Path Planning for Complex 3D Multilevel Environments

Leonel Deusdado[*], António Ramires Fernandes[†], Orlando Belo[‡]
CCTC
Universidade do Minho - Portugal

Abstract

The continuous development of graphics hardware is contributing to the creation of 3D virtual worlds with high level of detail, from models of large urban areas, to complete infrastructures, such as residential buildings, stadiums, industrial settings or archaeological sites, to name just a few. Adding virtual humans or avatars adds an extra touch to the visualization providing an enhanced perception of the spaces, namely adding a sense of scale, and enabling simulations of crowds. Path planning for crowds in a meaningful way is still an open research field, particularly when it involves an unknown polygonal 3D world. Extracting the potential paths for navigation in a non automated fashion is no longer a feasible option due to the dimension and complexity of the virtual environments available nowadays. This implies that we must be able to automatically extract information from the geometry of the unknown virtual world to define potential paths, determine accessibilities, and prepare a navigation structure for real time path planning and path finding. A new image based method is proposed that deals with arbitrarily a priori unknown complex virtual worlds, namely those consisting of multilevel passages (e.g. over and below a bridge). The algorithm is capable of extracting all the information required for the actual navigation of avatars, creating a hierarchical data structure to help both high level path planning and low level path finding decisions. The algorithm is image based, hence it is tessellation independent, i.e. the algorithm does not use the underlying polygonal structure of the 3D world. Therefore, the number of polygons as well as the topology, do not affect the performance.

CR Categories: I.3.7 [Three-Dimensional Graphics and Realism]: Virtual RealityAnimation; I.3.5 [Computational Geometry and Object Modeling]: Geometric Algorithms;

Keywords: path finding, avatars, image based

* e-mail: leodeus@ipb.pt
† e-mail: arf@di.uminho.pt
‡ e-mail: obelo@di.uminho.pt

1 Introduction

Virtual humans and avatars are slowly becoming an intrinsic part of virtual environments. The perception of scale, for instance, is greatly enhanced when avatars are present in virtual environments. Research in agents and avatars has provided these inhabitants of virtual worlds with skills that, when combined, give rise to complex and believable behaviours. Amongst these skills, the navigation in the virtual environment is of surmountable significance. This fundamental skill requires research in topics such as the study and analysis of the models topology, collision detection techniques, path finding and planning strategies. In order to achieve these goals, it is required to transform the polygonal definition of unknown virtual 3D worlds into higher level structures which facilitate the understanding of the navigational context, and enable the efficient application of navigation algorithms.

Research is abundant in areas related to 3D navigation in virtual environments, however this is not a closed subject yet. According to [Salomon et.al. 2003], research in navigation related issues can be classified in two major categories: those that seek to understand the cognitive processes that motivate the navigation, and those that are concerned with the actual navigation in 3D worlds.

The latter category is a long running research topic where many works can be found that present, classify, apply and test different techniques with a variable degree of success. However, when confronted with unknown 3D complex virtual words with multilevel passages, common in real environments, more research is required to perform a fully automated extraction of the navigation related information.

Determining the principles that rule navigation for 3D models and environments requires the analysis of the polygon soup that defines them, and the definition of procedures capable of simulating autonomous navigation with predefined goals. Multilevel passages, such as bridges that can be crossed over or below, or buildings with more than one floor, imply more navigation options. Hence, these features represent yet another issue in navigational related information extraction.

This article addresses the fully automated extraction of navigational related information from a priori unknown arbitrarily complex 3D virtual worlds, without requiring any information about the topological structure of the 3D world.

Section 2 presents a brief overview of the related work. Section 3 discusses some issues related to path finding. Section 4 presents the new method, detailing each of the stages. Section 5 discusses the usage of the method in a hierarchical approach in conjunction with A*. Practical

results of the application of the method are presented in section 6. Conclusions are finally presented in section 7.

