Implementing an Augmented Scene Delivery System

James E. Mower

Department of Geography and Planning, ES 218
University at Albany, Albany, NY 12222 USA
jmower@albany.edu

Abstract. This paper addresses the core issues confronting the design and use of an augmented scene delivery system (ASDS). An augmented scene is a real-time, interactive, symbolized, perspective view of an environment that serves as a graphical index to an underlying spatial database. It allows a person in the field to interpret and navigate through the environment without reference to an external map. Augmented scenes will enable users with underdeveloped map use skills to effectively interpret and analyze their environment in professional, educational, and recreational contexts. This paper discusses an ASDS implementation that acquires imagery from a user-controlled webcam. It focuses on issues of data sampling and representation.

Introduction

Cartographic mapping can be understood as a series of transformations between views of geographic entities. Most of these transformations (modeling the earth as a solid, projecting its surface to a plane, etc.) are handled by the cartographer or geographic information system (GIS). One important transformation left to the user is map orientation. Faced with a map and a view of the terrain in the field, the user tries to understand the environment with the map as its model, gradually developing a cognitive transformation with elements of 3-dimensional rotation, perspective scale foreshortening, and hidden surface construction. Under the best viewing conditions, users differ sharply in their ability to interpret maps and orient them to their environment. As conditions degrade, even the most experienced user finds it difficult to determine her position or to find a path.

Most users find that they can reduce some of the cognitive burden of orientation by aligning the map with their direction of view, perhaps in association with a compass. A paper map can be rotated by hand; a computer-generated display with a simple coordinate transformation. The latter may be of further help if it can model the landform in perspective from an arbitrary position. Still, users are left with two independent views: that of the symbolic map and the raw visual scene.

Mower [8] discusses a theoretical and technical framework for removing the distinction between these views, unifying them into a new type of map, the augmented scene. An augmented scene places symbolic information directly over the user's current view of the environment, allowing her to query geographic features within the view through graphic selection. Employing a video camera as an imaging device, the user acquires an image of the landform. The augmented scene delivery

P.M.A. Sloot et al. (Eds.): ICCS 2002, LNCS 2331, pp. 174–183, 2002.

system (ASDS) applies the viewpoint coordinates and the user's viewing parameters to build a perspective model of the landform registered to the image.

This paper discusses a prototype implementation of an ASDS, focusing on landform elevation data structures and sampling issues. To reduce the complexity of the implementation, the author has substituted a user-controllable webcam for a hand-held video camera, eliminating the need for a GPS, digital compass, and digital inclinometer. All observations occur from a fixed location with full freedom of movement in horizontal and vertical viewing angles.

Previous Related Work

The introduction of affordable and sufficiently powerful computers to the cartographic community in the 1980s allowed for the dynamic overlay of acquired imagery on perspective maps [5]. The recent introduction of high-resolution position finding and geodetic angle acquisition devices has enabled cartographers to take this work a step further through the real-time overlay of map symbols onto acquired perspective imagery of the landform. Recent work in augmented reality has established a baseline for this research. At present, most augmented reality research emphasizes indoor, short-range applications [1]. Of these, Dorai and others [3] describe image and model registration techniques for industrial applications. Drasic and Milgram [4] discuss techniques for overlaying a user-controlled pointer on captured stereoscopic video imagery to locate objects in the 3D world for robotic navigation within lab settings. Schutz and Hugli [10] and Gleicher and Witkin [6] apply pattern recognition techniques and user intervention to improve model and image registration for short range applications. Recent papers by Neumann and others [9], Azuma and others [2], and Kim and others [7] discuss techniques for tracking objects for augmented reality applications in interior and exterior environments, generally over distances of up to several hundred meters. This project extends the usage of image acquisition, registration, and overlay operations to real-time perspective views of the general environment for geographic applications over long distances (to the user's horizon).

Implementation Design

The Camera, Mount, and Image Server

The camera, its mount, and control software were purchased from Perceptual Robotics, a supplier of user-controllable webcam systems. The camera is currently mounted on the roof of Mohawk Tower, a 24-story dormitory on the campus of the University at Albany. A Pentium II computer controls its operation from the floor below in the University's Climate Observatory. Camera and mount commands are sent through direct serial cable connections. A coaxial connection feeds video from

the camera to the computer. The user controls camera, mount, and video functions through requests to the computer's web server that are formatted as URL strings.

