Sketch-Based Annotation and Authoring of Geometrically Sparse 3D Environments

Rowan Hughes, Jan Ondřej, Guarev Chaurasia

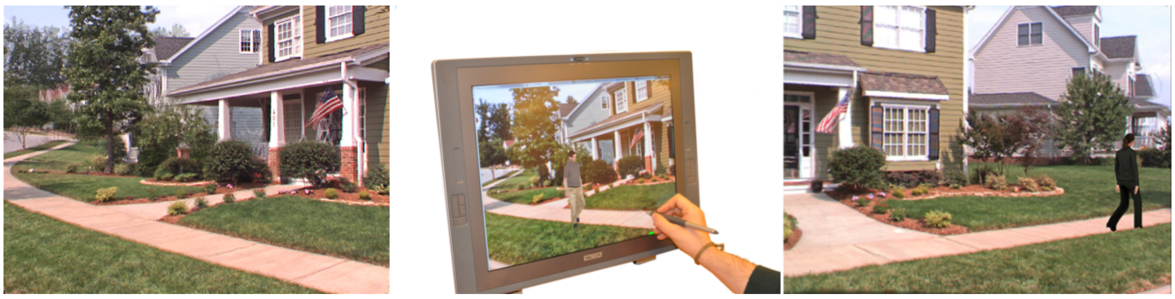

Figure 1: *Spring Training 2009, Peoria, AZ.*

Abstract

The annotation of scenes to facilitate path-finding and dynamic virtual agent behaviors has been the subject of much work, and although it could not be considered a solved problem, many modern real-time games exhibit very robust agent navigation. Most of this work, however, has dealt with geometrically rich 3D environments, and the underlying algorithms for annotating the environment relies totally on this information being readily available. Not all 3D scene representations contain a dense underlying geometric representation. This presents a major problem when we wish to populate such scenes with virtual objects and agents. We present a method to quickly and efficiently annotate a geometrically sparse, or indeed empty, 3D environment such that it can be populated with virtual objects and navigated intelligently by virtual agents. We then expand upon the concept to deliver a set of authoring tools that build upon the same approach.

1. Introduction

There is an ever increasing demand for perceived user "presence" in populated virtual environments. This enables users to truly immerse themselves in the scenario. In the case of virtual reality (VR) for rehabilitation, the need for realism is vital. In order to relieve patients from social phobias such as agoraphobia, physical impairments such as fear of falling and freezing of gait, or performance anxiety through the use of multisensory VR , the patient needs to have a feeling that they are in fact immersed in a familiar and living environment populated with intelligent virtual agents, otherwise the therapy may not have the desired effect.

The annotation of virtual environments, to facilitate their population with virtual agents capable of intelligent navigation, has been the subject of much work and although it could not be considered a solved problem, many modern real-time applications exhibit very robust agent behavior. Most of this work, however, has dealt with geometrically rich 3D environments and the underlying algorithms rely on this data in order to operate. An example of this approach is the navigation mesh [Sno00], which is a widely adopted industry solution to the problem. The vast majority of modern video games make use of this method for the description of the navigable areas in 3D scenes. This method facilitates virtual human path-finding and collision avoidance. Generation of these meshes is largely an automated process: algorithms analyze the underlying scene geometry to output a set of connected polygons which define the navigable area.

Modeling of, quality, virtual environments is a time consuming, labor intensive and highly specialized task. In cases where the creation of detailed geometrical models of the desired environments is not possible or practical, alternative techniques are often employed to deliver interactive scenes at a fraction of the time and effort. Image-based rendering (IBR) techniques can allow for the rapid development of such 3D environments [**?**, **?**]. While quite realistic results can be achieved through the use of these techniques, many open problems remain. When working with image based representations of real environments, there is little or no geometric data available. This is hugely problematic for the accurate placement and animation of 3D objects. In order to develop truly realistic and useful 3D environments it is crucial to facilitate the addition of static scenery and dynamic objects, such as intelligent virtual agents, to our scenes.

Over the past number of years there has been a proliferation of research into sketch-based modeling approaches, across a wide variety of disciplines. Systems adopting the intuitive and familiar nature of conventional drawing techniques have become a popular alternative to those that use more traditional user-input. In the design of our system we speciïňĄcally wanted to provide an interface that could be used by artists and other non-technical users alike, a sketch-based interface provided one such solution.

Our work was driven by the need to develop a set of clinical intervention tools in which an artist or non-technical user could take a locale to which a patient would be familiar and transform it, rapidly, into a fully realized, populated, 3D scene with which the patient could interact. Given these constraints traditional modeling based approaches to scene development were not an option and alternate method to environment representation, an image-based rendering (IBR) solution, was selected. The method described in this paper was borne out of the resulting problem; how to populate and author scenes in which the environment has little, or indeed, no geometric information. Our approach provides a sketch-based interface to facilitate the creation of navigation meshes in geometrically sparse scenes and further annotate the scene to facilitate the insertion and animation of intelligent virtual agents. We further extend this sketch-based approach to demonstrate some possibilities for further semantic annotation and how the approach can be used to augment and author crowd behavior.

The structure of the rest of the paper is as follows: first, we present our approach to annotation of the environment, followed by a description of behavior authoring. Next, we compare our results with a geometry-based method applied on a reconstructed mesh and discuss issues of our approach. Finally, we present possible directions for future work.

