5-2012

# Data Driven Approach to Multi-Agent Low Level Behavior Generation in Medical Simulations

Manan Gupta
*Clemson University*, mgupta@g.clemson.edu

# DATA DRIVEN APPROACH TO MULTI-AGENT LOW LEVEL BEHAVIOR GENERATION IN MEDICAL SIMULATIONS

---

A Thesis
Presented to
the Graduate School of
Clemson University

---

In Partial Fulfillment
of the Requirements for the Degree
Master of Science
Computer Science

---

by
Manan Gupta
May 2012

---

Accepted by:
Dr. Sabarish Babu, Committee Chair
Dr. Larry Hodges
Dr. Andrew Duchowski

## ABSTRACT

A multi-agent scenario generation framework is designed, implemented and evaluated in the context of a preventive medicine education virtual reality system with data collected from a sensor network at the University of Iowa Hospital. An agent in the framework is a virtual human that represents a healthcare worker. The agent is able to make certain decisions based on the information it gathers from its surroundings in the virtual environment. Distributed sensor networks are becoming very commonplace in public areas for public safety and surveillance purposes. The data collected from these sensors can be visualized in a multi-agent simulation. The various components of the framework include generation of unique agents from the sensor data and low level behaviors such as path determination, directional traffic flows, collision avoidance and overtaking. The framework also includes a facility to prevent foot slippage with detailed animations during the travel period of the agents. Preventive medicine education is the process of educating health care workers about procedures that could mitigate the spread of infections in a hospital. We built an application called the 5 Moments of Hand Hygiene that educates health care workers on the times they are supposed to wash their hands when dealing with a patient. The purpose of the application was to increase the compliance rates of this CDC mandated preventive measure in hospitals across the nation. A user study was performed with 18 nursing students and 5 full-time nurses at the Clemson University School of Nursing to test the usability of the application developed and the realism of the scenario generation framework. The results of the study suggest that the behaviors generated by the framework are realistic and believable enough for use in preventive medicine education applications.

## ACKNOWLEDGMENTS

I would like to acknowledge my advisor, Dr. Sabarish Babu, who was a tremendous source of information throughout the lifetime of this project. Without his guidance, I would not have been able to get as much accomplished on this subject. I would like to thank the members of the Virtual Environments group for being there to provide feedback on ideas.

I would also like to thank my parents for all their love and support.

## TABLE OF CONTENTS

Table of Contents (Continued)

## LIST OF FIGURES

List of Figures (Continued)

# CHAPTER ONE

# INTRODUCTION

Multi-agent simulations are employed in the field of computing because of their applications in creating rich visualizations. They can be used to make architectural walkthroughs [4], historical reconstructions [5], and teaching environments [6]. Here an agent is defined as an entity in the virtual environment that takes in the surrounding environment geometry and can make decisions in the environment that help it achieve its goal. Multi-agent systems have multiples of these agents that all act independently in order to accomplish their own goals. In these simulations, allowing the various virtual agents to reach their destination is not good enough. The virtual agents need to be able to travel within the virtual environment in a seamless manner without any foot slippage or any collisions with other agents, but they also need to be able to interact with their environments and also interact with each other.

Low level behaviors include the actions that are involved in each virtual agent traveling from their starting point to their destination. These low level behaviors could include path determination, maintenance of traffic rules, overtaking other agents, etc. This behavior can be contrasted with high level behaviors that include communication between agents, agents stopping to chat with each other, agents interacting with a painting on a wall, or agents walking in a social group. A compelling multi-agent low level behavior generator should be able to generate realistic low level behaviors that can have high level behaviors layered on top in order to make a compelling simulation.

Creating these multi-level behaviors can be a challenging computation problem because these behaviors need to be realistic and compelling on their own but also when they interact with each other. Thus, a virtual agent walking along a path talking to another agent walking alongside it should not end up with an emergent behavior that is not consistent with reality. Crowd simulations, so far, can accurately replicate the low level behaviors of the virtual agents. However, they don't easily allow for high level behaviors to be layered on top of the low-level behaviors. The framework presented here is one that models the low-level behaviors in a manner that allows for the layering of higher level behaviors on top.