ASDS Software Functions

The author has developed the current ASDS software implementation, *UrHere*, as a Microsoft Windows application written in C++. After loading a digital elevation model (DEM), the system creates a perspective viewing transformation that renders the DEM from the point of view of the camera. The area extending from the viewpoint to the horizon in all viewing directions will be referred to as the viewshed. A scene is defined as a portion of the viewshed rendered with respect to a fixed field of view, horizontal viewing angle (azimuth), and vertical viewing angle (altitude).

The user can select one or more geographic feature sets for labeling. Features are organized by theme (schools, businesses, landform features, etc.) and identified by their name, projected world coordinates (plus elevation), and other fields, including a URL specific to each feature. Any feature that occupies a point in the camera's current scene (modeled as a frustum) is a candidate for labeling. A user can change the orientation of the camera using one of three methods. First, she can elect to center the viewing plane on a given feature by clicking on its name in the feature database. Second, the user may enter viewing angles that move the camera horizontally (in azimuth) and vertically (in altitude). Azimuth angles are entered in degrees relative to true north; altitude angles are in degrees relative to 0 on the horizontal plane. The third method allows the user to center the camera on an arbitrary pixel in the current scene by picking it with a mouse click. The image coordinates of the pixel are employed to construct azimuth and altitude values that center the camera on its associated world coordinates.

Once the orientation of the camera has changed, features within the current feature sets having world coordinates intersecting those of the scene will be displayed as symbols at their projected locations, overlain on the returned camera image. The user can elect to render symbols as icons or pushbuttons. If the latter option is chosen, selection of a button launches a web browser that navigates to a feature-specific URL. All of the pushbutton URLs point to a single web server (independent of the camera hardware server) that returns a "home page" for each feature. Typically, the home page contains links to on-site and off-site sources of information. On-site sources can satisfy simple requests for text retrieval or more sophisticated text and graphic requests from a local internet map server such as ArcIMS or MapXtreme. Sample home pages are currently limited to text retrieval of attribute data. Figure 1 shows a cropped image of an augmented scene centered on the New York State Museum building, positioned approximately 7 km to the east of the camera. Its symbol is rendered as a pushbutton.

Fig 1. A feature symbol rendered as a pushbutton. The camera is viewing the New York State Museum in Albany, NY, USA, located approximately 7 km from the camera

To ensure consistent operation of the camera hardware, the hardware controller reboots each night, subsequently zeroing the azimuth and altitude drive motors. However, extended daytime use can cause the motors to "drift" slightly from their true bearings. *UrHere* provides a software registration procedure to fix drift. On the selection of this procedure, the system acquires an image at the current azimuth and altitude. It then overlays iconic symbols representing features in the scene that are members of a registration feature set. Features in this set are easily identifiable on the camera image in clear weather. To register the camera to ground coordinates, the user clicks a feature icon and then the icon's representative pixel on the image. The registration procedure determines the azimuth and altitude offset angles between the two locations in image space. Subsequent camera orientation requests add the offsets to compensate for drift in both the horizontal and vertical planes.

The registration procedure, as well as the procedures that aim the camera and place symbols on the acquired image, rely on the underlying surface model to provide a world frame of reference. The current model employs a triangulated, irregular network (TIN) with sampling density that decreases with distance from the camera viewpoint.

An earlier, variable density grid cell model was tried and rejected for this project. The structure of both models and their associated characteristics are addressed in the following section (Figure 2).

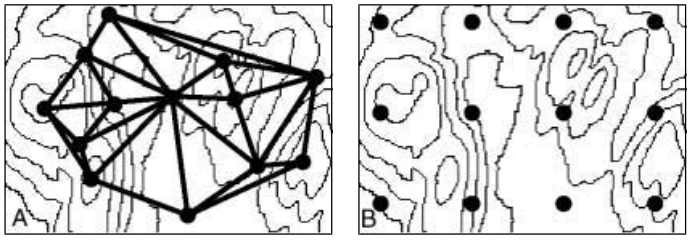

Fig 2. In A, the surface is sampled as a TIN at significant points. In B, the surface is sampled as a regular grid

Surface Data Structures

Rendering Perspective Scenes from Sampled Elevations

A mapping application that proposes to label features in a given scene must be able to model the world to its visible boundaries, determined by the elevations of the most distant features in the scene that intersect the plane of the viewing instrument (its "line of sight"). Here, feature elevation is measured relative to the surface of the sphere (the instrument horizon) with a radius equal to that of the earth plus the height of the viewing instrument. Since the instrument horizon is curved, the elevation of an intersecting feature must increase with distance from the viewpoint. Equation 1 provides the elevation of intersection (h) as a function of the radius of the instrument horizon (r) and angular distance from the viewpoint (θ). This calculation models the earth as a sphere and disregards atmospheric refraction. It returns the elevation of intersection in the same units provided for the radius.

$$h = (r / \cos(\theta)) - r \quad \textbf{(1)}$$

In Figure 3, a feature with an elevation of 500 meters above the instrument horizon will intersect its line of sight at a distance of approximately 80 km from the viewing instrument.