2 Background

Path finding deals with the search for a path according to some criteria or cost function. For performance reasons this function may actually be nonexistent and the algorithm may settle for the first path it finds. More complex solutions may attempt to minimize the distance, or incorporate different costs for certain areas of the virtual environment.

However, initially a navigation structure must be built, to enable the definition of accessible areas, waypoints, and planning of complex paths, namely in a hierarchical fashion.

The most direct method to address the path finding problem is to work directly on the geometry of the 3D world as input of the search method. However, this method imposes certain conditions on the modelling process, namely on its polygonal definition, to avoid having polygons with highly disparate sizes.
The work in [Kreylos 2005] is an example of this approach, but there are few researchers following this approach for complex 3D worlds. Most methods rely on some sort of pre-processing of the 3D world in order to obtain a higher level representation, more appropriate for navigation purposes. These methods can be divided in two main categories: those that receive as input a navigation map for known environments, and those that that are able to create the map for a priori unknown worlds.

Pre defined navigation maps
These maps can be used as tools for building graphs suitable for navigation purposes. A net of interconnected places, and the paths to follow the connections, can be previously manually defined, that allow a fast search for paths between two points in the world. This approach simplifies the algorithms and provides a degree of control difficult to obtain with automatic procedures. However it is not feasible for arbitrarily complex worlds. Furthermore, changes on the topology of the world require a manual intervention. The work in [Ballegooij and Eliéns 2001; Nitsche and Richens 2005] uses this approach.

Automatically generated navigation maps
Under this approach, the algorithm builds a hierarchical graph for a previously unknown 3D world based upon low level geometric information. This is a pre-processing stage that provides a result upon which the real time, or near real time path finding is performed.

The map construction process computes a network of paths on free and walkable spaces. An example of this approach is the work in [Arikan and Forsyth, 2001], it defines connections between pairs of points that are visible from each other. Voronoi diagrams are used for the free spaces, to generate the paths [Hoff et. al., 1999; Hoff III et. al., 2000].

Based upon a geometric description of the environment, [Lamarche and Donikian, 2004] propose a method to automatically extract the topology combining Delaunay triangulations, and computing the shortest distance between corners and walls of buildings. The resulting data is stored in a hierarchical graph, latter used for navigational purposes.

Another example of the application of this approach, seen in [Andújar et. al. 2004], is used for virtual visits to buildings such as museums. Also in [Loscos et. al. 2003] the world is decomposed in a hierarchical structure.

Some works allow for specific behaviours using the notion of potential fields. The methods are commonly based on regular grids where each cell has properties that are then used to define paths for the avatars. [Loscos et. al. 2003; Dapper et. al. 2006] are examples of pedestrian path finding using potential fields.

A large chunk of the word in this area, namely [Bandi and Thalmann 1998; Pettre et. al. 2005; Pettre et. al. 2006], comes from the VR lab from the Swiss Federal Institute of Technology, headed by Daniel Thalmann. The most recent works present a method to create a 3D navigation graph from a previously unknown world. However the method is not scalable for arbitrarily large and complex 3D worlds, consuming a large amount of resources and pre-processing time.

3 Discussion

The definition of a high level spatial representation of the underlying navigation possibilities in the 3D world, is essential for the success in path planning for previously unknown 3D worlds

Some of the work in this area has already been presented in the previous section, however none of the methods solves the problem for arbitrarily large and complex worlds in a fully automated fashion. Limitations include working only in two dimensions, which is not suitable for worlds including multilevel passages, scalability issues, and also requiring extra information to define the navigation maps.

Finding the walkable areas and creating automatically a navigation structure, based on the geometric information may provide costless collision detection, hence freeing the steps performed in real time for this task. The advantages of hierarchical spatial subdivision, for path planning purposes, combined with grid based approaches, which allow the use of the graphics hardware to help extracting information from the geometry, make it a natural option [Sturtevant 2005].

A balance must be accomplished in the definition of the hierarchy, graphs with too many nodes provide a detailed representation of the environment, but have impact on the performance (as can be seen in [Pettre et. al. 2006]), too few nodes provide a less refined representation but are more gentle on the performance.