Distributed sensor networks are becoming commonplace in public areas for public safety and surveillance purposes [7]. Using sensor network data to drive the movement of virtual agents in a given space has the benefit of generating real traffic patterns. The traffic patterns in these environments have the potential for being very complex which will lead to more compelling behaviors when high-level interactive behaviors are layered on top of these sensor-driven low level behaviors.

The key contributions here are the description of a framework that generates low level behaviors in a multi-agent scenario generator, incorporation of distributed sensor network data to drive the movement of the agents, evaluation of the framework, an application that uses the implementation of that framework and the study that evaluates the engagement, realism, believability, etc. of the behaviors generated by the scenario generation framework.

# CHAPTER TWO

# RELATED WORK

In this section, related work in the field of automatic movement generation for agents in simulations is described.

There are multiple ways to solve the problem of allow multiple-agents to traverse a single shared environment.  A central approach allows for a single entity to plan out the paths for all the agents in the environment.  This approach guarantees collision-free paths, however, the complexity of the solution increases as agents are added to the simulation.  Another approach which is discussed in this section treats every agent as a separate entity where the information that the agent can act on is limited.  This method is more efficient computationally for larger number of agents.

Sud et al. use a multi-agent navigation graph calculated from Voronoi diagrams in order find a collision-free route to an agent's destination.  This method is a very fast and can simulate hundreds of agents in crowd simulations [8].  Ming et al. use Adaptive Elastic Roadmaps (AERO), which is a lazy computation of a crowd dynamics model, in order to simulate heterogeneous crowds.  This algorithm features groups of agents that have similar behaviors and goals.  This algorithm does well to simulate the emergent behavior in low level movement [9].

Karamouzas and Overmars used experiments and observations of humans avoiding collision in order to formulate a model that could realistically avoid collision. Their approach relies on looking ahead in the path for every agent and making slight changes in the orientation and speed of the agent in order to get a collision-free path for the agent.  The main metric for these decisions is the predicted time to collision for an agent [10].  However, these approach mentions no means of wayfinding in a complex environment.  They simply suggest methods to allow agents to walk to their individual destinations in a collision-free manner.

Jur van den Berg solves the problem of multi-agent navigation by using a randomly sampled roadmap in order to provide path planning for the agents.  For collision avoidance, Berg uses a continuously updating method that excludes all velocities for an agent from consideration that would result in a collision with its nearby environment [11].

These methods are sufficient for generating low-level behaviors.  However, these methods are not data driven in nature and they don’t easily afford addition of high-level interactive space behaviors as a layer operating above them.  The contributions here are ones that make a data driven simulation possible with the ability to add interactive behaviors on top of the low-level behaviors generated by the framework.

# CHAPTER THREE

# SYSTEM FRAMEWORK

The framework is developed to be extensible so that other behaviors can be composed on top of the low level behaviors that are generated by the framework. The low level behaviors generated by the framework need to accurately represent the movement of the people recorded by the distributed sensor networks. The various components of the framework are shown in Figure 3.1.

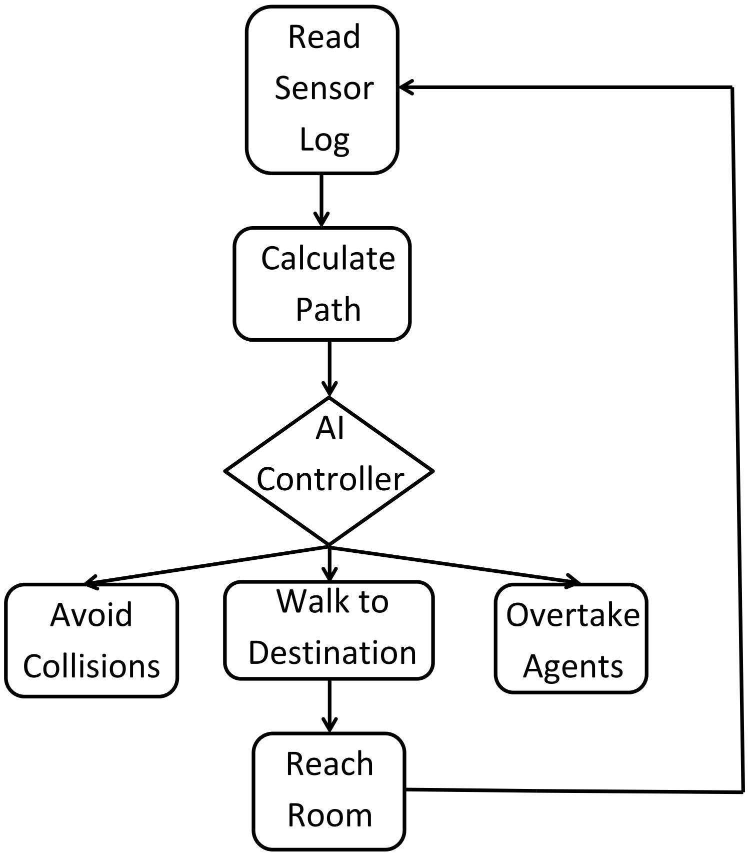
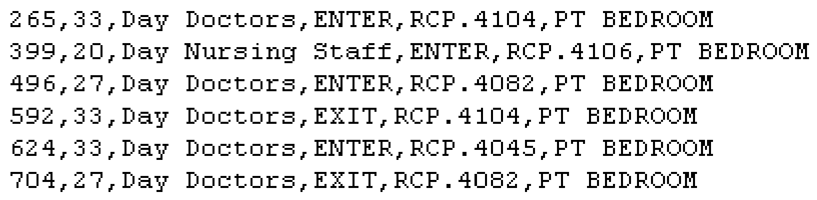