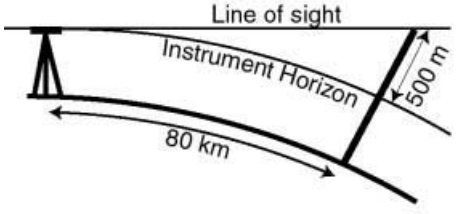

Fig 3. A feature rising 500 meters above the instrument horizon has an apparent relative elevation of 0 meters at a distance of approximately 80 km

For the current project, the camera, from its elevation 144 meters above sea level, can see peaks in the Catskill Mountains that are approximately 1200 meters above sea level and 55 km from the camera. Assuming the use of available DEMs with a regular grid sampling resolution of 10 meters in easting (x) and northing (y), modeling the surface to a 55 km radius from the viewpoint would require over 95 million elevation samples—clearly an amount too large for practical rendering operations. Two alternatives are immediately apparent: 1) to model the surface at a fixed, lower resolution, or 2) to vary the sampling resolution with distance from the viewpoint. Regardless of the applied sampling technique, the ground area covered by a rendered pixel in a perspective viewing model increases linearly with distance from the viewpoint. Given a half-angle θ for field of view and a distance d from the viewpoint, the width w of the viewing frustum at depth d is expressed in Equation 2.

$$w = 2dTan\theta \quad \textbf{(2)}$$

Assuming a camera viewing angle of 22° and a frame buffer 500 pixels in width, the width of a pixel representing a part of the scene at 1 km from the camera would represent approximately .8 meters on the ground. At 25 km its width would represent 19.4 meters. And at 55 km it would represent 42.8 meters.

Sampling Resolution

If sampling resolution is held constant, the number of samples per pixel will similarly increase with distance from the viewpoint. If the sampling resolution were fixed at 20 meters, for example, a pixel rendering the surface at 1 km from the viewpoint would cover .04 samples, at 25 km .97 samples, and at 55 km 2.13 samples. Pixels at the limit of the viewshed would over-sample a 1 sample per pixel ideal by a factor of 2, but pixels near the camera would be under-sampled by a factor of 25, enabling over-generalized renderings of foreground surfaces. To avoid this, a variable resolution sampling scheme would keep the per-pixel resolution constant with distance, neither under- nor over-representing any particular area.

A variable sampling technique adjusts the local resolution to account for perspective scale reduction with distance from the viewpoint. In the grid model, constant sampling resolution is maintained among bands surrounding the viewpoint. Resolution decreases at band borders as a stepwise linear function of distance from the viewpoint. Local irregularities cannot be accounted for without interrupting the integrity of the grid. In the TIN model, sampling resolution decreases continuously with distance from the viewpoint, modulated by local surface variability.

The variable resampling technique for grids partitions the viewshed into bands extending away from the viewpoint as square, non-overlapping bands (Figure 4). Elevations within a given band are resampled from a high resolution DEM (representing ground truth) as a linear function of the band's mean distance from the viewpoint. The effect of this technique is to create a viewing frustum that is partitioned into multiple regions, each characterized by successively lower sampling resolutions away from the viewpoint.

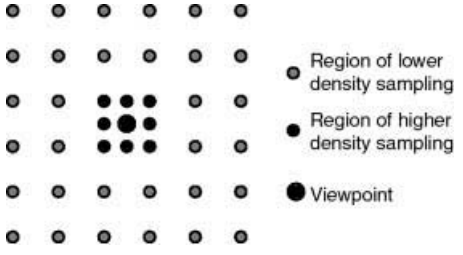

Fig 4. Variable resolution grid sampling

The grid model was tried but quickly abandoned for this project. Although resampled grids are relatively easy to construct, they have several disadvantages. This project works with a viewshed containing highly irregular features near the horizon—

the Catskill mountains to the south, the Taconics to the east, and the Adirondacks to the north. The resampling function was adjusted to ensure that such features were adequately rendered. The furthest visible feature, a peak in the Catskills approximately 55 km from the camera, needed to be rendered with a sampling of 43 meters per pixel to maintain a 1 sample per pixel ideal over a 500 pixel-wide frame buffer. Even if the entire data set were resampled at this resolution (thereby abandoning 1 sample per pixel representation in other parts of the viewshed) the entire viewshed would require over 5 million samples while still seriously underrepresenting terrain near the viewpoint. This analysis suggested that resampling techniques must account for surface variability as well as perspective scale reduction to enable efficient surface renderings.