2. Related Work

Environment representation for path planning. The field of environment representation for path planning and path planning itself was extensively studied in the past and an overview of these methods is provided in [Lat91, LaV06]. Three main approaches are used to represent an environment; potential field, road maps and cell decomposition. Potential field methods represent obstacles as areas with a high potential and use gradient methods to find a path to the goal [War89]. Road maps capture the connectivity of the free space by using a graph-based representation. They can be constructed by connecting visible vertices of the environment geometry [ACF01], by computing generalized Voronoi diagram inside the free space [SGA*07], randomly sampling free space [BLA02]. The Cell decomposition methods decompose the environment's free space into cells either by approximating it using predefined cell shapes, such as uniform grids [ST05] or circles [PLT05], or by representing it exactly using convex polygons [Sno00, KBT03, LD04, Lam09].

The acquisition of these simplified representations of the scene for path planning is usually done automatically from a polygonal geometry (e.g. [MHP13]) and thus is unsuitable in geometrically sparse environments such as those in image based rendering (IBR) adopted in our project [CSHD11].

Tools for annotating/authoring populated scenes Annotation and authoring of populated scenes is still in many cases a lengthy and a highly specialized task. Several industry packages exist such as, Massive [mas03] or Golaem [gol10]. However, the use of these tools requires a skilled user. A simpler and more intuitive approach was presented in [UCT04] where a crowd-brush tool, similar to image manipulation programs, is used to populate and author scenes with virtual agents.

The recent work described in [**?**] shows how three dimensional objects can be rendered into 2D images through the annotation of geometry in the scene. An initial estimate as to the geometry is made and then is manually fine tuned by the user. Occluders can also be annotated to create a more rich 3D environment. Once annotated, animated objects can be rendered into the scene in a highly believable manner. Our system uses a similar approach to describing the scene's geometry in cases where a proxy geometry cannot be constructed.

In [**?**] a gesture-based approach to behavioral authoring is described. Through the input of mouse described gestures a sequence of behaviors for virtual characters can be authored. Our work differs from this approach in that although we can directly author virtual characters behavior through sketch based input, the thrust of our authoring approach is to author the scene and and thus allow the environment to affect character behavior.

The work described in [**?**] details an approach to realistically populating a game world with virtual agents. Specifically, they control the distribution of the crowd in the scene through *Navigation Flows*, a subset of user specified agents that endlessly navigate between desired waypoints. We attempt to control crowd distribution also, although in our case through direct sketch input. Larger sketched waypoints represent more populous areas of the map and vice-versa. In this way we can achieve the desired crowd flow for our scenes.

Sketching provides an approach that allows users without necessary technical skills to work intuitively on a highly specialized tasks. Indeed, sketch-based interfaces are used in many areas of computer graphics [OIL09] including authoring of populated scenes, animations and behaviors of virtual humans. In character animation, sketching can be used to sketch sequences of animated motions (e.g. walk, jump or flip) [TBvdP04] or to manipulate and constrain motions from a physically-based animation system [MCC09]. In crowd simulation, sketches are used to control formation shapes [SPN08,GD11], Oshita and Ogiwara [OO09] use sketched paths for navigation and to compute basic parameters for the crowd following the path (speed of agents and distance between agents), or to sketch directions for navigation fields [PvdBC*11].

In this paper we present novel sketch-based tools to annotate semantic information in geometrically sparse environments and to populate them with virtual humans. First, we provide a sketch-based scene annotation tool that caters for the cases where a proxy geometry both can and cannot be automatically generated. It must be noted that when sufficient geometry is present to perform a proxy mesh construction, as is the case with the IBR system, the resulting mesh may not have the detail required for a successful automatic annotation of the navigable areas, as shown in Section 4. Second, we provide clinicians with a basic set of crowd authoring tools, which can be easily extended in the future, to allow them intuitively augment and author virtual humans' behavior. Lastly, thanks to the distributed nature of our crowd system we developed a prototype of a network-connected tool that will let clinicians adapt the behavior of virtual humans on-line during patient's interaction with the scene.

3. Method

Although our approach can be utilized with fully modeled 3D scenes we were interested in designing a practical solution to authoring and annotation in cases where little geometric detail is available. With this in mind, and in order to provide a test bed for our method we made use of the IBR technique as described by Chaurasia and Drettakis [CSHD11]. Using this approach, 3D scenes can be reconstructed from photographs which fit well with the needs of our proposed solution.

Figure 2 gives a broad overview of the structure of our system. We split the description of our method into two distinct subsections, annotation and authoring. We define annotation as the process of scene description to enable intelligent scene navigation by virtual agents. We define authoring as the process of modifying virtual agent behavior according to user input. Our sketching system does not make use of any geometric data in the scene, sketches are made upon user defined proxy geometry or planes, which we will refer to as navigation surfaces from this point. Our approach can make use of any geometric data that does exist for the semi-automatic placement of these surfaces can make use of, the details of which we will outline later.

Sampling of the userâĂŹs sketch. User sketched contours are the input to describe the navigation mesh. To generate contours in our system the user sketches the desired outline locations of the mesh onto the navigation surfaces within the scene. Simple screen-space unprojection in conjunction with ray-triangle intersection is used to determine the sketch intersection points in the navigation surface, which lie in world space. These sketches are visualized in real time in order to ensure that the user has maximum control over the desired contour.