Figure 3.1: A flowchart depicting the various components of the framework, and how they affect each other.

Generally, the framework reads in all the sensor log data into memory in order to prevent any delays caused by disk input/output during simulation time. Second, the framework processes the log data associating a virtual agent with each worker in the sensor log data. The framework is also able to determine the start and end position of each agent in the simulation at this phase. Next, the framework finds a path from each agent's current location to the destination for the agent. The AI controller controls the movement of the agent at each simulation step in the simulation. The controller composes the various behaviors such as path traversal, collision avoidance and overtaking in order to get the net movement of each agent at every frame of the simulation. The framework repeats its procedures when the virtual agent reaches its destination.

## Reading Human Movement Data

Using a distributed sensor network, human movement data was recorded over a period of two weeks at the Carver Pavilion of the University of Iowa Hospital. The sensor network consisted of a radio attached to badges worn by healthcare workers, a radio attachment on patient beds, and radios on alcohol based soap dispensers. The sensor network uses the strength of the signal received from the radio attachments on the patient beds and badges in order to determine proximity. The system has the ability to discriminate short distances between patient beds and worker badges and long distances between them. Thus, the data collected by the sensors only includes instances when the workers are close to the stationary attachments on patient beds [12]. The sensor network data was provided in a flat file as seen in Figure 3.2, and every file consists of human movement over the period of a 12 hour shift. The fields of the log data are timestamp in seconds from beginning of the shift, a unique identification for every human that is tracked by the sensor network, a description of the role that the tracked person plays in the environment, the type of event that was logged, the location of the sensor that logged the event, and a description of the location of the sensor. The current sensor network data only records when a person either enters or exits the range of the sensor. Using the sensor log data, the framework is able to generate traffic patterns and low level behaviors for the traffic patterns that mimic the traffic in the real world.



```
265,33,Day Doctors,ENTER,RCP.4104,PT BEDROOM
399,20,Day Nursing Staff,ENTER,RCP.4106,PT BEDROOM
496,27,Day Doctors,ENTER,RCP.4082,PT BEDROOM
592,33,Day Doctors,EXIT,RCP.4104,PT BEDROOM
624,33,Day Doctors,ENTER,RCP.4045,PT BEDROOM
704,27,Day Doctors,EXIT,RCP.4082,PT BEDROOM
```

Figure 3.2: An excerpt from one of the sensor log files that is used to generate the traffic for the virtual agents.