The A* algorithm also has some issues when working in real 3D [Holte et. al., 1996; Maio and Rizzi, 1994]. The heuristic functions commonly used [Lester, 2004] do not contemplate the up direction in a particularly meaningful way [Frolich and Kullmsann, 2002]. This implies that the avatar may walk directly upwards without taking into account the feasibility or requirements of such an action.

The goal is to devise a method to divide, catalogue, and build a hierarchical structure for a soup of polygons, without any manual intervention of further information, that is able to deal with arbitrarily large and complex worlds, including environments that include multilevel passages. The method must be fully automatic and generate a sustainable number of nodes and connections to allow good performance levels for real time path finding algorithms. Such a method is presented in the next sections.

4 Spatial Subdivision Graph

The work presented in here is based upon [Ramires and Deusdado, 2006]. This previous work provides efficient conservative collision detection for unknown virtual worlds with multilevel passages. We propose an extension that is geared towards the navigation in the virtual world, namely path finding and path planning.

The simple 3D world in figure 1 will be used to exemplify and show the features of the method. In this simple world, the avatar may navigate in four levels, climb ramps, overcome small obstacles, detect collisions with the remaining objects, and is not allowed to jump from one level to the other (or the floor). The avatar must be able to find the path between any two points, where such path does exist, if necessary changing levels to do so.

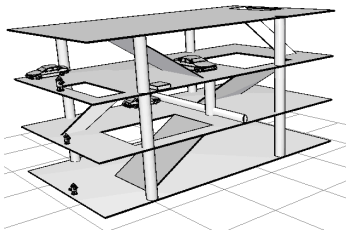


Figure 1. Simple Multilevel Virtual World

The method can be divided in five stages:

- Information gathering from the geometry to determine walkable areas
- Area classification based on image processing
- Determining interconnection zones
- Hierarchical Graph construction
- Path planning and path finding

4.1 Information Gathering

Based upon the height based segmentation process defined in [Ramires and Deusdado 2006], slices with navigation information have been identified. These slices may include ramps or stairs (either total, or partially). The algorithm obtains the minimum number of slices required for navigation purposes based on hardware Z-buffer rendering, hence it is efficient performance wise.

This is the only stage that deals with the polygon soup directly, and the result is not dependent on the topological structure of the 3D world. Performance wise, this stage is fast since only a part of the world is rendered for each slice.

The next step, once the required slices have been identified, is to identify the connection points between slices, in practice this amounts to determine which of slices contain passages between levels of the virtual building. In figure 2, where 7 slices have been found, 3 of them, namely slice 1, 3 and 5, contain passages between levels, in this case access ramps of the parking lot.

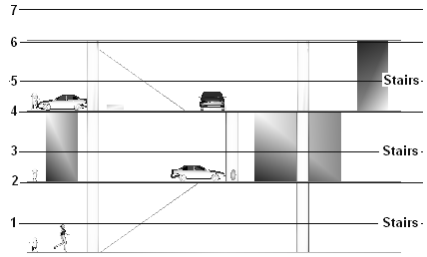


Figure 2. Slices found in the first step.

A simple algorithm was defined to locate and classify the accessible areas at different heights, based upon the height information present on the slices, generating binary images for each slice as can be seen in figure 3. The slice's information already defines what areas can be visited by the avatar, so this step is extremely simple. White areas are accessible, black are not accessible.