It is important to note that the sampled points represent the vertices in our final navigation mesh. It is, therefore, highly desirable to limit these points in order to limit the number of polygons present in the final mesh as this directly influences the complexity of agent navigation. With this in mind we capture user sketches at the highest frequency available, to capture as much data as possible. When the sketch is complete, we iterate through the list of connected points and remove any whose distance from the previous point is less than a pre-determined sample threshold α. This technique greatly reduces the number of samples, thus allowing for explicit control of the complexity of the final mesh. We include the ability to smooth the sketch and resultant sample points through spline interpolation but we have found that this, while visually appealing, introduces too large a deviation from user input.

3.1. Annotation

Our approach to scene annotation follows a pattern similar to that of traditional approaches. The general order of annotation is as follows:

1. Depending on nature of input data place or sketch navigation surface(s)
2. Sketch navigable area(s)
3. Sketch obstacle(s)
4. Sketch waypoint(s)

Depending on the type of input data to the system, proxy geometry may need to be manually laid upon which to sketch. Most of our testing was performed using the system described in [CSHD11]. As such we may, depending on the level of noise in the data, reconstruct a suitable proxy geometry upon which to sketch directly from the point cloud generated by this system. If one is working with data that does not allow for automatic construction of proxy geometry such as 2D images [?] or if a suitable geometric repre-

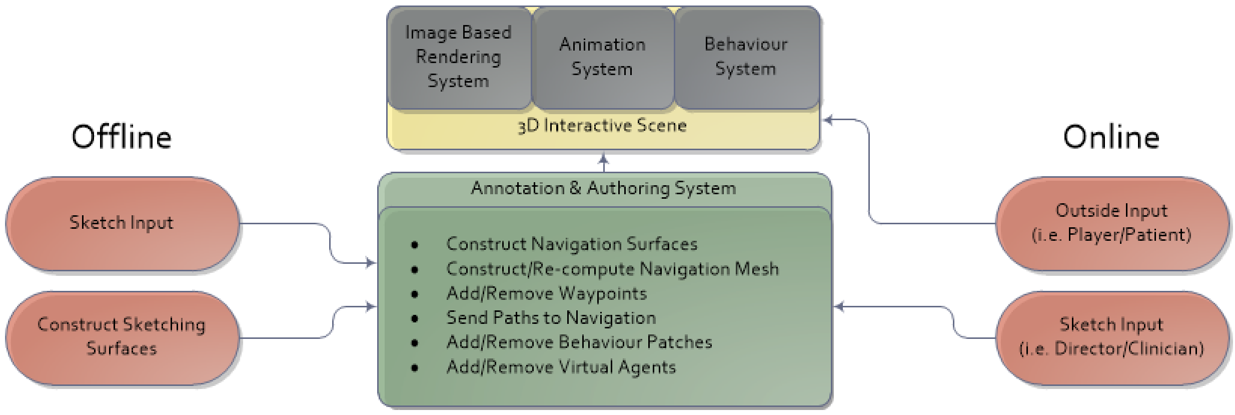

Image Based Rendering System
Animation System
Behaviour System
3D Interactive Scene
Offline
Online
Sketch Input
Construct Sketching Surfaces
Annotation & Authoring System
Construct Navigation Surfaces
Construct/Re-compute Navigation Mesh
Add/Remove Waypoints
Send Paths to Navigation
Add/Remove Behaviour Patches
Add/Remove Virtual Agents
Outside Input (i.e. Player/Patient)
Sketch Input (i.e. Director/Clinician)

Figure 2: *Overview of our system.*

sentation can not be automatically constructed then we must manually construct a proxy geometry. To this end the first step in our approach is to fit planes or navigation surfaces to the desired areas in the scene. Figure 3 shows an example of a placed navigation surface.

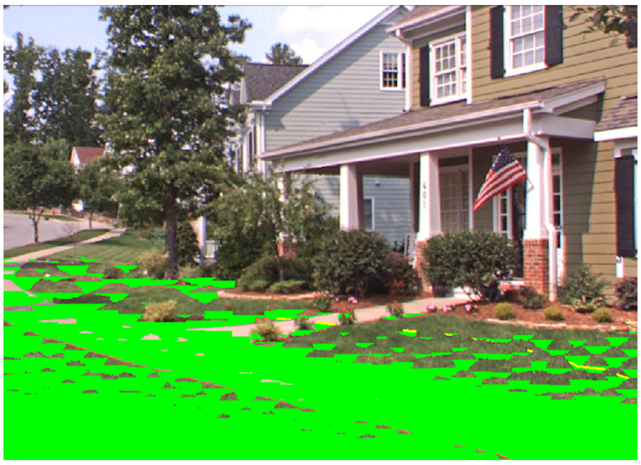

Figure 3: *Large user placed navigation surface, in green, upon which the navigable area can be sketched.*

Considering the case where automatic construction of the proxy geometry is not possible our system provides a method of constructing such. Navigation surfaces can be placed in the scene in a number of ways. If there is sufficient geometry in the scene then the user can sketch on the proxy surface where there is deemed to be a navigable area, an approximate navigation surface is automatically generated from the sketched points. If there is inadequate geometry for this approach then a plane can simply be added to the scene which then can be adjusted by the user until the desired fit has been acquired. Quite often, given the constraints of image based scene reconstruction, the scenes are relatively simple and only a small set of navigation surfaces need to be placed. Scenes with a more complex geometry are catered for by our system as many surfaces can be placed and adjusted by the user. Once a coarse proxy geometry has been described upon which a user can sketch, construction of the final navigation mesh can begin.