Each virtual agent in the framework has a component for reading sensor log data. This component reads the file into memory and filters out the events from the log file that pertain to the agent that the component operates on, i.e. if the log reader is associated with a virtual agent that is responsible for mimicking the actions of agent number 40 in the sensor logs, then the log reader filters out the events for only agent number 40. The assumption for the filtered data is that the sensor log data for every virtual agent alternates between entrance and exit events. At every simulation frame, the sensor log reader compares the current simulation time with the timestamp of the next event in the order of events for the sensor log reader on that particular virtual agent. If the current simulation time is past the timestamp of the next event, the sensor log reader instructs the framework act on the next event in the sensor log and then advances its pointer in the sensor log. The sequence of actions that the framework performs depends on the type of event that the entry in the sensor log represents. Currently on enter and exit events are handled. If the next event in the sensor log data is an exit event, the framework notes the timestamp of the exit event and marks the event as processed. The timestamp of the exit event is used to calculate the speed at which the virtual agent needs to travel in order to reach its next destination on time. After processing an exit event, the sensor log pointer is advanced, and the sensor log reader instructs the framework to calculate a path from the virtual agent's current position to the destination indicated in the enter event that the sensor logs points to as the next event in the series. The speed of travel for the virtual agent is calculated based on the difference in the timestamp of the last exit event that was processed for the agent and the timestamp of the upcoming enter event along with the travel distance between the virtual agent current position and the destination of the virtual agent. Enter events require no processing by the framework except to advance the pointer in the sensor log reader. However, the framework can be passed in function pointers in order to do custom code execution when an enter event is processed. This allows for other types of behaviors to be layered on top of the low level behaviors that are generated by the framework such as behaviors that might interact with the environment.

## Path Determination

The sensor log data files provide the destinations for each of people that were tracked by the sensor network. In order to represent the behaviors of these people accurately in the virtual environment, the virtual agents associated with these people need to be able to find a reasonable route from one destination to another in the sensor log data. A reasonable route for a person to take is assumed to be a direct route to the person's destination.

To allow the framework to be used on any complex public indoor environment, we used waypoints in order to define the paths in the environment. A waypoint graph was made from the waypoints used to define the traversable parts of the environment. In this waypoint graph, a node represents an intersection between two or more traversable paths in the environment. The edges in the graph indicate that there is a traversable path between the two given nodes. The framework is capable of reading in the node locations from a strictly defined XML file. Once the nodes of the waypoint graph are created, we can create edges between the waypoint nodes based on clear line-of-sight between any two given nodes. The framework considers every pair of nodes, checks if there is a clear line of sight between the pair of nodes, and then creates an edge if there is a clear line of sight. A clear line of sight between two nodes can be determined in a 3D virtual environment if there are no triangles from the mesh describing the environment that intersect the line segment defined by the two nodes. Thus, the framework generates a waypoint graph that describes traversable parts of the environment if given the intersections between paths in the environment and the environment geometry. Figure

3.2 shows the waypoints that compose the traveling portion of an environment and the edges that connect the waypoints to form the graph.

A weighted directed cyclic graph was constructed of all the possible paths by augmenting the basic waypoint graph with weights on each edge that denote the distance between the nodes that the edge connects in the virtual environment. This data structure allows for many different path determination algorithms from a given point to a given destination. Since we assume that people take the most direct route to their destination, we used Dijkstra's shortest path algorithm to compute the ordered list of waypoints that would need to be visited in order for the virtual agent to travel from its current position to its destination location [13].

