# Tour Generation for Exploration of 3D Virtual Environments

Niklas Elmqvist*
INRIA/LRI, Univ. Paris-Sud

M. Eduard Tudoreanu†
University of Arkansas at Little Rock

Philippas Tsigas‡
Chalmers University of Technology

## Abstract

Navigation in complex and large-scale 3D virtual environments has been shown to be a difficult task, imposing a high cognitive load on the user. In this paper, we present a comprehensive method for assisting users in exploring and understanding such 3D worlds. The method consists of two distinct phases: an off-line computation step deriving a grand tour using the world geometry and any semantic target information as input, and an on-line interactive navigation step providing guided exploration and improved spatial perception for the user. The former phase is based on a voxelized version of the geometrical dataset that is used to compute a connectivity graph for use in a TSP-like formulation of the problem. The latter phase takes the output tour from the off-line step as input for guiding 3D navigation through the environment.

**CR Categories:** H.5.2 [Information Systems]: User Interfaces; I.3 [Computer Methodologies]: Computer Graphics

**Keywords:** tour generation, navigation aids, navigation assistance

## 1 Introduction

Spatial understanding of the structure of a 3D virtual world is vital for a user to be able to navigate and solve tasks efficiently, yet this understanding is exceedingly difficult to attain as the worlds become increasingly complex and increasingly transient [Chittaro and Burigat 2004; Darken and Peterson 2001; Darken and Sibert 1996]. New advances in technology allow designers to increase the visual realism (and thus also the visual complexity) of their 3D worlds to hitherto unseen levels, exacerbating this problem. Furthermore, many worlds are today dynamically created for a specific purpose, such as in response to a search query or as the result of a computation, and will exist only for the duration of the interaction. Our users must then be regarded as tourists in these worlds, lacking specific knowledge about the environment they are exploring, yet in need of solving their tasks as rapidly as possible. There is an obvious conflict in this state of being.

In this paper, we propose to bridge this gap between the prevalence of complex and unknown 3D worlds and the user desire to navigate and traverse these worlds effortlessly using computer-generated grand tours through the 3D environment. In essence, instead of forcing the user to expend precious time learning an environment, we devise a method to let the computer explore the environment and extract the vital paths prior to presenting the environment to the user. We then use this information to "hold the user's hand" as he or she traverses the world. Depending on the level of interaction desired by the user, we can impose constraints on the path, speed, deviation, and camera direction as the user moves through the world. Furthermore, even if the user wants to navigate freely, the path information can be used to smooth the user's ride, avoid jarring collisions (if collision detection is implemented) or disorienting ghosting through walls (if no collision detection), and ensure that the user visits all targets.

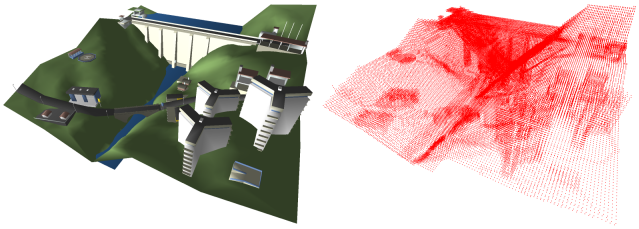

**Figure 1:** *Voxelization process for a complex 3D environment.*

The application domains of this method are many and varied: it can be used for visual storytelling when introducing a new 3D environment, familiarizing a 3D modeler or designer with an unknown or half-forgotten project, presenting all the relevant information in a visualization space, and more. The off-line computation step is designed to be as efficient as possible, providing acceptable tour information with a minimum of time investment. The on-line component can be configured to either be unobtrusive, merely nudging the user in the right direction, or take full control of the user's movement through the world.

## 2 Related Work

The need for effective navigation through a three-dimensional computer environment arises for any environment larger than what can be seen from a single viewpoint. This situation forces the user to rely on a mental representation of spatial knowledge, often called a *cognitive map* [Chase 1986; Tolman 1948]. While navigation can be a challenging task even in the physical world, the absence of many sensorial stimuli in the virtual world compounds the problem even further [Chittaro and Burigat 2004].

*Wayfinding* is typically defined as a cognitive aspect of navigation with the purpose of planning and forming strategies prior to executing them, i.e. where the actual navigation is not the goal of the interaction but the means to solve some specific task [Darken and Sibert 1996]. The wayfinding task is conducted on the user's cognitive map, and thus it is clear that if the user lacks an accurate mental representation of the environment, performance will suffer.