Unlike the grid model, samples in a TIN elevation model generally represent topologically significant points on a surface. For this project, the procedure GridToTin was developed and implemented to extract such points from a grid cell DEM. The procedure iteratively subdivides a grid into triangles until no sample is further than a predetermined tolerance from the plane of its enclosing triangle.

Procedure GridToTin

Starting with a square grid of elevation samples and a base tolerance,

Form 2 triangles from the 4 samples at the grid corners. Add the samples to a list of significant samples and the triangles to the processing stack.

While there are triangles on the stack,

- Extract a triangle from the stack and find the equation of its plane.
- For each sample falling within the triangle's interior,
  - Compute the local tolerance as the base tolerance weighted by the sample's distance from the viewpoint.
  - Find the orthogonal distance of the sample to the plane.
  - If the distance exceeds the local tolerance,
    - Add the sample to the list of significant samples
    - Subdivide the triangle into 3 new triangles sharing a vertex at the saved sample
    - Add the new triangles to the processing stack

Discussion of Procedure GridToTin

The procedure becomes increasingly insensitive to deviations in sample elevations from their interpolated planar values as distance increases from the viewpoint. For rendering purposes, significant features such as peaks, ridgelines, and slope breaks can be retained out to the horizon with an appropriate base tolerance value and local tolerance function. A local tolerance value would be considered appropriate if 1) it

does not create a triangle that covers an area less than that covered by a pixel and 2) if the further subdivision of a triangle would result in any of the child triangles sharing the shading value of its parent. The first criterion is identical to the 1 sample per pixel suggested for the grid model. The 2nd is more difficult to quantify, however. It ultimately depends on the color resolution of the rendering engine and the display controller. For this project, the base tolerance was set empirically by examining its effect on the selection of points near the horizon. The local tolerance was computed by multiplying the base tolerance by the distance of the sample to the viewpoint.

The expected running time of the procedure increases with the number of gridded elevation samples, the variance of the samples, and decreasing base selection tolerance values. Worst-case performance occurs when no sample within the interior of any triangle is within the local tolerance distance of its representative plane. This maximum grid partitioning is given in Equation 3 where T is the resulting number of triangles, r is the number of rows in the grid and c the number of columns.

$$T = 2(r - 1)(c - 1) \quad \textbf{(3)}$$

In practice, T will only equal the maximum partitioning if the local tolerance function returns zero for all distances from the viewpoint or if the elevation variance is very large.

At any given time during execution, the triangles on the processing stack represent mutually exclusive regions. Because such triangles can be processed independently of and simultaneously with one another, GridToTin can be adapted to a MIMD environment using a master/worker model. The master distributes triangles on the stack to available worker processors having read-only, shared access to the original gridded elevation data. If a worker finds a significant interior sample, it writes the sample to a local list, subdivides its triangle, and copies the resulting children to the master triangle-processing stack. After the stack is empty, each local sample list is written to a global list on the master.

Since most of the execution time will be spent processing triangles, the expected speedup for the MIMD version would be directly related to the number of available processors. Workload will remain balanced as long as the number of triangles on the stack exceeds the number of available workers. The only sequential bottlenecks occur when workers copy their child triangles to the master stack and when their significant sample lists are combined at the end of processing.

The expected running time of GridToTin is not very important to the fixed viewpoint prototype application since all of the viewsheds are modeled from a single TIN. In the mobile application, however, new TINs must be generated every time a shift in the user's field position extends the horizon or results in insufficient ground resolution near the viewpoint. To minimize the weight, complexity, and cost of a mobile ASDS, it would be reasonable to offload viewshed construction to a location-based service (LBS) provider. A GridToTin implementation would run on a server, returning a relatively small TIN to the user. A MIMD application would not be out of the question in such an environment.

Following the application of GridToTin, the selected samples are triangulated with *Triangle* [11] using the Delauney option.

The *UrHere* Implementation

The author wrote UrHere as a C++ program for the Microsoft Windows NT and 2000 environments using Visual C++ 6.0 and DirectX 8.0.

The elevation data for this project originated from grid cell DEMs produced by the New York State Department of Environmental Conservation (ENCON) at a horizontal ground resolution of 10 meters. The file data structure corresponds to that of the US Geological Survey 1:24,000 regular grid series. To simplify the TIN resampling procedure, the author first reformatted elevations across the ENCON data files covering the research area into 10 km by 10 km grids. As the TIN resampling procedure searches for significant samples, elevations are recomputed to account for curvature of the earth and the effects of atmospheric refraction with regard to the viewpoint.

