Some Remarks on Automated Sketch Generation for Mobile Route Descriptions

Jörg Baus, Andreas Butz, Antonio Krüger, Marco Lohse
Universität des Saarlandes
Dept. of Computer Science
PO box 151150
66041 Saarbrücken, Germany
[baus,butz,krueger,lohse]@cs.uni-sb.de

ABSTRACT

The next generation of route description systems will not only give way information at the beginning, but also during the way-finding task. The information will be brought to the user by stationary and mobile displays (e.g. info kiosk or PDA). Especially graphics play an important role to convey way finding information. In this paper we explain an approach, to generate such graphics that are tailored to the use on mobile devices.

Keywords

hybrid user interfaces, navigation, resource adaptivity, user adaptivity

1. INTRODUCTION

The next generation of route description systems will not only give information at the beginning, but also during the way finding task. The information will be brought to the user by stationary and mobile displays (e.g. info kiosk or PDA). For a better understandingof how these systems will have to be designed we developed a hybrid navigation system that is able to adapt to various restricted resources during the generation of such route descriptions in different modalities. This implies dealing with various technical resources on one hand, and taking into account a user's limited cognitive resources to decode and understand the presentation on the other hand. In order to cover all types of limited resources the system consists of three major parts:

1. An information booth, the 3D-graphics workstation, where we a virtual walkthrough of the environment is presented by a virtual presenter accompanied with spatial utterances and metagraphics which complement each other.
2. An outdoor navigation system, using a small laptop with head mounted display, actually a clip-on for glasses, a GPS system to determine the users actual position and an electronical compass to track the users orientation in the environment and a digital map of the environment.
3. An indoor navigation system based on strong infrared transmitters mounted at the ceilings of the building and small PDAs as presentation devices for simple sketches of the environment received via infrared.

Smart Graphics '01 March 21-23, 2001, Hawthorne, NJ

One problem is how to generate graphics for all these different output media and in this paper we will concentrate on how graphical way-finding information can be generated for mobile devices in the indoor scenario.

2. TOWARDS ROUTE DESCRIPTIONS FOR MOBILE DEVICES

Route maps are a common descritpion for routes. They show a path from one location to another and can be an effective way for visualizing and depicting directions. Maps depict the route as a sequence of turning points connected by lines. Following [5] the structure of route maps are essentially the same as the structure of route directions. Like route directions, route maps could be devided into segments containing starting and ending landmarks, orientation and actions.Therefore, a route will be divided into segments in order to be described. Each segment consists at least of four parts belonging to different categories: *starting point, reorientation (orientation), path/progression* and *ending point.* Paths are continuous, the ending point of one segment serves as the starting point for the next. Segments are mostly separated by changes in direction, because in the presentation we have to communicate a change in direction clearly.

A segment represents a partial view of the environment by integrating different kinds of knowledge, e.g. knowledge about landmarks. Landmarks in navigation are used to identify choice points, where decisions are made, identify destination points and providing route progress and influence expectations, providing orientation cues for homing vectors and suggesting regional differentiating features ([3]). They are the memorable cues that are selected along a path and also enable the user to encode spatial relations between objects and paths, enhancing the development of a cognitive map of a region. A landmark may be any element in an environment that is external to the observer and that serves

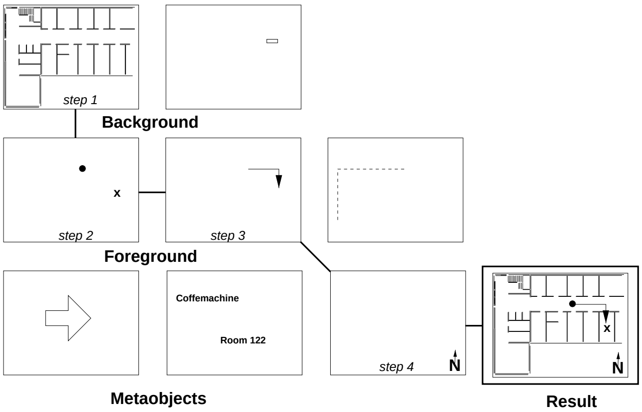

Figure 1: Some graphical components of way descriptions

to define the location of other objects or locations. In our proposed model, we represent knowledge about paths in a graph. The graph also contains references to the objects in the environment. Knowledge about the objects is stored in an annotated 3D-modell. With this information about paths through the environment and the objects along paths, we are able to visualize 3D walkthroughs and overview-maps at the 3D information booth. Such a visualization of a virtual walkthough is a special case of a complete route description. Complete route descriptions describe the entire route at once and are not suitable for mobile devices carried with the user while walking through the building on the strength of different resource limitations.