### 2.1 Spatial Design

One approach to improve wayfinding is to organize the virtual environment in a way that promotes understanding and orientation, in essence making it easier for the user to construct an accurate cognitive map. Due to the similarities with navigation in physical space [Vinson 1999], we can leverage existing research from urban planning, geography, and psychology. For example, Darken and Sibert suggest a number of design guidelines for organizing a

*e-mail: elm@lri.fr
†e-mail: metudoreanu@ualr.edu
‡e-mail: tsigas@cs.chalmers.se


VRST 2007, Newport Beach, California, November 5–7, 2007.
© 2007 ACM 978-1-59593-863-3/07/0011 $5.00

virtual environment to facilitate the acquisition of spatial knowledge [Darken and Sibert 1996], further extended in [Darken and Peterson 2001]. Similarly, Vinson [1999] argues for the importance of landmarks for navigation in a 3D world, and gives a comprehensive set of guidelines for their placement, design, and composition.

### 2.2 Navigation Aids

Visual aids can be used to great effect for improving 3D navigation. Static or interactive maps are the obvious examples. Chittaro and Burigat [2004] present an array of different compass-like navigation widgets for helping the user to find important objects and places in a virtual environment. Trails [Ruddle 2005] help users utilize previous explorations to improve their current search. Path drawing [Igarashi et al. 1998] lets the user draw an intended path directly on the 2D view of the world to aid navigation.

### 2.3 Motion Control

Another powerful class of navigational aids is motion control, i.e. different methods of traveling through a virtual environment and potentially guiding or constraining the user's movement. Bowman et al. [1997] present a taxonomy of first-person motion control techniques for manual viewpoint travel that is useful for evaluating such methods. Witmer and Singer [1998] discuss the value of having control for navigation efficiency and presence.

Guided navigation techniques augment the user's spatial knowledge with additional information. Wernert and Hanson [1999] present a taxonomy of assisted navigation, and also discuss a "dog-on-a-leash" approach to guidance through a 3D world. This approach is similar to the "river analogy" introduced by Galyean [1995], where the viewpoint is tethered to a vehicle following a path through the virtual environment and some degree of control is retained by the user. The guidance technique presented in this paper builds on both the river and dog metaphors, yet supports variable interaction to a higher degree.

Another notable technique is the virtual guide of Chittaro et al. [2003] that the user must follow actively; the guide's path is also automatically computed using an algorithm operating on a 2D occupancy matrix similar to the tour generation algorithm in this paper, but our method can handle any general 3D environment and not just one-floor buildings.

Finally, constrained navigation techniques essentially assume full control of viewpoint motion, sometimes even moving the gaze of the user in the desired direction. By reducing the freedom of the user, navigation and wayfinding can be simplified, and eliminate the need for expensive features such as collision detection. Examples of this approach include that of Hanson and Wernert [1997], who employ invisible surfaces to constrain user movement, and of Hong et al. [1997], who provide an interactive navigation where care is taken to avoid collisions with the environment.

The Way-finder system presented by Andújar et al. [2004] algorithmically computes an exploration path through a 3D environment. This algorithm is based on a voxelized version of the 3D world, just like the tour generation algorithm presented in this paper, but employs another method to compute a cell and portal graph for use with a backtracking tour generator, whereas our algorithm builds disjoint visibility subsets and performs TSP computations on the resulting connectivity graph.

CubicalPath [Beckhaus et al. 2001] uses a similar off-line voxelization of the 3D world and then employs dynamic potential fields for guiding exploration at run-time.

## 3 Automatic Tour Generation

The objective of the automatic tour generation algorithm is to build a grand tour of a 3D world given a geometry dataset, a set of landmarks (or targets), and a starting point.

The tour should start and end in the starting point and visit all of the landmarks in the world. By enforcing the tour being a loop, we support repeated traversal of the tour for familiarization, but this is an optional constraint. Beyond these simple requirements, we can add a few more: the generated tour should be "good" in some sense, and the process should be robust in the presence of inaccessible landmarks, i.e. landmarks that are landlocked and cannot be visited due to surrounding geometry.

The definition of a "good" tour is open to debate; in this work, we take it to mean a tour of as short length as possible (not necessarily optimal) that visits all landmarks as few times as possible. Furthermore, the tour should not stray outside the bounding box of the 3D world to avoid trivial (but impractical) solutions.

An important observation is that visiting a landmark in this context is equivalent to seeing it, so it is not necessary (and in fact undesirable) to pass through the same spatial location as the landmark for it to be regarded as having been visited. At the same time, our algorithm allows for specifying a maximum visibility distance, i.e. the furthest away the tour may pass a landmark in order to visit it.

Finally, the representation of landmarks is significant; in our implementation, we choose to use 3D points for simplicity, but the algorithm can support other 3D primitives as well.