To test the registration of surface world coordinates with image coordinates, the author used a GPS with post-processing to collect the coordinates of geographic features that occupy relatively small areas on the surface and that are clearly visible in camera imagery. With post-processing, the accuracy of the positions were estimated to be on the order of 3 meters or less from their true position. All surface and feature coordinates refer to Universal Transverse Mercator (UTM) zone 18 north, North American Datum (NAD) 1927.

Results

From the top of Mohawk tower on the campus of the University at Albany, the viewshed of the camera extends approximately 55 km to the south and somewhat less in other directions. To model the viewshed elevations, samples from ENCON DEMs at 10 meter horizontal resolution over a 90 km by 90 km region were converted to a TIN. Using a 2-meter vertical tolerance at the viewpoint that increases linearly to 133 meters at a distance of 55 km from the camera, the TIN renders the viewshed model with 39,329 triangles, far less than the maximum partitioning of approximately 162,000,000 triangles. The author found through inspection that the selected vertical tolerance range provided an adequate frame of reference for feature labeling. Running in Windows 2000 on a Pentium 4 at 1.3 GHz and rendering with an NVIDIA Quadro2 Pro accelerator rated at 30 million triangles per second, the model required 5 seconds to load (real time). Although profiling data were not available for this project, new scene renderings in the viewshed were displayed with no visible delay. The images in Figures 5 show a peak, High Point, at a distance of approximately 24 km from the camera, at magnification factors of 2X and 12X. Both images have been cropped from a larger scene. The original 24-bit color images have been converted to gray scale and reduced to 75% of their original dimensions.

Fig. 5a. High Point at 2X magnification

Fig. 5b. High Point at 12X magnification

Conclusion

This project has demonstrated the feasibility of using webcams from fixed locations for perspective mapping. Future work will extend the fixed model to a mobile platform, enabling a user to create viewshed maps on the fly. Such an application would benefit from recent work in wearable computing equipment, digital compasses and inclinometers, and location-based service technology.

References

1. Azuma, R. (1997). A Survey of Augmented Reality. In: Presence, Teleoperators and Virtual Environments. Vol. 6, No. 4, pp. 355-385.
2. Azuma, R., B. Hoff, H. Neely III, R. Sarfaty, M. Daily, G. Bishop, V. Chi, G. Welch, U. Neumann, S. You, R. Nichols, J. Cannon (1998). Making Augmented Reality Work Outdoors Requires Hybrid Tracking. Web address: http://www.cs.unc.edu/~azuma/ARpresence.pdf. Proceedings, 1st Int.Workshop on Augmented Reality, San Francisco.
3. Dorai, C. G. Wang, A. Jain, and C. Mercer (1998). Registration and Integration of Multiple Object Views for 3D Model Construction. In: IEEE Transactions on Pattern Analysis and Machine Intelligence. Vol. 20, pp. 83-89.
4. Drasic, D. and P. Milgram (1991). Positioning Accuracy of a Virtual Stereographic Pointer in a Real Stereoscopic Video World. In: SPIE Stereoscopic Displays and Applications II. Vol. 1457, pp. 302-312.
5. Faintich, M. (1986). Digital Cartographic Data Bases: Advanced Analysis and Display Technologies. In: Proc., 2nd Int. Symp. on Spatial Data Handling, Seattle, pp. 600-610.
6. Gleicher, M. and A. Witkin (1992). Through-the-Lens Camera Control. In: Computer Graphics. Vol. 26, No. 2, pp. 331-340.
7. Kim, J., H. Kim, B. Jang, J. Kim, D. Kim (1998). Augmented Reality Using GPS. In: Proceedings SPIE, Stereoscopic Displays and Virtual Systems V, Vol. 3295, pp. 421-428.
8. Mower, J. (1997). The Augmented Scene: Integrating the Map and the Environment. In: Proceedings, Auto-Carto 13, Seattle, pp. 42-51.
9. Neumann, U., S. You, Y. Cho, J. Lee, J. Park (1999). Augmented Reality Tracking in Natural Environments. In: Int. Symp. on Mixed Realities, Ch. 6, Ohmsha Ltd. & Springer-Verlag, Japan.
10. Schutz, C. and H. Hugli (1999). Augmented Reality Using Range Images. In: Proceedings SPIE, Stereoscopic Displays and Virtual Reality Systems IV, Vol. 3012, pp. 472-478.
11. Shewchuk, J. R (1996). Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangulator. In: 1st Workshop on Applied Computational Geometry (ACM), Philadelphia, pp. 124-133.