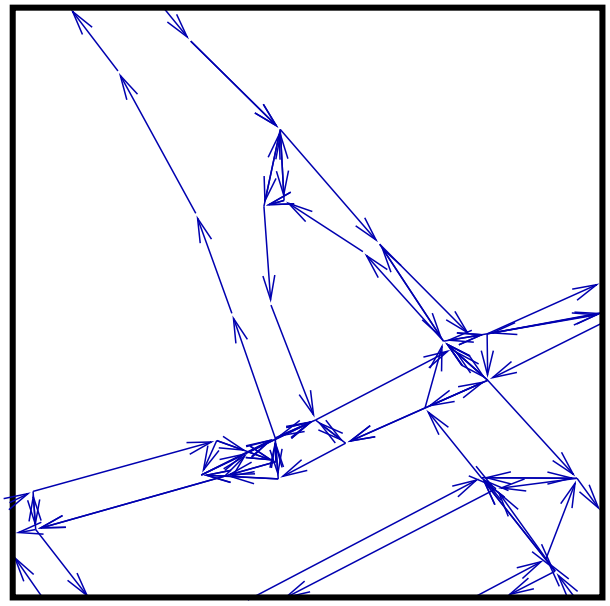


(b)

Figure 3: (a) Paths produced by connecting adjacent triangles. (b) Path after application of heuristic.



(a)



(b)

Figure 4: (a) Directed graph produced by the edge tracing algorithm. (b) Enlarged version, showing details of an intersection.

to maintain a constant speed. Spline control fails at cul-de-sacs, where an about turn is required. At these points abrupt changes in direction would occur. Limiting the amount of rotation allowed for each time step caters for this situation, and allows a slow pan of the landscape before travel is resumed.

The graph is marked to indicate which points have been traversed so that unexplored areas can be preferred. A random choice is made when paths of equal weight exist. The marks are adjusted for each point in the sequence for the current spline, with large weights being added in front of the vehicle, and most removed at the back. This prevents doubling back on the path and circling at cul-de-sacs, and gives greater weight to unexplored avenues.

Velocity of the virtual vehicle is controlled by the curvature in the path at each time step. The general principle is to accelerate on the straight and slow down around corners. Maximum and minimum speeds keep these values within acceptable limits.

Occasional flaws can be seen where the splines cut corners, but in general the result is very effective. Examples of the path generated are illustrated in Figure 5.

7.2 Creating virtual worlds with multimedia content

The format selected for providing virtual worlds linked to other forms of information is VRML, because of its association with the World Wide Web, and ability to link to information across the Internet. Both VRML 1.0 and VRML 2.0 are produced, the latter providing facilities to program more complex behaviour in response to the user's actions.

The VRML code for objects is produced indirectly, via a conversion program from the files used with the RhoVeR system. The multimedia content for each object is provided at a URL associated with the name of the object. This URL and a short description of the object are automatically added to each VRML file during the conversion process.

Most VRML browsers allow the user to click on a particular object which brings up the short description of the object. Clicking again will start up a World Wide Web hypertext browser which will load the page specified by the associated URL. The various multimedia items associated with the object are placed on, or linked to, that page.

8 Doing It "Right"

While the process described previously is technically sufficient to rapidly extrude an entire city, there are several practical aspects to the process which affect the quality of the final product.