### 3.1 Voxelization

Our tour generation algorithm operates on a voxelized version of the 3D world, so the initial step of the process is to transform the geometry dataset into a volume representation (see Figure 1 for an example). We first compute the bounding box of the world and enlarge it in all directions by a single voxel width to allow for the algorithm to skirt along the perimeter of the 3D world if necessary. Then we voxelize the world using incremental 3D scan-conversion.

The process of incremental 3D scan-conversion builds a volume representation of a 3D boundary representation such as a triangle mesh by iteratively scan-converting the 3D primitives into a voxel buffer. Kaufman and Shimony [1986] give algorithms for scan-converting all manners of 3D primitives; our method is based on a recursive subdivision of 3D space into an octree representation and testing the triangle against each volume using a fast triangle-box intersection test [Akenine-Möller 2001] (see Figure 2).

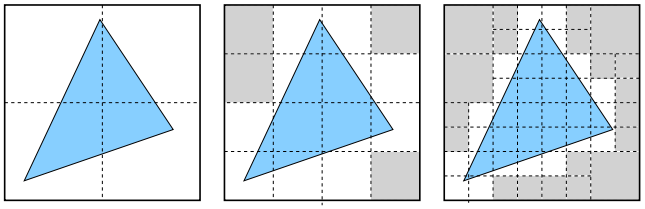

**Figure 2:** *Recursive 3D scan-conversion using an octree.*

### 3.2 Visibility Calculation

Armed with a volume representation of the 3D dataset, we can now calculate the visibility information of all voxels given the set of landmarks the tour should visit. We use an integer one-pass voxel

traversal algorithm [Liu et al. 2004] to determine if there is a clear line of sight between the current voxel and a specific landmark (this particular step must be generalized for landmarks represented as something more complex than a point). Additional constraints can also be imposed at this point; we currently ensure that the distance between the voxel and the landmark is within the maximum visibility distance, but other constraints are plausible.

Having derived the visibility information for all voxels, we then group them into disjoint subsets that we call *visibility sets* using a breadth-first search algorithm. Each set is built so that its members are contiguous and have the same visible landmarks. A special case is made for voxels with no visible landmarks; they form "zero-visibility" sets, and are necessary for connectivity in the world.

The visibility sets together form a connectivity graph specifying the general visibility structure of the 3D world. At this point, it is possible to subject the connectivity graph to an optional optimization step. Many visibility sets are redundant or useless and may be removed from the graph; examples include zero visibility leaf nodes as well as nodes whose visibility is subsumed by its neighbors. Care must be taken not to remove nodes so that the graph no longer is connected, however.

The final step of the visibility calculation phase is to identify the *border voxels* for each visibility set, i.e. the voxels that are adjacent to voxels in another set. We know that that in order to travel from one neighbor to another through a specific node, the tour will have to pass at least one voxel in each border set. Again we can optimize the problem (but this time by an approximation); our implementation identifies a single "entry point" border voxel for each neighbor by minimizing its average distance to the other neighbors in the visibility set together with its counterpart border voxel in the neighboring set.

### 3.3 Tour Generation

The stage is now set for generating the tour through the 3D environment. We use a TSP-like formulation of the problem. It is important to remember that it is not necessary to visit all of the visibility sets in the connectivity graph (as in traditional TSP), just enough to cover the all of the landmarks. Thus, we can model our problem as what is known in the literature as a *Generalized Traveling Salesman Problem* (GTSP), where the $n$ nodes in the undirected graph $G$ are partitioned into $m$ disjoint subsets called *clusters*, and where it is sufficient to visit only one node in each cluster.

Given that GTSP reduces to TSP when $m = n$, GTSP is clearly NP-hard, so in our implementation we do not aim for an optimal solution of the problem. Instead, we use the border voxels computed in the previous phase to reformulate the connectivity graph as a *border graph* with the border voxels as nodes and the interior of the visibility sets as edges. The length of the edges connecting border voxels can either be found using a shortest-path algorithm such as $A^*$ or simply approximated by the Euclidean distance. See Figure 3 for an example of a simple border graph for 3D world represented by four visibility sets ($A$, $B$, $C$, and $D$) and with the paired border voxels as white boxes.

Using this representation, we can now employ a standard TSP heuristic [Cormen et al. 1990] based on computing the minimum spanning tree of the connectivity graph and deriving a Hamiltonian cycle from it. Our unique modification is the added termination condition to quit when all landmarks have been visited. In Figure 3, with a starting point in visibility set $A$, it is easy to see that a tour would proceed in the following order: $A,C,B,C,D,C,A$.