The user defines the navigable areas through sketching outlines as desired directly on the scene. Once the sketch is complete the contour is automatically closed and the resultant affine hull is then triangulated [KBT03] and outputted in a format that can then be processed by a behavior planning system. We use the method from [LD04] for global path planning and [POO*09] for local dynamic collision avoidance. In the simplest case, one single closed contour can describe the navigable area. Often, however, where more complex geometry is involved it is not feasible to describe the entire navigable area with a single sketch. There are many cases in which this can occur and in an attempt to cater for this, our system allows the user to move about the scene and describe, by sketch input, multiple navigation areas. These areas may or may not be connected. Our system merges connected sketched contours into a single navigable area and also allows for multiple disconnected areas. The next stage in completing our navigation mesh is the description of obstacles within the mesh (i.e. cars, trees, lampposts, etc.). We use exactly the same sketch-based approach to outlining these obstacles. Once the borders of the obstacles have been sketched, via outlining their bases on the navigation surface, the resulting borders are added to the final navigation mesh. Figure 8 shows two examples of an outputted final navigation mesh for a simple scene (bottom row).

Now, that a final navigation mesh has been constructed, we can begin to populate the scene with virtual characters. In order to allow agents that are capable of intelligent navigation, rather than random traversal, we require additional information. We cater for this situation again via sketch-based input. Hotspots, or way-points, can be sketched directly on the navigation surfaces. These hotspots are the areas in the scene between which the agents will navigate. Although the hotspots may be used in the construction of a traditional waypoint map, they can also represent additional information via their size. Larger hotspots represent more active and more populous areas in our scene. As an example, for a large square in a scene we can use a larger hotspot as we wish more agents to navigate to this area, wish them to linger in the area longer before moving on, etc. Using a hotspots size as input, therefore, allows us to implicitly author a set of agent and system behaviors in that the size of the sketched hotspot directly affects the agent navigation and behavior systems. Our sketch-based approach leads to an intuitive authoring of these features as larger sketched contours represent larger hotspots. Any number of these hotspots can be placed in the scene according to user preference. In conclusion, hotspots as well as providing the information required for scene traversal also provide a useful semantic for crowd behavior as well as initial scene population.

3.2. Authoring

The annotation phase facilitates the placement of crowds within our scene. To provide a richer set of agent behaviors and a more meaningful user experience, we require a method of scene authoring. No completely autonomous agent model can sufficiently capture the entire space of possible agent behaviors and interactions. In order to produce customizable scenes we propose a method of providing scriptable agent behavior through a set of tools designed for artists or other non-technical users. Through these tools we allow for the offline and online modification of agent behavior through sketch-based input, including:

- Sketch-based population of environment
- Sketching paths
- Sketching behavior patches
- Sketching interaction patches

Scenes can be populated automatically according to the number and sizes of the hotspots or by manually placing agents into the scene one at a time. We also allow for the selective deletion of agents via sketching.

Users may select an agent, or agents, by sketching around them. Once the agent(s) have been selected the user may sketch trajectories for them to follow. Figure 4 shows one such user defined path. These paths override the scene waypoints. The agents may stop once they reach their destination or, if desired, may resume navigation of the scene. This allows a user to set up the scene with a combination of static and dynamic characters in order to realize their desired scenario and/or to manipulate the scenario in real-time according to need.

Although our system caters for a traditional pick and control interaction seen in many modern video games (i.e The Sims), the main objective of our approach was to provide a quick and intuitive solution to author the scene itself. The authored scene then affects the virtual agents within it. In order to imbue our virtual agents with a richer behavior set we include the ability for the user to describe *behavior patches*. Simply having crowds navigating the scene correctly provides limited usability for our potential scenarios, we need crowds to demonstrate a wider behavioral variety to allow for increased user "presence" in the virtual environment. Therefore, we need some method of scripting agent behaviors, the sketching of these patches provides this ability. The user can outline an area, much like the description of waypoints, inside the navigation mesh that he wishes virtual

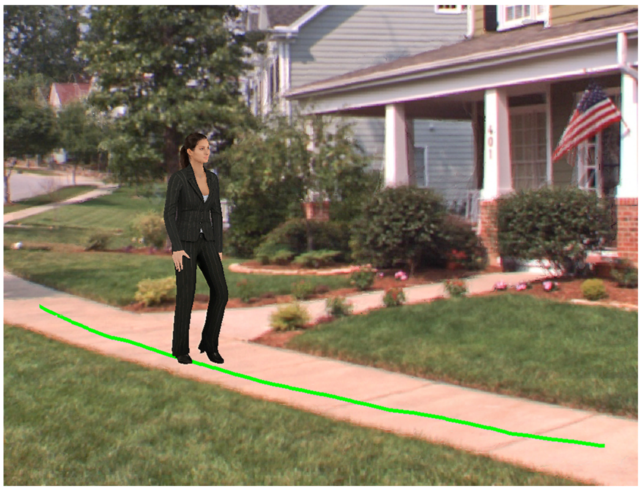
Figure 4: *Sketch described path that the selected virtual agent will follow.*