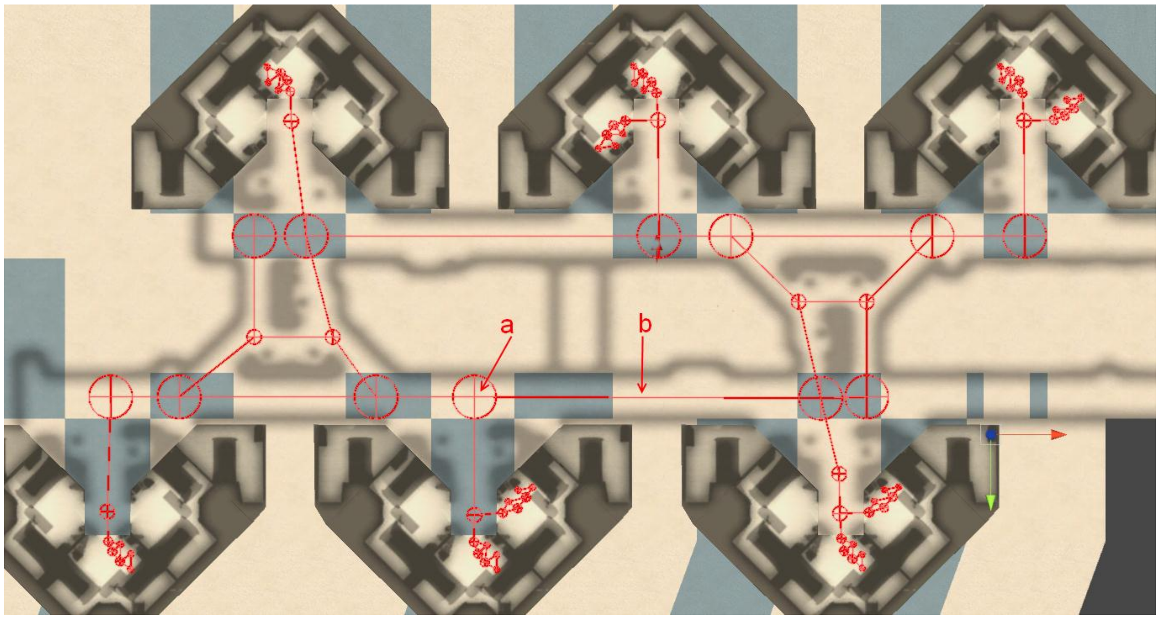


Figure 3.3: A waypoint network to traverse the corridors in this environment. The waypoint nodes (a) are placed at the intersections of paths and edges (b) connect waypoint nodes that have clear line of sight between them

Every virtual agent in the environment contains a component that allows the agent to plan its path to its destination. This component takes the weighted directed cyclic graph as its input and calculates the ordered list of waypoints that need to be reached in order to reach the destination waypoint. Since the virtual agent's position at the moment of path calculation is not guaranteed to be co-located with a waypoint in the waypoint graph, the planner uses closest waypoint to the associated agent in the virtual environment using the Euler distance between the virtual agent and a given waypoint node. The speed of walking for each agent is determined by the timestamp difference between the entrance and exit events for the agent and the sum of the distances for each edge that has to be travelled on the Dijkstra path to the virtual agent's destination.

Since the agents can walk at different speeds, we have to change the walk animation of the agents so that there is no foot slippage. Foot slippage is the condition when the walking animation of a virtual agent does not match the speed of walking of the agent such that the agent appears to be floating through the environment. Preventing foot slippage is important because animation fidelity has an impact on copresence. Garau, et. al. showed that the degree of animation fidelity impacts the sense of presence and copresence that users feel in an environment [17]. Jörg, et. al. demonstrated that the quality of the animation has an impact on the user's understanding of the scenario [18]. In order to prevent foot slippage, the rate of playback of the walking animation is also varied while the stride length of the agent's walk animation remains unchanged. Empirically, we found that varying the rate of playback of the walk animation by a linear factor was enough to prevent any noticeable foot slippage in the range of walking speeds that are possible for humans, i.e. 1.0 meters per second to 3.0 meters per second.

## Directional Traffic

A waypoint based path determination algorithm generally only determines the order that the waypoints have to be visited in order to reach the destination. This algorithm, if used without modification, would cause the virtual agents to walk in the center of every path while travelling to the destination since the waypoints are placed at the center of each intersection of multiple paths. This behavior is not realistic and does not accurately depict how people travel in closed spaces. People generally tend to travel towards one side of a pathway when walking to a destination, forming two lanes of opposing traffic direction [14]. In order to replicate this behavior without unnecessary duplication of the waypoint, one for each direction of traffic, a method was devised that allows the virtual agents to travel at an offset from the direct path between any two given waypoints. This method is combined with the direct walk method in order to get a combined path walking algorithm that is reflected in the "walk to destination" node in the overview in Figure 3.2.

Not all paths formed in the virtual environment pass through regions where it is possible to form two lanes of traffic. In order to accommodate this property, each waypoint contains a variable that determines whether the edges connecting to and from that waypoint should be traversed using lane traffic or simply a direct route. The framework can automatically decide whether a given waypoint should allow lane traffic or not based on a threshold on the amount of space there is around the waypoint node provided in the XML input data file. If the last waypoint node that the virtual agent passed and the waypoint node that the virtual agent is headed towards both allow for two lanes of traffic, then the virtual agent walks on a path that is offset from the direct path by half the shoulder width of the virtual agent. Otherwise, the virtual agent walks directly towards the next waypoint. Figure 3.4 shows how a virtual agent would transition on and off the offset path in traveling to a destination waypoint.