Finally, the last step of our tour generation phase is to derive a detailed voxel-level tour given the connectivity graph tour computed in the previous step. We do this by iteratively moving along the graph tour, calculating the shortest path from the current position to any border voxel in the next visibility set to visit. Each such instance is constrained to the particular visibility set, cutting down the search space considerably.

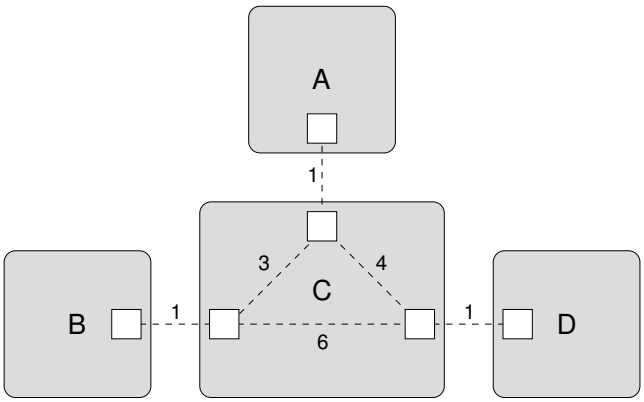


**Figure 3:** *Border graph representation of a 3D world for four visibility sets ($A,B,C,D$) connected via border sets. Edge numbers represent distances.*

## 4 Guided 3D Navigation

Our method for 3D navigation guidance is designed to both help users discover all of the specific landmarks in the world, as well as helping them to build an accurate cognitive map of the world as a whole. To achieve the former, we employ a grand tour of the world, either created manually by a human designer or generated automatically by an algorithm such as the one described above. To achieve the latter, we allow users to retain some control over their movement along the tour, seeking to engage them as active participants in the exploration.

The technique uses a spring-like umbilical cord with which the viewpoint is connected to the grand tour. See Figure 4 for an overview. Depending on the level of interaction desired, we can impose constraints on the following properties:

- **Speed.** Movement along the tour can either be computer-controlled or user-controlled.
- **Viewpoint direction.** The direction of the camera can either be slaved to the direction of movement, fixed to follow the currently closest landmark, or fully user-controlled.
- **Local deviation.** To facilitate active participation, we allow deviations from the tour path in the navigation guidance method. Using a simple interaction technique, the user can smoothly zoom the viewpoint forward or backwards in the direction of movement to the full extent of the virtual spring.

## 5 Results

Performance measurements of the tour generation phase applied to the four different scenarios (indoor, outdoor, an abstract information landscape, and a conetree) are presented in Table 1. The measurements were conducted on a Intel Xeon 3 GHz computer with 1 GB of RAM. The main bottleneck of the algorithm is the last step, i.e. the derivation of local paths within the voxel sets. Currently, this is performed using a variant of Dijkstra's shortest path algorithm, but more complex and optimized solutions are certainly possible.

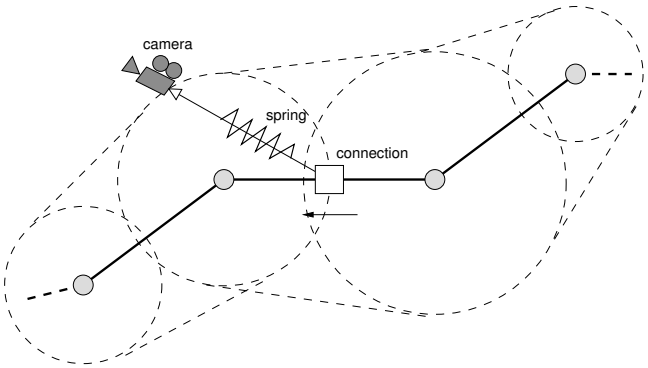


**Figure 4:** *Spring-zooming overview (the circles show the free space around each node.)*

As can be seen from the results, the visual complexity of the scene is more or less irrelevant; the voxelization phase is a very small fraction of the total time. Rather, the important metric is the degree of occlusion in the world. For the indoor scenario, the occlusion is high despite high visual complexity, resulting in fast computation. For the outdoor scenario, on the other hand, its open nature yields very large visibility sets, causing high computation time.