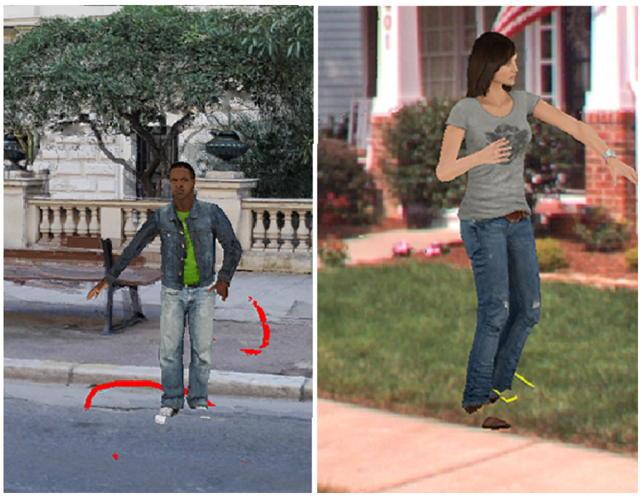
Figure 5: *Examples of sketched behavior patches, upon arrival the animated agents will perform an action. (In these cases the agents dance.)*

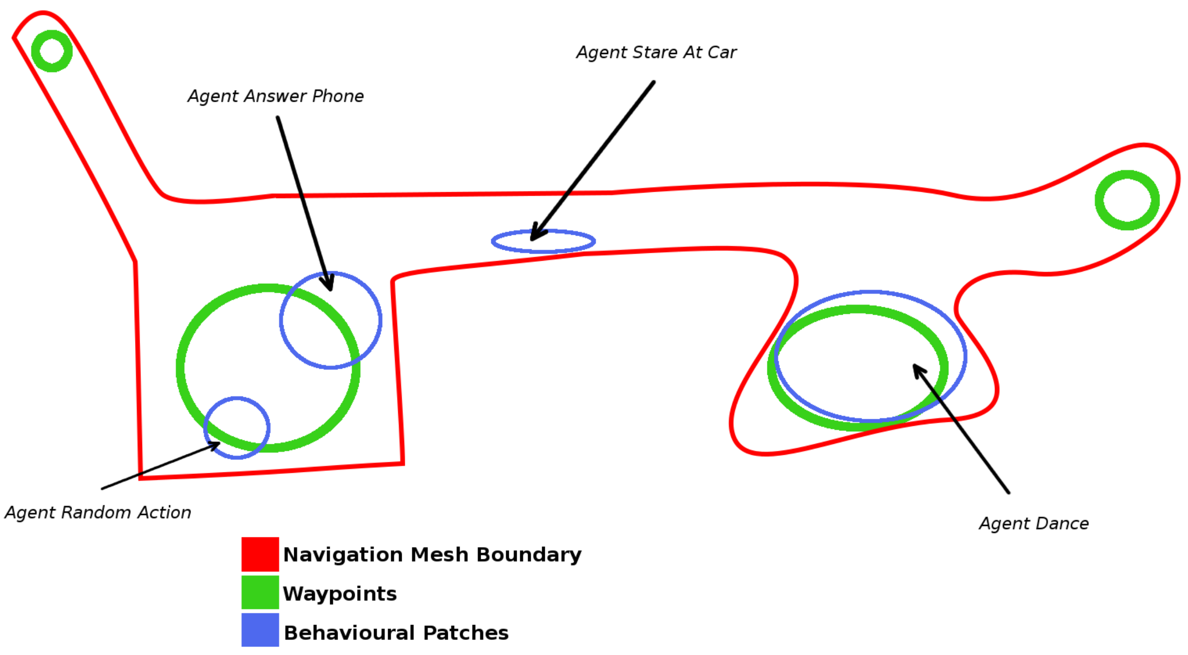

Figure 6: *Visualization of a simple scene that has been hand authored by a user using sketch based input. (Note that annotated elements have been abstracted for the purposes of clarity.)*

agents to interact with in some way (i.e. Agents should stop and look at a car, take a phone call, etc.). Figure 5 shows an agent whose behavior has been modified through a behavior patch. Our system allows for agents to hit these patches randomly or they can be authored, via a sketched path, to do so. Another important feature of behavior patches is that they can be used as in interface to more scripted portions of the user experience. As a user guides an avatar over these patches we can use this as an event that can be used to drive agent behavior. This allows for far richer crowd behavior and demonstrates the extensibility of our approach in rapidly annotating, in a highly customizable fashion, the scene through sketch input.

Behavior patches allow for the scripting of single agent behaviors that interact with the environment or change their state somehow. In order to facilitate the scripting of inter-agent behaviors we adapt the idea of *interaction patches* [SKSY08] [KHHL12] to our sketch-based system. The user may select any number of agents and sketch an interaction patch. The selected agents will navigate to the patch and perform the desired interaction, the behavior and animation systems ensures spatio-temporal correctness. This provides another powerful tool in the creation of custom scenarios.

Figure 6 shows a fully authored simple scene. Note the varying sizes of the waypoints, this gives a rough indication of the population distribution within the map.

4. Discussion & Future Work

The presented sketch-based solution to annotation and authoring of 3D environments is robust and intuitive. The tools we provide allow users to annotate geometrically impaired scenes with navigable areas, obstacles and waypoints, populate them with virtual humans and author their behaviors.

As a preliminary evaluation of the navigation meshes created by sketching, we compare the navigation meshes created by our sketching tool to the navigation meshes automatically constructed by the state of the art tool-set, Recast [MHP13]. In order to perform the automatic construction, we leveraged the availability of the point cloud in our IBR data sets to reconstruct a mesh suitable for use with Recast using a 3D modelling tool. Figure 8 clearly shows that this approach does not generate suitable results, displaying errors in mesh size, orientation and continuity. The meshes simply do not contain the geometric granularity to produce satisfactory results.