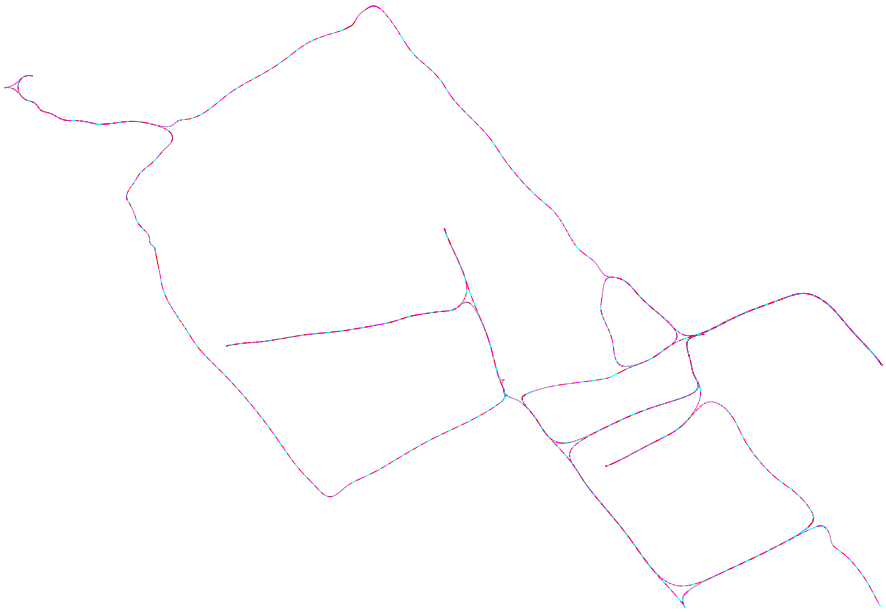
8.1 Selecting a height for the building

The problem of setting the heights of the buildings has no immediately obvious solution. The maps give the horizontal dimensions of the building but no value for the vertical dimension, apart from the height of the terrain at the building site. Initially, height values are chosen so that the appearance of the virtual buildings matched the model in the mind of the person performing the digitising. This appears to work relatively well, as may be expected when the buildings in question are relatively well known landmarks (or a structure in which one has worked for a number of years). Problems are noticed when combining the separate buildings into a single virtual world. It seems that buildings with substantial horizontal dimension are given additional vertical dimension in the mind of the viewer, and so the vertical axis is exaggerated compared to other sleeker buildings.

An alternative approach involves capturing the aspect ratio of the building from the building itself. This can easily be done from a perpendicular view of a face of the building, and measuring the relative horizontal and vertical dimensions of the building. Since only aspect ratio is required, the absolute magnitude of these values is unimportant. This ratio can then be used to set the height of the building, relative to the length of the base given on the map.

The use of aspect ratio measurement provides correct values for the relative heights of the buildings. The substantial horizontal scope of the virtual world dominates in this case, and the

(a)



(b)

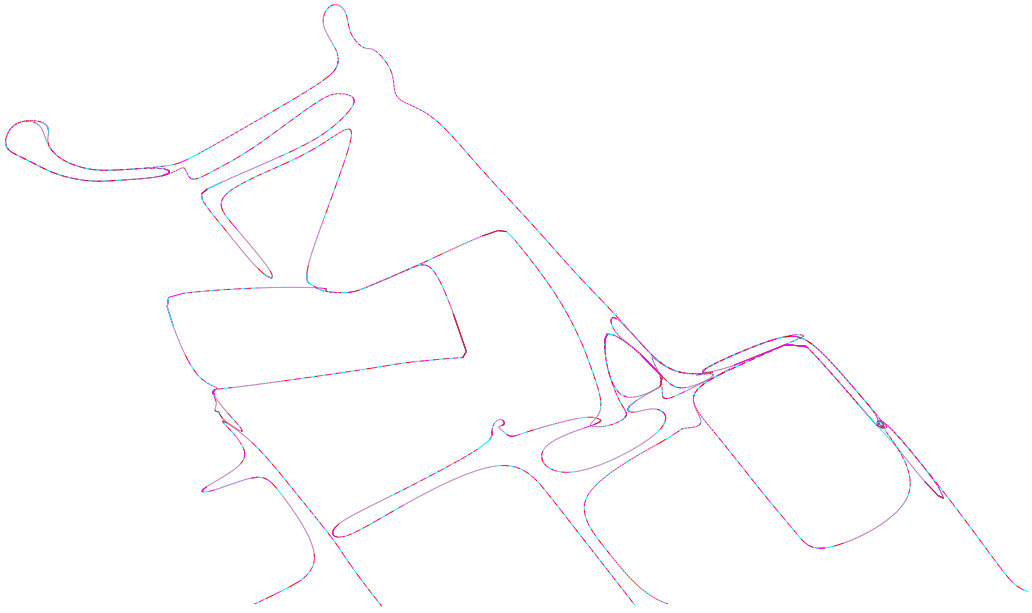


Figure 5: (a) An example of the path followed, using a map produced by the centre following algorithm. (b) An example of the path followed, using a map produced by the edge tracing algorithm.

buildings appear relatively insignificant. For the final version of the virtual world, the heights of the buildings is exaggerated by factors of 1.5 to 2, relative to their level of significance, to produce a more satisfying viewing experience.

8.2 Previewing the building

Due to time constraints, the extrusion tool does not contain a built-in previewer for the objects under construction. This proves superfluous once the functionality to export to VRML and RhoVeR is provided. Either of these systems is more than capable of rendering the object at any stage of construction in all details required for the final virtual world.

The advantage of using a virtual reality toolkit such as the RhoVeR system is the ability to customize the preview application. It is a simple matter to set up a refresh control to reread the shape description of an object, and link this to the regeneration of the file by the extrusion tool. Thus the effect of changes can be quickly and easily seen.