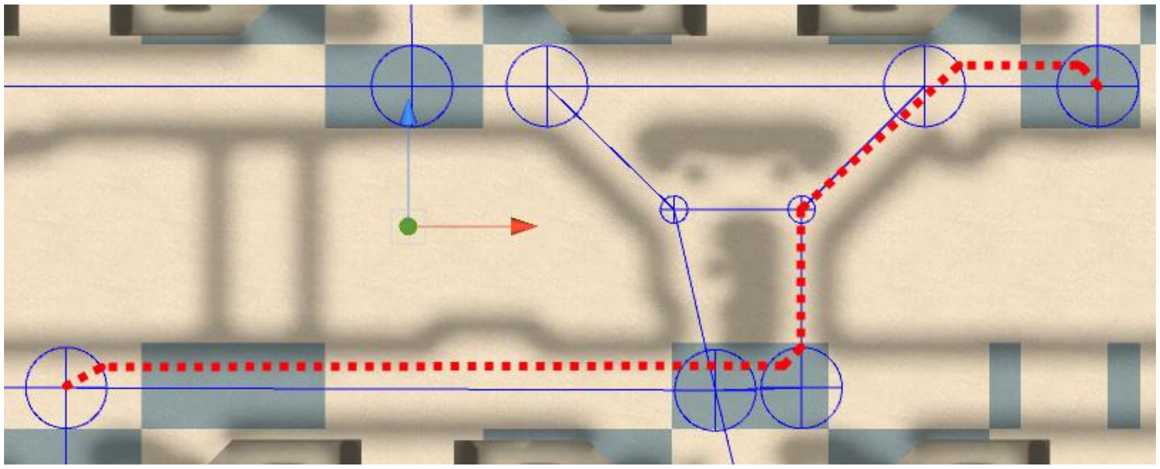


Figure 3.4: A virtual agent plans a path between two waypoints. The solid blue line shows the path the virtual agent would take if it walked directly between each waypoint. The dotted red line shows the path the agent takes if it is permitted to walk on an offset path.

The directional traffic algorithm can be broken down into 3 phases. The virtual agents transition from their direct path onto the offset path in the first phase. The merge into the offset path is accomplished by assuming that the ideal path is formed by a plane consisting of the location of the waypoint node that the virtual agent is walking towards, the previous waypoint node that the agent passed, with the normal of the plane facing in the direction of the offset, orthogonal to the traffic flow direction. In order to calculate the plane composing the offset path, the plane for the ideal path is simply translated by the offset in the direction of the normal of the plane. In the first phase, the virtual agent walks towards the plane for the offset path at a maximum angle of 45 degrees from the ideal path until the virtual agent is walking on the offset path. The walking direction of the virtual agent is simply Normalize(n * d + p) where n is the direction that the ideal walk plane was translated in order to achieve the offset walk plane, d is the distance between the virtual agent and the offset walk plane, and p is a vector that is parallel to the ground and lies in the ideal walk plane.

The second phase of the directional traffic algorithm consists of the virtual agent walking on the offset path. So, the virtual agent's walk direction can simply be computed as the direction Normalize(p) where, as before, p is a vector that is parallel to the ground and lies in the ideal walk plane. In the third phase, the virtual agent merges from the offset path onto the ideal path. This is simply the opposite direction from the first phase defined as Normalize(-n * d + p).

In order to implement smooth turns for virtual agents so that the agents are not merging back onto the ideal path at every waypoint, a waypoint was considered to be reached if the agent came within a hit radius of the waypoint. The hit radius is the same radius that the framework calculates for every waypoint node to determine whether lane traffic should be allowed near the waypoint. This hit radius indicates how much clearance there is around a given waypoint node. This smooth turning change is what accounts for the early turns seen in Figure 3.4 when the agent transitions from offset path to ideal path and vice versa.

## Collision Avoidance

Collision avoidance is necessary within the framework because it is possible for virtual agents to be walking in opposite directions on a path that is not marked for lane traffic. In this case, both virtual agents would be walking towards each other, walking in the center of the path linking the two waypoints. Without collision avoidance, the agents would collide, which is not a realistic behavior.