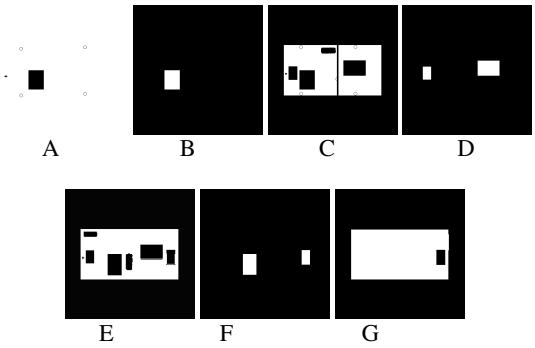


Figure 3. Images generated based on the slices information

Note that image A, from figure 3, represents the ground floor and it is therefore almost fully walkable. Also note that, in image F from figure 3, the largest ramp (on the left) has no continuity in image G, this can be seen in figure 1, where one of the ramps that leads to the top floor is actually blocked. At last, an obstacle that the avatar can jump over is not present in image E from figure 3 (the small obstacle on the third floor near the pillar in figure 1).

4.2 Area Classification Based on Image Processing

This step subdivides the space and identifies independent areas in each slice where the avatar can navigate. For instance in image C from figure 3, we can identify 2 separate areas on the same slice. In images D and F the two ramps are considered independent as well.

Image processing algorithms, available for instance in [OpenCV], allowed for the flooding of inner areas, and contour detection and area identification.

Note that these images are only used to establish the high level navigation graph, hence the missing details are not relevant. The low level navigation algorithm, i.e. the algorithm that determines navigation in each of the areas, uses the information available in the corresponding slices images, and not these binary images.

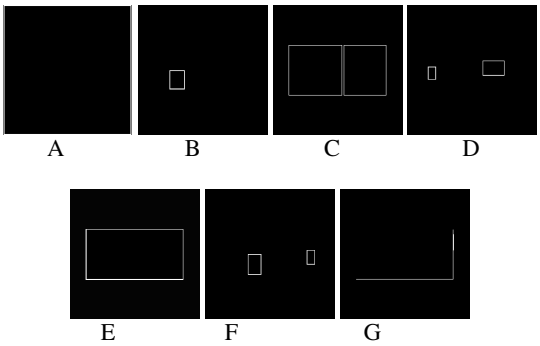


Figure 4. Result from the image processing stage to locate the areas where navigation is allowed.

4.3 Locating the interconnections

After determining the independent areas for each slice it is necessary to find the connections between slices, i.e. the passages from one slice to the next, the waypoints.

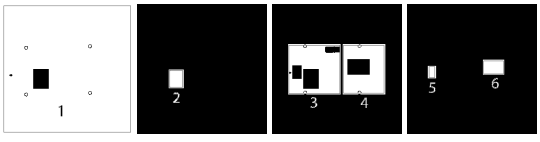
Based on the area information obtained on the previous step, for each slice, the connections between slices are searched for in adjacent slices. For instance, image B from figure 3, corresponding to slice 2 from figure 2, can only have waypoints with images A or C from image 3, respectively for slices 1 and 3 in figure 2.

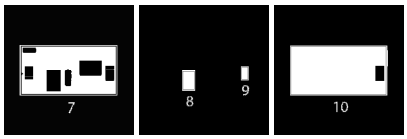
To be able to determine the presence of a waypoint between two adjacent images the heights recorded on the respective slices are compared. When the borders of the areas for different slices have common height values, and if the number of these equal height points is large enough for an avatar to pass through, then a waypoint is discovered.

4.4 The Hierarchical Navigation Graph

Once the waypoints are determined the navigation graph is easily built. At the highest level, the graph contains the identification of the areas as nodes, and the waypoints as the connections between nodes. At a lower level, for each node, the corresponding area of the slice is stored. Hence at a higher level we are able to decide if and how an avatar can move from one independent area to another independent area, and at a lower level, we decide how the avatar moves inside each independent area, this time using the slice's grid and the waypoints as the origin and destination.

For instance, considering the simple 3D world in figure 1, we obtain the following graph, as presented in figure 5.