- Mobile devices, like PDAs, have some technical limitations the presentation has to survey. They have small displays with limited colors, little computational abilities and a limited bandwidth to receive infrared transmissions.
- The user's resources are limited. They depend e.g. on his walking speed, cognitive load and his familarity with the environment. So while walking the user might be under time pressure or distracted by other people in the building.
- Limitations of the system's sensory information. The accuracy of tracking the user's position and orientation may vary during the way-finding task.

The main idea is to generate incremental route descriptons for mobile devices, where the user can compare graphics and environment directly instead of relying on a recollection of the complete way description. Since not only the technical resources are limited, but also the cognitive resources of the user, the amount of available information has to be reduced to a minimum. It is important to present only the relevant details and thus focussing the viewer's attention on the important parts of the way description, i.e. the trajectory itself.

The next section explains how we genrate route maps for PDAs using the same data about the environment as for generating 3D-walkthroughs.

3. GENERATING MOBILE ROUTE DESCRIPTIONS

Automatically generated graphics for mobile szenarios have to be tailored to the various technical, cognitive and sensory constraints that we described earlier in this paper. The graphics generation process itself can be divided down in two parts: the content selection and the rendering of the graph. Earlier work in intelligent graphics design (e.g. in [6]) suggested to interweave both steps, to reflect effects of the rendering result in the content selection phase. For the sake of efficiency we decided to separate both steps. In order to do so, we partioned the way finding task in several subtasks.

The generated graph consists of two basic graphical elements: the background and the foreground. Whereas the background shows parts of the surroundings (e.g. the floor map for navigating through a building), it includes also important landmarks. The foreground is build of graphical elements that include: the actual position of the user, the actual target of the way description, arrows that indicate changes of the direction, a line that depicts the path that is still left to go and the path that lies already behind the user. Those parts of the path can be depicted in different styles. Additional metagraphics help to improve the understanding of the overall graph. We use textual annotations to label depicted objects and, if the system is completely unsure about the user's orientation, a compass rose to indicate the external frame of reference. Or if the system does exactly know where the user is and in which direction he is looking, the map can be rotated in order to match the actual walking direction of the user. Another possibility is to display only an arrow indicating the walking direction like a compass.

Figure 1 summarizes all these different parts of a graphical way-description and shows how a result can be generated by composing these subparts to a graph. In the example the user wants to find his way to a particular room in the building and he is already very close to his target. First of all the system includes a floormap as background element in the graph (*step 1*, see next section for more details, how this can be achieved automatically from a 3D-model of the building). After that (*step 2*) the system renders the actual position (circle) and the target (cross). Then information about the path is added in form of an arrow (*step 3*).

Now assumed that the system does exactly know where the user is, but is lacking information about his orientation. This implies for the system that it has to make sure that the user understands the external frame the system is referring to. This can be accomplished e.g. by adding a compass rose indicating North in the graph (*step 4*).

The automation of these steps consists of several subproblems that have to be solved. First of all the system must decide how the graph has to be composed. i.e. which components of the way description of figure 1 are used and which not. This process depends on the ressource limitations of the situation the user is in. We use a ressource adaptive planning approach to consider different constraints on the outcoming graph (for more details see [1]). Whereas some tasks can be highly complicated, especially the annotation of the graph with textual objects, (see [2, 4]), others are easy to solve, like the visualisation of the remaining path. Since the trajectory is already stored in segments, these can be used directly for visualization (e.g. as straight or dotted

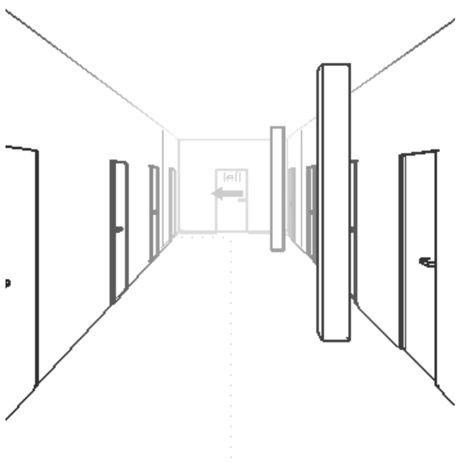

Figure 2: Sketch rendered from an egocentric point of view.

line or arrow).