It is often the case that, even when created from fully modeled geometry, the output of automatic navigation mesh construction tools contain areas where the author would not wish virtual agents to navigate. While methods are available to deal with these cases, many are cumbersome, involving vertex manipulation. A sketch-based approach could be used in conjunction with these tools to provide a more intuitive user experience.

Understanding the limits of this approach to authoring is key. While we can direct general crowd behavior and their navigation strategy reasonably well, to achieve truly complex, event driven experiences for users some offline authoring is essential. One major advantage of our approach is that, through behavior patches, it does provide a customizable interface to these more heavily scripted portions of the narrative.

A current drawback of our approach is the requirement that, in the cases where suitable geometry can not be automatically constructed, proxy geometry has to be constructed by the user. This proves to be the most challenging aspect of scene annotation stage. In cases where there is no geometry available it would be beneficial to leverage work done in the automatic construction of the proxy geometry through image analysis [**?**]. The test data sets we used mostly consisted of simple scenes with a relatively simple geometry, an obvious issue would arise given a scenario with complex as the number of surfaces that would need to be defined in this case could quickly become unfeasible. Our system was designed with specific deployment scenarios in mind, specifically small outdoor scenes that potential patients would be familiar with. Figure 3 shows a typical environment that we would wish to model and author though our system. So while complex geometry may be a problem in general application of our method, it presents little problem for our planned clinical applications.

The other issues we face in our application are related to the state of the art in image-based rendering and crowd simulation and animation, all of these systems are of critical importance in determining how well the final scenes perform. Depending on the reconstruction that is used there can be a myriad of graphical artifacts both major and minor, for example the ground surface in the 3D representation that we used as a test-bed is not perfectly flat having a slight undulating texture. This leads to small visual anomalies at the agents feet. Occlusion artifacts can occur in those portions of the reconstruction where there exist inaccuracies in depth. All of these issues are becoming less and less of a problem as these techniques have, even since the authoring of this paper, taken large strides in graphical fidelity [CDSHD13].

4.1. Future Work

As we developed our tool as a real-world authoring solution we investigated a number of deployment avenues. One of the prototypes featured a distributed framework where a "director" (clinician) can change agent behavior in a scene via sketch-input from a tablet or mobile device. The scene that the "director" interacts with is a lower dimensional representation of the 3D scene, in that it does not depict the geometry of the scene, only that subset that allows scene direction. Figure 7 shows an example of this low dimensional representation. Much of the same functionality was utilised with the user being able to define navigable areas, obstacles, waypoints. This approach successfully allows for dynamic agents selection and path sketching in the scene that the user is interacting with. In this case the user was interacting with a fully realised scene and we believe this demonstrates the flexibility of the sketch based approach to scene authoring. We are planning to extend this basic functionality with more gestures to allow the "director" intervene in a variety of ways that allows even more personalized treatment.

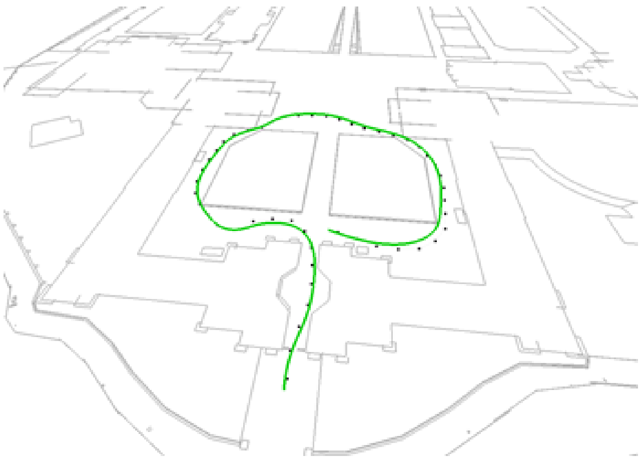

Figure 7: *View of the "directors" interface in the distributed sketch prototype. Only those aspects of the 3D scene that allow the user to direct the scene are represented. Obstacle edges are represented by grey lines, sketch input is depicted in green*

Sketch-based approaches provide an exciting, almost infinitely extensible, solution to scene authoring. Our method only provides a small subset of possible tools but the system was designed with ease of extensibility in mind. The combination of our technique with current annotation tools, such as Recast, would allow for an intuitive interface for navigation mesh construction and refinement. This combined with a simlilar sketch-based approach to scene authoring would provide artists with a powerful set of content creation tools. A major further extension to our approach to authoring would be the creation of a sketch-driven "interactive storyboarding" solution which would allow for the sequencing of multiple linked events, creating interactive stories in realtime. While there are a number of problems that need to be overcome we are confident that the sketch-based approach as described in this paper could be extended to deliver such a system.

As our tools are meant for artists and other non-technical users, intuitive interaction with the system is expected. A novice user should be able to quickly an accurately describe a navigation mesh and associated waypoints. An important part of our future work, therefore, is to run a user study to evaluate usability and effectiveness of our interface.