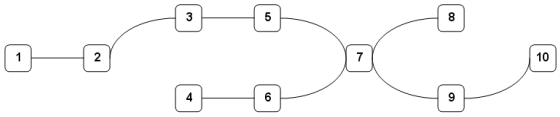


Figure 5. Navigation graph for the simple world

4.5 Navigation and path planning

The preprocessing stage provides a hierarchical graph containing all the pertinent information for navigation purposes. The start and end points may be defined by a user or by algorithms that deal with the cognitive aspects of the navigation, such as task oriented procedures. Note that each avatar should have its own start and end points, the method presented does not address groups, but rather individuals. This implies that each avatar will have its own path to follow, and that this path can be influenced by the location of other avatars.

In the examples provided, the starting point for each avatar is chosen at random in the ground floor near a point, and only the destination is common. As mentioned before, path planning will be computed for each avatar individually, hence each avatar may follow a different route.

To build the roadmap, we initially determine the node where each avatar is positioned and the node that contains the goal position. Then the higher level graph is used to determine which nodes the avatar must visit to achieve its goal. Finally for each node, path finding is used again to determine the navigation inside the node.

For the higher level of navigation the A* is used to determined the sequence of nodes to be visited. In the lower level, i.e. inside each node, A* is used again to determine the path inside each node.

5 A* Combined with the Hierarchical Graph

The preprocessing described on the previous section, and the analysis and spatial partitioning of the environment, led to an hierarchical navigation graph that can be combined with the A* algorithm to perform an efficient path planning that is scalable even when considering a significant number of individual avatars. These features are now presented in more detail.

Sustainable number of nodes and connections:

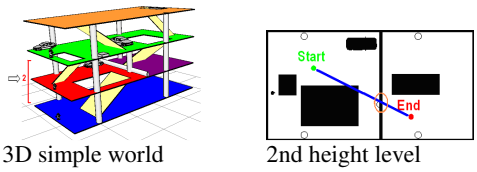
The number of nodes and connections in a graph has direct implications on the performance of path finding algorithms. Graphs with an excessive number of nodes are not applicable in situations where real time path finding is required as recognized in [Pettre et. al. 2005; Pettre et. al. 2006].

The usage of a hierarchical navigation structure enables the planning of decisions on two different levels, thereby increasing performance. At a higher level the search is performed only amongst the nodes and their connections, whereas at a lower level the search is restricted to the inside of a node. In this way the search is always performed in 2D

Fast Results:

The hierarchical structure enables fast results using the A* algorithm, that would not be achievable if the search space contained the whole world. This is more evident when considering situations where there is no path between two distinct points in the same level. In these situations the A* would attempt to explore all possibilities inside the same level just to reach the conclusion that there was no direct path between them.

This is the case in the second floor of our simple world, where there is a barrier dividing the floor in two, hence requiring a change of level to go from one side to the other, as can be seen on figure 6. Using the hierarchical approach this impossibility arises naturally at no cost.



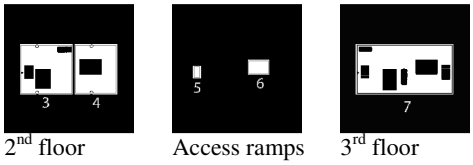
3D simple world 2nd height level

Figure 6. Example of inaccessible areas inside the same level.

Dynamic search:

The usage of the hierarchical structure provides an analysis of the navigation possibilities that divides the effort between the low level navigation and higher level path planning, reducing the search space in each situation.

For instance consider an avatar that has to be moved from a point on the second floor in region 3, see figure 7 A and B, to a point on region 4 on the same floor. The avatar must go through ramp 5 to the third floor and then go down again using ramp 6. During its movement five A* searches will be performed (regions 3, 5, 7, 6, and 4). Each will have as its goal to reach a waypoint, and in the last search the destination itself.



2nd floor Access ramps 3rd floor

Figure 7 A. independent areas found during the pre processing stage.

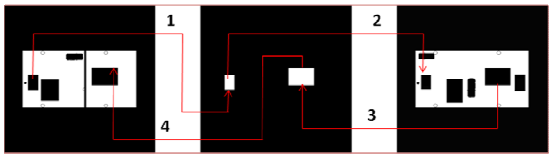


Figure 7 B. High level path.