In the following section of this paper we will concentrate more on the generation of the 2D-sketches that are derived directly from the 3D-model of the building. This does not only inlcude top-view floorplans but, as shown in figure 2, also views from within the scene.

4. SKETCH GENERATION

Handdrawn black and white sketches are commonly used to convey route descriptions. Humans are not distracted by very abstract sketches, which leave out unimportant details of the surroundings (like rooms, which are not entered) or include metagraphical elements (like arrows and text). This is not the case for photorealistic computer-generated images.

We describe an algorithm to obtain a black and white sketch from a 3D modell in real time. Furthermore we are able to generate a vector graphics representation of the sketch, which has three advantages.

- Since our 3D modells are mainly buildings and building interiors, which often underlie special architectural constraints, we can assume that the rendered sketches mostly consist of long, straight line strokes and fewer short ones. As Vector graphics only needs to store two 2D vertices for each of these line segments, they are consuming less memory and communication bandwidth, which is important in our szenario, where graphics are transferred to mobile devices.
- After being transferred, vector graphics can be translated, rotated and scaled without loss of image quality. Again, this point is important because of today's low-resolution displays of mobile devices. These vector transformations are also much faster to compute than 2D bitmap transformations.
- Vector graphics enable us to transmit data incrementally. This feature helps to minimize the apparent transmission time, which is the delay between the start of transmission and the first element displayed (see Figure 3).

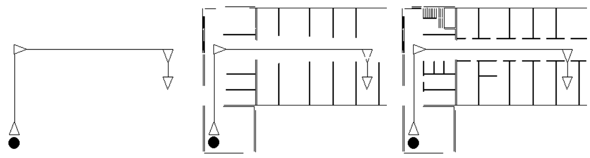

Figure 3: Incremental transmission of vector graphics: First an arrow depicting the path is shown, then the floor plan is transmitted in several steps, longest lines first.

4.1 Algorithm

Our algorithm consists of two parts. First, a sketch of the polygonal 3D modell is rendered. Then, a vector graphics representation of this sketch is generated.

4.1.1 Sketch Rendering

Since we only want to generate informative sketches of buildings and paths in buildings, we use a simple sketch rendering algorithm which does not produce artistic or stylistic shading. An *edge* is a line segment which is shared by two polygons. In a preprocessing step of the algorithm, all edges are classified either as *soft* or *hard* edges. An edge is hard, if the angle between two adjacent polygons is bigger than a certain threshold, otherwise it is classified as soft. Hard edges are important for the shape of the modell and are viewpoint independent. The *silhouette* of a rendered 3D modell is viewpoint dependent and consists of all soft edges which are between adjacent front facing and back facing surfaces.

A sketch is now generated in two steps: First, the whole modell is rendered as usual but in the background color and without shading. Then, all hard edges and all soft edges which lie on the silhouette are rendered [1]. Edges on the silhouette are found by simple computing two dot products for each soft edge: the dot product of the viewing vector (the vector from the eye to the scene) and the normal vector of each of the adjacent polygons of the edge. If these values differ in sign, the edge is on the silhouette and will be rendered. These computations are necessary whenever the viewpoint or one of the modells in the scene changes. We found that these computations do not affect total rendering time too much.

4.1.2 Vector Graphics

Now that black-and-white sketches can be rendered, one possibility to get a vector graphics representation of the image is to find the longest line segments in the generated 2D bitmap. Since this approach is either very time consuming or introduces a lot of unnecessary vectors, it is not suitable for our task.

A simple modification of the sketch rendering algorithm allows us to determine which pixels of the rendered image belong to the same edge of the 3D modell: every edge which

[1]This step requires the ability to render edges with a z-Buffer offset to prevent them from being partly hidden by polygons.