The voxel size can be used to somewhat control both computation time as well as memory consumption; the larger the voxels, the shorter computation time and the less memory is used. On the other hand, larger voxel size implies a less accurate volume representation, causing the quality of the generated tours to suffer.

| Scenario | Triangles | Time |
|---|---|---|
| outdoor | 558,130 | 9 minutes 4 seconds |
| indoor | 484,673 | 59 seconds |
| infoscape | 12,844 | 7 minutes 49 seconds |
| conetree | 16,576 | 2 minutes 1 second |

**Table 1:** *Off-line tour generation performance.*

## 6 Conclusions

We have presented a method for navigation guidance in the exploration of general 3D environments intended to both promote building a cognitive map of the environment as well as to improve visual search task performance. To achieve this, we compute a grand tour of the environment that visits all of its important landmarks, while allowing users to deviate from the tour. This last step is vital for making users active participants in the navigation instead of passive recipients.

## References

AKENINE-MÖLLER, T. 2001. Fast 3D triangle-box overlap testing. *Journal of Graphics Tools 6*, 1, 29–33.

ANDÚJAR, C., VÁZQUEZ, P.-P., AND FAIRÉN, M. 2004. Wayfinder: guided tours through complex walkthrough models. In *Proceedings of EUROGRAPHICS*, 499–508.

BECKHAUS, S., RITTER, F., AND STROTHOTTE, T. 2001. Guided exploration with dynamic potential fields: the CubicalPath system. *Computer Graphics Forum 20*, 4, 201–210.

BOWMAN, D. A., KOLLER, D., AND HODGES, L. F. 1997. Travel in immersive virtual environments: An evaluation of viewpoint motion control techniques. *Proceedings of the IEEE Conference on Virtual Reality*, 45–52.

CHASE, W. G. 1986. *Handbook of Perception and Human Performance, Vol II: Cognitive Processes and Performance*. John Wiley and Sons, ch. Visual Information Processing.

CHITTARO, L., AND BURIGAT, S. 2004. 3D location-pointing as a navigation aid in virtual environments. In *Proceedings of the ACM Conference on Advanced Visual Interfaces*, 267–274.

CHITTARO, L., RANON, R., AND IERONUTTI, L. 2003. Guiding visitors of Web3D worlds through automatically generated tours. In *Proceedings of the ACM Conference on 3D Web Tech*, 27–38.

CORMEN, T. H., LEISERSON, C. E., AND RIVEST, R. L. 1990. *Introduction to Algorithms*. MIT Press, Cambridge, MA.

DARKEN, R. P., AND PETERSON, B. 2001. Spatial orientation, wayfinding, and representation. In *Handbook of Virtual Environment Technology*, Lawrence Erlbaum Associates, K. M. Stanney, Ed.

DARKEN, R. P., AND SIBERT, J. L. 1996. Wayfinding strategies and behaviors in large virtual worlds. In *Proceedings of the ACM CHI'96 Conference on Human Factors in Computing Systems*, 142–149.

GALYEAN, T. A. 1995. Guided navigation of virtual environments. In *Proceedings of the ACM Symposium on Interactive 3D Graphics*, 103–104.

HANSON, A. J., AND WERNERT, E. A. 1997. Constrained 3D navigation with 2D controllers. In *Proceedings of the IEEE Conference on Visualization*, 176–182.

HONG, L., MURAKI, S., KAUFMAN, A., BARTZ, D., AND HE, T. 1997. Virtual voyage: interactive navigation in the human colon. In *Computer Graphics (SIGGRAPH '97 Proceedings)*, vol. 31, 27–34.

IGARASHI, T., KADOBAYASHI, R., MASE, K., AND TANAKA, H. 1998. Path drawing for 3D walkthrough. In *Proceedings of the ACM Symposium on User Interface Software and Technology*, 173–174.

KAUFMAN, A., AND SHIMONY, E. 1986. 3D scan-conversion algorithms for voxel-based graphics. In *Proceedings of the ACM Workshop on Interactive 3D Graphics*, 45–75.

LIU, Y. K., ZALIK, B., AND YANG, H. 2004. An integer one-pass algorithm for voxel traversal. *Computer Graphics Forum 23*, 2, 167–172.

RUDDLE, R. A. 2005. The effect of trails on first-time and subsequent navigation in a virtual environment. In *Proceedings of the IEEE Conference on Virtual Reality*, 115–122.

TOLMAN, E. C. 1948. Cognitive maps in rats and men. *The Psychological Review 55*, 4 (July), 189–208.

VINSON, N. G. 1999. Design guidelines for landmarks to support navigation in virtual environments. In *Proceedings of the ACM CHI'99 Conference on Human Factors in Computing Systems*, 278–285.

WERNERT, E. A., AND HANSON, A. J. 1999. A framework for assisted exploration with collaboration. In *Proceedings of the IEEE Conference on Visualization*, 241–248.

WITMER, B. G., AND SINGER, M. J. 1998. Measuring presence in virtual environments: A presence questionnaire. *Presence 7*, 3 (June), 225–240.