The performance gain is due to the restriction of the search space where the avatar can move at each particular moment in time, and this search is performed dynamically, when the avatar reaches a waypoint, hence the avatar can have its path influenced in real time by the navigation status of each area.
The cost function:
Adding variable costs to the terrain may be a way to reflect the effort an avatar has to make to go up or down. The cost can also be used to implement certain behaviours in the avatars, such as avoidance of certain areas. Dynamic adaptation of the navigation cost is also possible based on the density of avatars in an area. The A* would then try to find a less crowded area. Since the paths are computed in real time, all these costs would be reflected on the path immediately.

6 Tests and Results

Two tests were performed to show the difference between our hierarchical approach and the A* used alone. Initially a 3D world that can be transformed into a 2D problem, see figure 8, was tested with both methods.

The search grid is 512x512 for each slice, however note that only a part, the area where the avatar is contained, is searchable for each node.

When considering large number of avatars, their graphical representation was simplified to a cube to avoid having the graphics performance causing a strong influence on the results. When using a smaller number of avatars an animated articulated 3D person was used see figure 10.

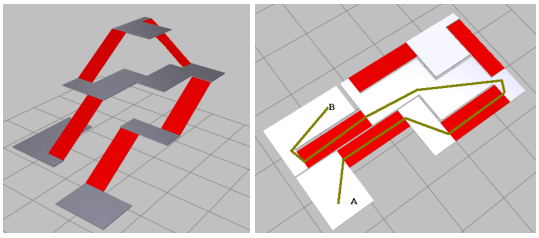


Figure 8. 3D world and its 2D equivalent

The data in table 1, shows the results in seconds for both methods. The waiting period reflects the time that it was required to initially compute the path. The moving period shows the time it actually took to move the avatars from the starting point to the end point. Note that the points in figure 8 are only a reference, i.e. actually a small area around the point was considered. This implies that each avatar has its own path, so n different paths were computed.

It is clear from table 1 that there is a substantial difference when computing the paths. As expected the

hierarchical approach is much faster in this stage. Moving the avatars, after the path has been computed, is roughly equivalent for both situations.

The results in table 2 show the results for the same problem, but this time considering collision detection between avatars. When a collision occurs, i.e. an avatar tries to occupy a position where another avatar is placed, the path is recomputed for the first avatar with the current position as the starting point. The waiting periods reported on table 2 refer to all the path computations, both initially, and when a collision occurs

	1 avatar	
2D classic A*	waiting period	0
	moving	1,71
	total	**1,71**
3D Hierarchical A*	waiting period	0
	moving	1,45
	total	**1,45**
	100 avatar	
2D classic A*	waiting period	18,9
	moving	2,75
	total	**21,65**
3D Hierarchical A*	waiting period	0
	moving	2,57
	total	**2,57**
	1000 avatar	
2D classic A*	waiting period	149
	moving	12
	total	**161**
3D Hierarchical A*	waiting period	1,07
	moving	11,58
	total	**12,65**

Table 1. Results for world in figure 8. The results are in seconds.

	10 avatar	
2D classic A*	waiting	14,5
	total	**16,2**
3D Hierarchical A*	waiting	0,4
	total	**2,05**
	50 avatar	
2D classic A*	waiting	43,2
	total	**45,2**
3D Hierarchical A*	waiting	4,67
	total	**6,6**
	100 avatar	
2D classic A*	waiting	104,8
	total	**112**
3D Hierarchical A*	waiting	10,53
	total	**13,1**

Table 2. Results for world in figure 8 with collision detection between avatars.

Again it is clear that the hierarchical approach represents a very substantial improvement over the classical, non hierarchical approach. This is to be expected since in the hierarchical approach the path is only recomputed inside each node, whereas in the classical approach the path is recomputed until the end area.