Collision avoidance was accomplished by creating a conical bounding volume associated with each virtual agent that is present in front of each virtual agent. When two agents are in close proximity of each other and are about to collide, the conical bounding volume of at least one of the virtual agents intersects the other virtual agent. This imminent collision is avoided in the framework by using a right first approach. Whenever there is an imminent collision in the framework, every agent detecting the imminent collision veers off to the right in order to avoid the collision as shown in Figure 3.5. This collision avoidance algorithm can be represented simply as walkDirection is Normalize(n + p) if conicalVolume intersects the position of the agent, otherwise, walkDirection is Normalize(-n * d + p). In these expressions, n is a vector in the direction that the agent needs to move in order to avoid collision (normal of the ideal walk plane), d is the distance to the ideal path plane from the agent, and p is the vector that is parallel to the ground and contained within the ideal path plane.

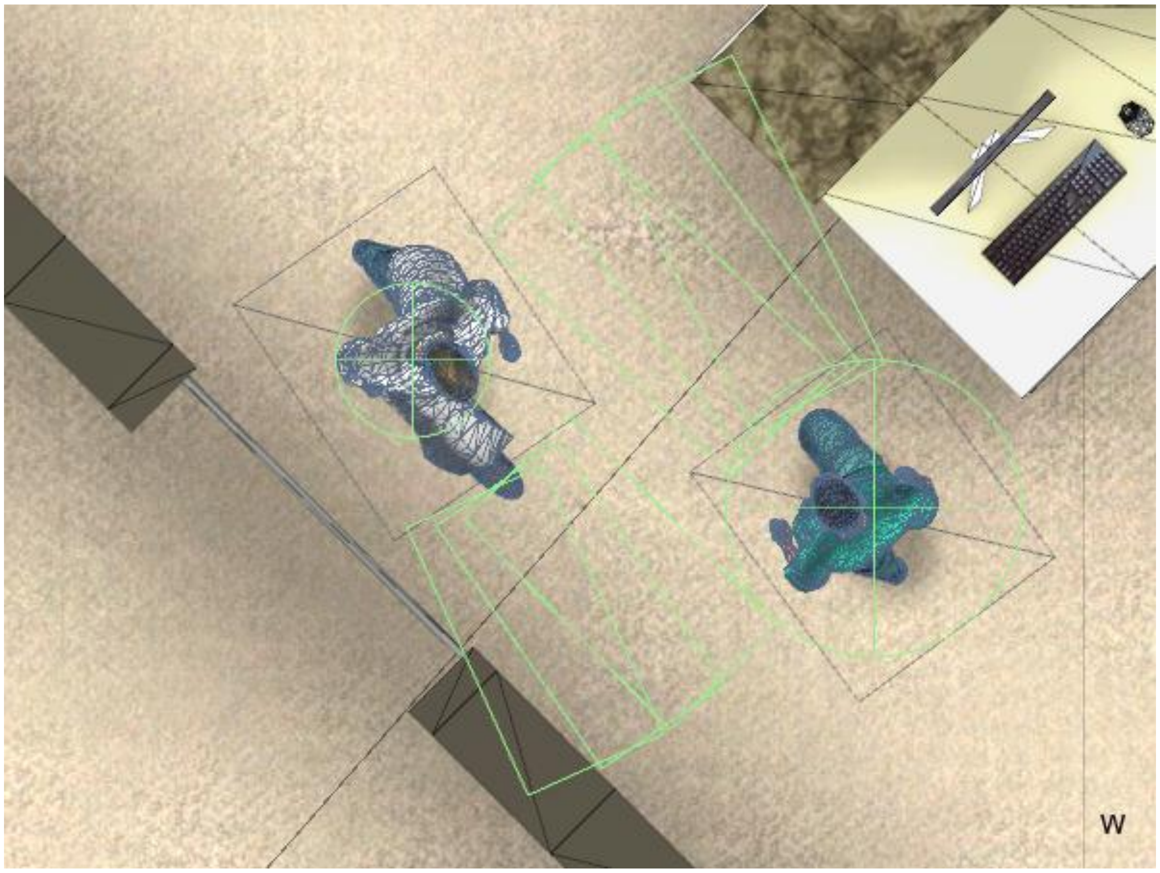


Figure 3.5: Two agents avoid colliding with each other in a narrow pathway that doesn't allow two-lane traffic. The conical bounding volumes of both agents are visible that allow for the collision avoidance.