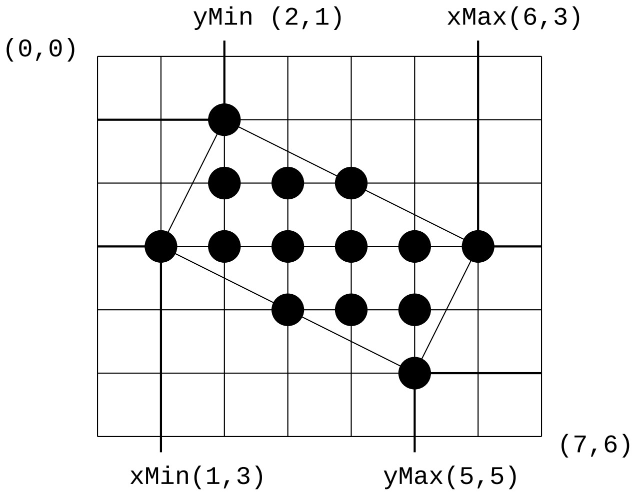

Figure 4: Extension of the vector graphics algorithm. Instead of storing only two 2D points per color(`yMin` and `yMax`), two additional 2D points (`xMin` and `xMax`) are stored to compute the thickness of the vector.

will be rendered is assigned a unique color (which differs from the background color). This is possible, because most systems support more than 16 million different colors and therefore even large modells can be handled this way. Now, all pixels in the rendered bitmap which have the same color belong to the same edge. And that is why all pixels of a particular color must lie on a line. Knowing this, a vector graphics representation can easily be derived with following algorithm:

- Scan all pixels from left to right and top to bottom.
- For every pixel, whose color is not the background color, check if this particular color has already appeared before. If not, store the current x and y coordinates as starting point and ending point of the vector together with its color; otherwise, store the current x and y coordinates as (new) ending point of the vector.
- If all pixels have been scanned, generate a vector for each color using the starting and ending point.

Of course, edges might be occluded by other parts of the modell and cut into pieces. For each of these pieces, a separate vector must be created. This leads to the first extension:

- Every time a new ending point is set, check if the old ending point lies *too far away* (which is a well defined parameter). If yes, store the current starting and ending point as vector and insert the current x and y coordinates as new starting and ending point.

Also, lines in the 2D bitmap might be thicker than one pixel. This thickness can be derived by using and updating four points per vector instead of two. Figure 4 shows an example for this case. A third extension would be to postprocess all generated vectors:

- Delete all vectors which are *too short* (again, a well defined parameter).

4.2 Implementation

The implementation of these two steps is straight forward. For the sketch renderer, two lists of edges need to be generated, one for hard and one for soft edges. For each soft edge, adjacent polygons and eventually their normal vectors need to be stored; the color of all edges needs to be accessible for the second step.

The vector graphics algorithm is implemented using a hash table; the color of the pixel is used as key and the starting and ending point are stored as values. Vectors are generated by finally enumerating all entries in the hash table.

Since none of the techniques require direct access to the rendering pipeline, these algorithms can be implemented with any graphics API; we chose Java 3D.

5. REFERENCES

[1] A. Butz, J. Baus, A. Krüger, and M. Lohse. A hybrid indoor navigation system. In *IUI2001: International Conference on Intelligent User Interfaces*, New York, 2001. ACM.

[2] T. Rist, A. Krüger, G. Schneider, and D. Zimmermann. Awi: A workbench for semi-automated illustration design. In *Proc. AVI '94*. ACM Press, 1994.

[3] M. Sorrows and S. Hirtle. The Nature of Landmarks for Real and Electronic Spaces. In C. Freksa and D. Mark, editors, *COSIT 99: Spatial Information theory: cognitive and computational foundations of geographic information science*, pages 37–50. Springer, 1999.

[4] T. Strothotte. *Abstraction in Interactive Computational Visualization: Exploring Complex Information Spaces.* Springer, 1998.

[5] B. Tversky and P. U. Lee. Pictural and Verbal Tools for Conveying Routes. In C. Freksa and D. Mark, editors, *COSIT 99: Spatial Information theory: cognitive and computational founda tions of geographic information science*, pages 37–50. Springer, 1999.

[6] W. Wahlster, E. André, E. Bandyoppadhyay, W. Graf, and T. Rist. Wip: The coordinated generation of multimodal presentations from a common representation. In A. Ortony, J. Slack, and O. Stock, editors, *Communication from an Artificial Intelligence Perspective: Theoretical and Applied Issues*, pages 121–144. Springer-Verlag, Berlin, 1992.