The world in figure 9 was tested only with the hierarchical approach as it is a world with multilevel passages. The starting area is on the centre of the ground floor, and the finishing point is on the left zone of the last floor. The search grid is also 512x512 for each slice. As mentioned before, only the graph node where the avatar is placed is searched, hence the search space is greatly reduced, except in the ground floor where almost all the grid is walkable.

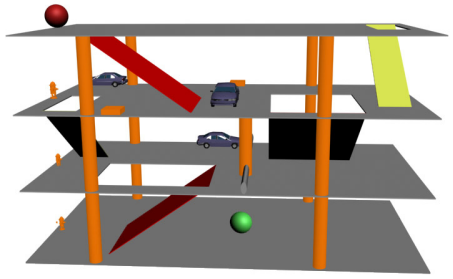


Figure 9. 3D multilevel word with start (green ball) and end (red ball) positions.

The results in table 3 and 4, show the required times without and with collisions respectively.

1 avatar	
waiting period	0
moving	3,11
total	**3,11**
100 avatar	
waiting period	1,14
moving	6,48
total	**7,62**
1000 avatar	
waiting period	9,23
moving	20,19
total	**29,42**

Table 3. Results for world in figure 9.

Note that, as mentioned before, each avatar has its own path, therefore there are as many searches as there are avatars. Although this is not the optimal situation in this particular case, where there is a common goal to all the avatars, it shows how the algorithm scales with the number of avatars. This is not a restriction of our method, as we can deal with groups adding another level to the hierarchy for crowd management, and compute a single path for the leader.

When comparing the results in tables 3 and 4, it is clear that adding collision detection between avatars has a

significant impact on the performance. However note that the collision detection per se is not relevant as shown in [Ramires and Deusdado, 2006]. The increase in seconds is due to the recomputation of the path each time there is a potential collision.

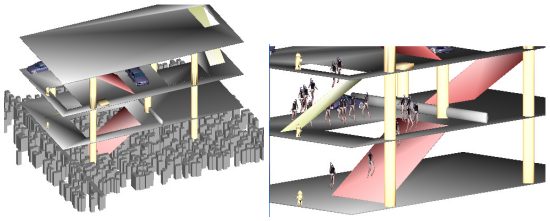


Figure 10. Tests with avatars

10 avatar	
waiting period	0,47
total	**3,6**
50 avatar	
waiting period	9,1
total	**13,7**
100 avatar	
waiting period	24,68
total	**32,3**

Table 4. Results for world in table 9 with collision detection between avatars.

7 Conclusions

The paper addresses the issue of path planning on arbitrarily large and complex virtual environments, with multilevel passages. This is the case in real world buildings, and large man made infrastructures such as stadiums, or industrial facilities, although the method is not conceptually limited to any particular type of world.

The method described is a new approach towards the pre-processing of an unknown polygon soup in order to obtain accessibility information and construct a hierarchical navigation structure that when combined with A* provides an efficient and complete solution.

The only stage that deals directly with the polygon soup is the initial stage where the depth maps are obtained for the world. All other stages are image based.

Being essentially an image based algorithm the performance does not suffer significantly when the polygon count increases. Furthermore, the method does not rely on any particular topology, hence the polygon soup layout does not affect the navigation maps where path planning is performed.

The process is fully automated, without any user intervention, and it is capable of dealing with multilevel passages in a natural fashion. The algorithm is capable of extracting navigation features, accessibilities, and construction of a hierarchical navigation structure.

The method provides a sustainable number of nodes, although more nodes are probably desired in some circumstances, for instance in the ground floor of the simple world. This can easily be achieved through partioning of larger areas, creating more nodes, or even adding another level on the navigation hierarchical structure.

Since the result is by construction a hierarchical navigation structure, when with traditional path finding algorithms provides good results, even when considering collisions between avatars.

Bibliography

SALOMON B., GARBER M., LIN M., MANOCHA D., 2003, Interactive navigation in complex environments using path planning, In *Symposium on Interactive 3D graphics*, ACM Press, New York, NY, USA, 2003.