8.3 Doing the "Right Thing"

A major component of the task of constructing the objects in the extrusion tool is supplying the values for each face of the polyhedral object. Parameters such as height, colour, texture map, and active texture area need to be supplied in each case. An important factor in simplifying- and enhancing the speed of- the digitising process is providing appropriate values for these parameters.

In this regard extensive use is made of coherence in the supply of data. For example, the previous height supplied for a face is used as the default for the next face. This is extended by supplying the height of an overlapping segment if such exists. As mentioned previously (section 4.1.2) a similar process is applied to the selection of the heights for the triangle vertices.

Automatic extrusion of complete simple buildings requires only three points for the rectangular outline, the fourth being inferred from the other three. This simplifies matters by reducing the amount of data that must be captured, and allows the capture interface to overlap with that for entering a triangular polygon. Initial building heights are those used for the previous building. The use of a keystroke to initiate the building process as opposed to the selection of a button on the graphical user interface allows multiple buildings to be captured quickly without the need for many long distance mouse movements between drawing window and building control panel.

The height of roads and terrain can also be captured more efficiently make making use of the 1-1 relationship between position and height of a point. Not only can initial height be set from that from an overlapping vertex, but any corrections to the height of the vertex of one triangle can be echoed back to any other overlapping vertices of other triangles. This reduces the need to hunt backward through the triangle list to track down mismatches in the object.

9 Completing Construction

The first portion of code for the extrusion tool was laid on the 1st of June 1997. By the end of the 30th June 1997, the city was standing and walkthroughs of the virtual environment and access to multimedia content via the virtual representation was possible. On the 1st of July 1997 the city went on display at the National Festival of the Arts in Grahamstown. The entire city was built in 30 days.

Some views of the final product can be seen in Figure 6.

Since then, the extrusion tool has been used on a number of other rapid prototyping projects:

- A mockup of a new building (designed, but not constructed in reality) has been modelled in virtual reality. In this case the interior detail of the building is also included in the model. Textures, captured from existing edifices, are used to provide both interior and exterior detail. It took two hours to produce the model of the one of the floors of the building.

(a)

(b)

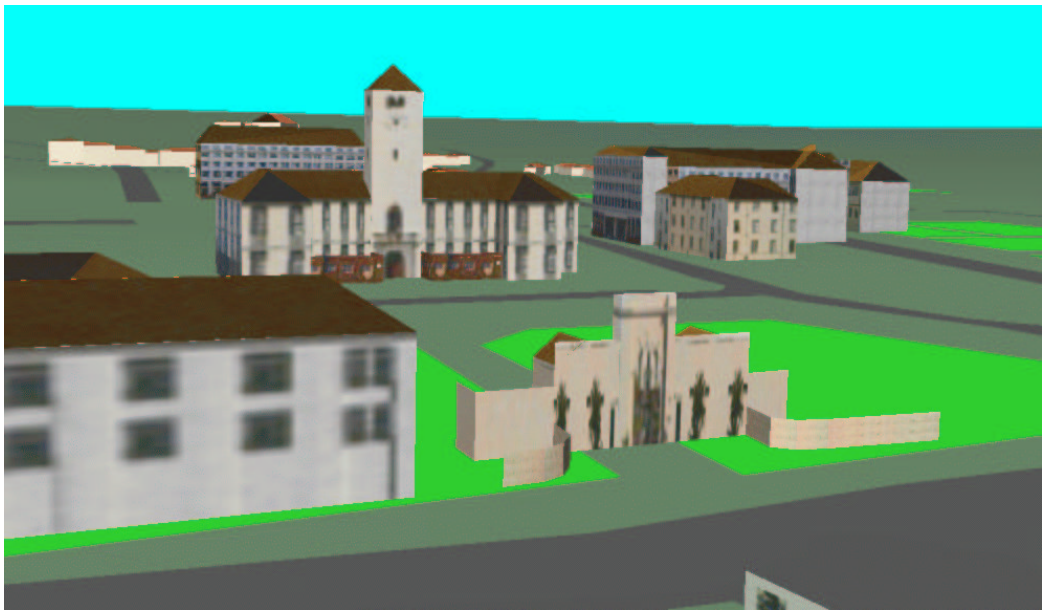


Figure 6: (a) View of Grahamstown from behind the 1820 Settlers Monument, the principle festival venue. (b) View of the Rhodes University campus.

- A chemical plant, part of a design project in another department, was successfully modelled in an afternoon. The VRML model is used to examine access routes between buildings, particularly with regard to accommodating large vehicles. A video sequence generated while exploring this model was used during the presentation of the design.