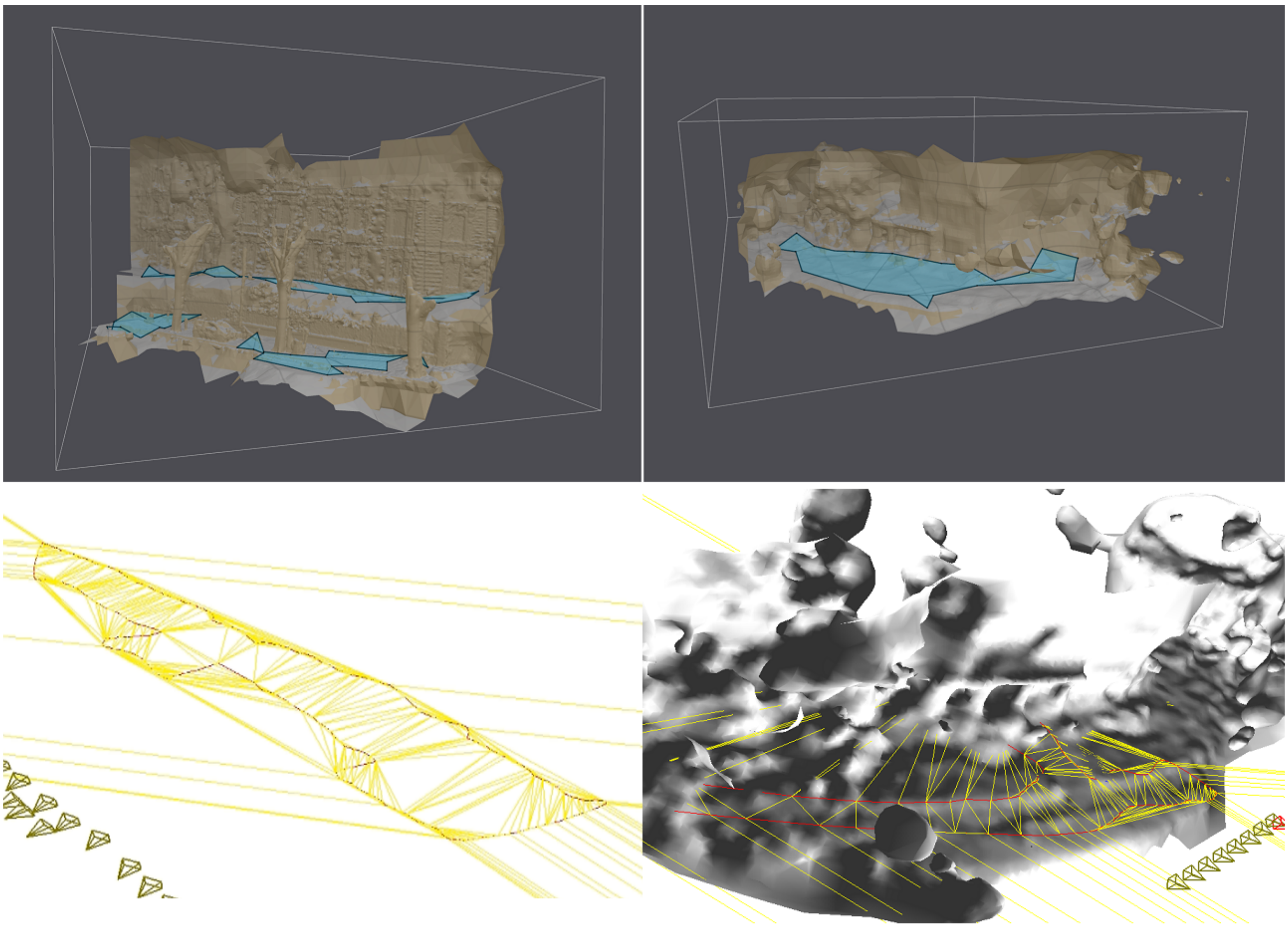

Figure 8: *Top Row: Visualisation of the navigation meshes automatically constructed through the widely adopted Recast tool. In both cases the mesh is a triangulation of the sparse depth map which is used for 3D reconstruction. We can see that the automatic reconstruction cannot describe accurately the desired navigable areas of the scene. Bottom Row: Visualisation of navigation mesh as described by our tool, navigable area is described within the red contour.*

5. Conclusion

Image based reconstruction of 3D scenes has been for some time an active area of research. As these reconstruction methods become more and more believable and accurate [CDSHD13] their true value as an alternative to fully modeled scenes, especially in scenarios where cost and rapid deployment are critical factors, should become apparent. In order to make full use of these representations, we must have some way to annotate, populate and author them. In this paper we present a sketch-based approach to the construction of navigation meshes in geometrically sparse environments and further demonstrate how a similar approach can be used to allow for both off and on-line annotation and authoring. It can facilitate the rapid delivery of immersive, customizable, interactive 3D environments populated with realistic and intelligent virtual agents. In our case this technology is designed for the rapid development of personalized intervention scenarios by physicians, but the approach outlined here can be applied to many domains.

Acknowledgments

This work is funded by the FP7 VERVE project.

References

[ACF01] Arikan O., Chenney S., Forsyth D.: Efficient multi-agent path planning. In *Computer Animation and Simulation 2001*. Springer, 2001, pp. 151–162. 2

[BLA02] Bayazit O. B., Lien J.-M., Amato N. M.: Roadmap-based flocking for complex environments. In *Proceedings of the 10th Pacific Conference on Computer Graphics and Applications* (Washington, DC, USA, 2002), PG '02, IEEE Computer Society, pp. 104–. 2

[CDSHD13] Chaurasia G., Duchêne S., Sorkine-Hornung O., Drettakis G.: Depth synthesis and local warps for plausible image-based navigation. *ACM Transactions on Graphics to appear* (2013). 7, 8

[CSHD11] Chaurasia G., Sorkine-Hornung O., Drettakis G.: Silhouette-aware warping for image-based rendering. *Computer Graphics Forum (Proceedings of the Eurographics Symposium on Rendering) 30*, 4 (2011), 1223–1232. 2, 3