KREYLOS O., 2005, Path Finding In Complex Maps And The Black Art Of Linear Algebra, In *http://graphics.cs.ucdavis.edu/~okreylos/Private/AlgorithmCorner/.*

BALLEGOOIJ, A. V., ELIÉNS A., 2001, Navigation by query in virtual worlds, In *Virtual Reality Modelling Language Symposium - 3D Web technology*, Paderbon, Germany.

NITSCHE, M., RICHENS P., 2005, *Combining linear content and spatial design for Mindstage*, In Media in Transition 4: The Work of Stories, Boston.

ARIKAN O., FORSYTH A., 2001, *Efficient multi-agent path planning*, In Computer Animation and Simulation O1, Springer-Verlag.

HOFF III, K., CULVER T., KEYSER J., LIN, M., MANOCHA D., 1999, In *Fast Computation of Generalized Voronoi Diagrams Using Graphics Hardware*, SIGGRAPH.

HOFF III, K., CULVER T., KEYSER J., LIN, M., MANOCHA D., 2000, *Interactive motion planning using hardware-accelerated computation of generalized Voronoi diagrams*, In International Conference on Robotics and Automation.

LAMARCHE F., DONIKIAN S., 2004, *Crowd of virtual humans: a new approach for real time navigation in complex and structured environments*, In Eurographics 2004, Volume 23. Nr 3.

ANDÚJAR, C., VÀZQUEZ, P., FAIRÉN M., 2004, *Way-Finder: guided tours through complex walkthrough models*, In Computer Graphics Forum.

LOSCOS, C., MARCHAL, D., MEYER A., 2003, *Intuitive Crowd Behaviour in Dense Urban Environments using Local Laws*, In Tpcg.

DAPPER, F., PRESTES, E., IDIART M., NEDEL, A., LUCIANA, P., 2006, *Simulating Pedestrian Behavior with Potential Fields*, In Computer Graphics International 2006 (CGI), Hangzhou, China.

BANDI, S., THALMANN D., 1995, *An Adaptive Spatial Subdivision of the Object Space for Fast Collision Detection of Animated Rigid Bodies*, In Eurographics '95.

BANDI, S., THALMANN D., 1998, *Space discretization for efficient human navigation*, In Eurographics '98.

PETTRE, J., LAUMOND, J.P., THALMANN, D., 2005, *A navigation graph for real-time crowd animation on multilayered and uneven terrain*, In First International Workshop on Crowd Simulation (V-CROWDS´05), Lausanne, Switzerland.

PETTRE, J., LAUMOND, J.P., THALMANN, D., 2006, *Real-time navigation crowds: scalable simulation and rendering*, In Computer Animation and Virtual Worlds 2006, vol. 17- pp 445-455.

STURTEVANT, N. M. B., 2005, *Partial path finding using map abstraction and refinement*, In Twentieth National Conference on Artificial Intelligence and the Seventeenth Innovative Applications of Artificial Intelligence Conference.

HOLTE R., PEREZ M., ZIMMER, R., MACDONALD, A., 1996 *Hierarchical A*: Searching abstraction hierarchies efficiently*, In Thirteenth National Conference on Artificial Intelligence (AAAI-96), 1996.

MAIO, D., RIZZI, S., 1994, *A hybrid approach to path planning in autonomous agents*, In Second International Conference on Expert Systems for Development.

FRÖHLICH, T., KULLMANN, D., 2002, *Autonomous and Robust Navigation for Simulated Humanoid Characters in Virtual Environments*, In First International Symposium on Cyber Worlds (CW'02).

LESTER, P., 2004, *A* Tutorial*, in http://www.policyalmanac.org/games/aStarTutorial_port.htm, 2004.

RAMIRES A., DEUSDADO L., 2006, *Efficient Conservative Collision Detection for Populated Virtual Worlds*, In Ibero-American Symposium on Computer Graphics - SIACG(06), Santiago de Compostela, Spain.

OPENCV, in http://www.intel.com/technology/computing/