10 Building Bigger and Better

The addition of a hierarchical relationship structure to the system is currently being explored to improve object reuse, and simplify the creation of complex models. This is implemented by extending the primitives in the system (wall segments and triangles) to include other models produced by the extrusion tool. This approach has several benefits:

- Buildings can be combined, by placing the detailed building models on a larger, but less detailed base. This improves building layout, and obviates the need to combine several high resolution maps.
- Objects can be produced at various levels of detail. The detail of the model can be reduced by pruning the hierarchy tree, or substituting complex sub-objects with simple generic ones.
- It becomes feasible to populate the building shells with furniture and more complex internal structure.
- A variety of building blocks can be used as construction primitives.

Current limitations in the system which impair the illusion of reality involve the placement of buildings within the terrain. Terrain is captured manually which involves considerable effort, and is error prone. Frequently there are mismatches in positions and heights of supposed adjacent polygons. As a result only limited numbers of terrain features are captured, and even these are relatively low resolution. Most buildings are positioned without the benefit of surrounding terrain, creating the effect that they are floating in space.

Building placement is also limited by the details provided by the map. The contours are given at intervals of 2 metres. This distance is a substantial portion of the height of a building. Since buildings are captured independently of the terrain, the advantages of automatically calculating position from the height of the terrain (which is interpolated between contour levels) is lost.

Future developments will look at wrapping terrain around buildings, generating terrain polygons from point samples on the map, and smoothing the faceted terrain by using the control points to generate a spline mesh.

11 Making it Possible

I would like to thank the following for their assistance with this project:

- Lindsey Sheard who followed this work from the beginning and acted as guinea pig for various versions of the extrusion tool, and whose suggestions directed much of its development.
- Michael Preston whose project supplied the motivation for this work, and who contributed much time, effort and resources into building the VRML worlds.
- Luis Casanueva, Mike Rorke, Adrian Smith and Austin Poulton who assisted with the digitising process.
- The members of the MCF of the Distributed Multimedia Centre of Excellence who provided many of the suggestions on the nature of the exhibit.

References

[Ban96] Shaun Bangay, James Gain, Greg Watkins, Kevan Watkins, *RhoVeR: Building the Second Generation of Parallel/Distributed Virtual Reality Systems,* First Eurographics Workshop on Parallel Graphics & Visualization, Bristol(UK), 26-27 September 1996.

[3DL97] 3DLabs, *Virtual TOKYO*, available via the WWW as:
http://www.planet9.com/earth/tokyo/index.htm,
July 1997.

[Par97] Paragraph International, *Virtual Home Space Builder*, available via the WWW as:
http://www.paragraph.com/products/i3dfamily/vhsb/,
May 1997.

[Web97] Roger Webster and class, *Millersville University VRML Virtual Campus*, available via the WWW as:
http://zansiii.millersv.edu/work2/campus.dir/,
March 1997.

[Boy97] Mark Boyns, *SDSU 3D*, available via the WWW at:
http://teamball.sdsu.edu/~boyns/vrml/,
July 1997.

[Sti96] Stilla U, Michaelsen E, Lütjen K, *Automatic extraction of buildings from aerial images.* In: Leberl F, Kalliany R, Gruber M (eds) Methods for extracting and mapping buildings, roads and other man-made structures from images. IAPR TC-7 Workshop, München: Oldenburg, 1996.

[Lew96] Rick Lewis, *Generating Three-Dimensional Building Models from Two-Dimensional Architectural Plan*, Technical Report CSD-96-902, Computer Science Division, University of California at Berkeley, May 1996.

[Ofe97] Eyal Ofek, Erez Shilat, Ari Rappoport and Michael Werman, *Multiresolution Textures from Image Sequences*, IEEE Computer Graphics and Applications, Volume 17, Number 2, March-April 1997.

[Goe96] Martin Goëbel, *Projects in VR*, IEEE Computer Graphics and Applications, Volume 16, Number 1, January 1996.

[Goo95] David Goodfellow and Aaron Aubin, *Virtual World Simulation: Bermuda*, School of Urban and Regional Planning, University of Waterloo, Ontario, Canada, available via the WWW at:
http://www.fes.uwaterloo.ca/u/gbhall/research/bermuda/bermuda.html,
Summer 1995.

[Axi98] AxisMedia Inc., *3D Skyline Models Catalog*, available via the WWW at:
http://www.skypoint.com/~jperry/axismedia.html,
July 1998.

[Aic95] Oswin Aichholzer, Franz Aurenhammer, David Alberts and Bernd Gartner, *A Novel Type of Skeleton for Polygons*, Journal of Universal Computer Science, 1 (12), 1995, 752-761.