[GD11] Gu Q., Deng Z.: Formation sketching: an approach to stylize groups in crowd simulation. In *Proceedings of Graphics Interface 2011* (School of Computer Science, University of Waterloo, Waterloo, Ontario, Canada, 2011), GI '11, Canadian Human-Computer Communications Society, pp. 1–8. 3

[gol10] Golaem: Crowd simulation software. http://www.golaem.com/, 2010. 2

[KBT03] Kallmann M., Bieri H., Thalmann D.: Fully dynamic constrained delaunay triangulations. In *Geometric Modelling for Scientific Visualization*, G. Brunnett B. Hamann H. M., Linsen L., (Eds.), first ed. Springer-Verlag, Heidelberg, Germany, 2003, pp. 241–257. ISBN 3-540-40116-4. 2, 4

[KHHL12] Kim M., Hwang Y., Hyun K., Lee J.: Tiling motion patches. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Aire-la-Ville, Switzerland, Switzerland, 2012), SCA '12, Eurographics Association, pp. 117–126. 6

[Lam09] Lamarche F.: TopoPlan: a topological path planner for real time human navigation under floor and ceiling constraints. *Computer Graphics Forum 28*, 2 (2009), 649–658. 2

[Lat91] Latombe J.-C.: *Robot Motion Planning*. Kluwer academic publishers, Norwell, MA, USA, 1991. 2

[LaV06] LaValle S. M.: *Planning algorithms*. Cambridge University Press, 2006. 2

[LD04] Lamarche F., Donikian S.: Crowd of virtual humans: a new approach for real time navigation in complex and structured environments. *Computer Graphics Forum 23*, 3 (2004), 509–518. 2, 4

[mas03] Massive, crowd animation software for visual effects. http://www.massivesoftware.com/, 2003. 2

[MCC09] Min J., Chen Y.-L., Chai J.: Interactive generation of human animation with deformable motion models. *ACM Transactions on Graphics 29*, 1 (Dec. 2009), 9:1–9:12. 3

[MHP13] Mononen M., Hart C., Pratt S.: Recast: Navigation-mesh construction toolset for games. https://code.google.com/p/recastnavigation/, 2013. 2, 6

[OIL09] Otaduy M. A., Igarashi T., LaViola Jr. J. J.: Interaction: interfaces, algorithms, and applications. In *ACM SIGGRAPH 2009 Courses* (New York, NY, USA, 2009), SIGGRAPH '09, ACM, pp. 14:1–14:66. 3

[OO09] Oshita M., Ogiwara Y.: Sketch-based interface for crowd animation. In *Proceedings of the 10th International Symposium on Smart Graphics* (Berlin, Heidelberg, 2009), SG '09, Springer-Verlag, pp. 253–262. 3

[PLT05] Pettre J., Laumond J.-P., Thalmann D.: A navigation graph for real-time crowd animation on multilayered and uneven terrain. *First International Workshop on Crowd Simulation (V-CROWDS'05) 43*, 44 (2005), 194. 2

[POO*09] Pettré J., Ondřej J., Olivier A.-H., Cretual A., Donikian S.: Experiment-based modeling, simulation and validation of interactions between virtual walkers. In *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2009), SCA '09, ACM, ACM, pp. 189–198. 4

[PvdBC*11] Patil S., van den Berg J., Curtis S., Lin M. C., Manocha D.: Directing crowd simulations using navigation fields. *IEEE Transactions on Visualization and Computer Graphics 17*, 2 (Feb. 2011), 244–254. 3

[SGA*07] Sud A., Gayle R., Andersen E., Guy S., Lin M., Manocha D.: Real-time navigation of independent agents using adaptive roadmaps. In *Proceedings of the 2007 ACM symposium on Virtual reality software and technology* (New York, NY, USA, 2007), VRST '07, ACM, pp. 99–106. 2

[SKSY08] Shum H. P. H., Komura T., Shiraishi M., Yamazaki S.: Interaction patches for multi-character animation. In *ACM SIGGRAPH Asia 2008 papers* (New York, NY, USA, 2008), SIGGRAPH Asia '08, ACM, pp. 114:1–114:8. 6

[Sno00] Snook G.: Simplified 3d movement and pathfinding using navigation meshes. In *Game Programming Gems*, DeLoura M., (Ed.). Charles River Media, 2000, pp. 288–304. 1, 2

[SPN08] Silveira R., Prestes E., Nedel L. P.: Managing coherent groups. *Computer Animation and Virtual Worlds 19*, 3-4 (Sept. 2008), 295–305. 3

[ST05] Shao W., Terzopoulos D.: Environmental modeling for autonomous virtual pedestrians. In *SAE Symposium on Digital Human Modeling for Design and Engineering* (2005), pp. 1–8. 2

[TBvdP04] Thorne M., Burke D., van de Panne M.: Motion doodles: an interface for sketching character motion. *ACM Transactions on Graphics 23*, 3 (Aug. 2004), 424–431. 3

[UCT04] Ulicny B., Ciechomski P. d. H., Thalmann D.: Crowdbrush: interactive authoring of real-time crowd scenes. In *Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation* (Aire-la-Ville, Switzerland, Switzerland, 2004), SCA '04, Eurographics Association, pp. 243–252. 2

[War89] Warren C.: Global path planning using artificial potential fields. In *International Conference on Robotics and Automation, 1989.* (1989), pp. 316–321 vol.1. 2