Since this method assumes that every agent avoids collision to the right, this method is not able to reproduce the shuffle that takes place when people avoid collision in the same direction, stop to avoid collision and change direction to avoid collision again. However, this method is guaranteed to avoid collision between two mobile virtual agents. Unlike reciprocal velocity obstacles (RVO), this method of collision avoidance only perturbs the direction of travel for an agent instead of changing the direction and speed of travel in RVO. RVO also depends on all the agents in the simulation following the same collision avoidance algorithm. Thus, if an agent is stationary and not executing collision avoidance, RVO can result in a collision since the stationary agent does not perform its responsibility of collision avoidance [11]. However, the proposed algorithm of collision avoidance is able to avoid this case by not expecting reciprocal collision avoidance. Simultaneous collision avoidance by both agents can reduce the amount of modification needed in the path of the agents; however, this condition is not required.

## Overtaking

Overtaking is a necessary behavior in the framework because the walking speed of each virtual agent is determined by the timestamps in the sensor log data. Therefore, it is possible that an agent walking at a faster speed might end up behind an agent walking at a slow speed. In order to prevent the faster agent from bumping into the slower agent and also to keep the faster agent from falling behind schedule, an overtaking behavior is provided in the framework. Some challenges that we overcame with this solution are that the overtaking behavior could not prevent the agents from following their path. Furthermore, the overtaking behavior should not affect the opposing lane of traffic in two-lane traffic paths.

Overtaking proceeds in three phases: approach, pass and merge. In the approach, the faster agent walks up to the slower agent. When the faster agent is sufficiently close, the collision avoidance behavior prevents a collision between the two agents. This behavior is inverted so that the collision avoidance happens to the left, i.e. the faster agent pulls to the left of the slower agent. In the passing phase, the faster agent speeds up in order to overtake the slower agent swiftly and walks parallel to the slower agent. Once the faster virtual agent is beyond the conical bounding volume of the slow agent, the merge phase begins in which the faster agent slows back down to normal speeds and merges back onto its normal path at a shallow angle to prevent collisions with the slower agent. This algorithm can be summarized as walkDirection is Normalize(-n + p) if the conicalVolume intersects another agent, p if the position of the agent is between the position of the slow agent and the furthest point on the slow agents conical volume, and Normalize(n * d + p) otherwise where n is a vector in the direction that the faster agent needs to move in order to avoid collision, p is the vector parallel to the ground and contained within the ideal walk plane, and d is the distance between the faster agent and the ideal walk plane. In this algorithm, the first second and third branches mimic the three phases of overtaking.

## Combining The Behaviors

Each behavior described that the virtual agent exhibits using the framework is executed in parallel and independent of any other behavior that may be exhibited by the agent. Each behavior has the possibility of generating a direction that the agent should walk in. Each behavior can also have a modifier on the speed of walking for the agent, such as the overtaking behavior requires the agent to speed up in the passing phase. In order to combine all the various directions and speeds into one walking direction and speed for the virtual agent at each frame of the simulation, the framework generates an average walk direction by adding up the vector that signifies the walk direction for every behavior and normalizing the resulting vector. Each behavior also returns a speed modifier. The resulting speed is calculated by multiplying all the speed modifiers into the speed of the agent. In this manner no single behavior completely controls the agent. The agent continues to follow its directional traffic flow path even while it is in the act of overtaking a slow agent.

## Evaluation of the Framework

The framework's performance was evaluated at a high level in order to verify whether the behaviors generated by the framework and computational complexity of the framework could be computed at a reasonable rate on a desktop computer. Thus, a framerate analysis of the framework was conducted where the number of agents in the environment that were controlled by the framework was varied to see the effect on the number of frames rendered every second. The simulation was run for one and a half minutes with the specified number of agents and the framerate was measured every half second over that period. A graph of the results is presented in Figure 3.6. The evaluation was performed using Unity3D on a computer with Intel Core i7 CPU 920 @ 2.67 GHz with 12 GB of RAM and a GeForce GTX 260 graphics card on a Windows 7 (64-bit) operating system.

A top-down view of the environment was rendered when performing this evaluation of the framework so that the entire environment could be viewed at once and the multiple agents that are controlled by the framework would all be in the frame. The benchmarks were recorded without any level of detail optimizations or occlusion culling or frustum culling of the environment or agents. Every agent and the environment were rendered with all the rich details present in the 3D models. We wanted to evaluate the framework in the context that it would be used, with all the aspects enabled so that the performance evaluation would be accurate to its use case scenarios.

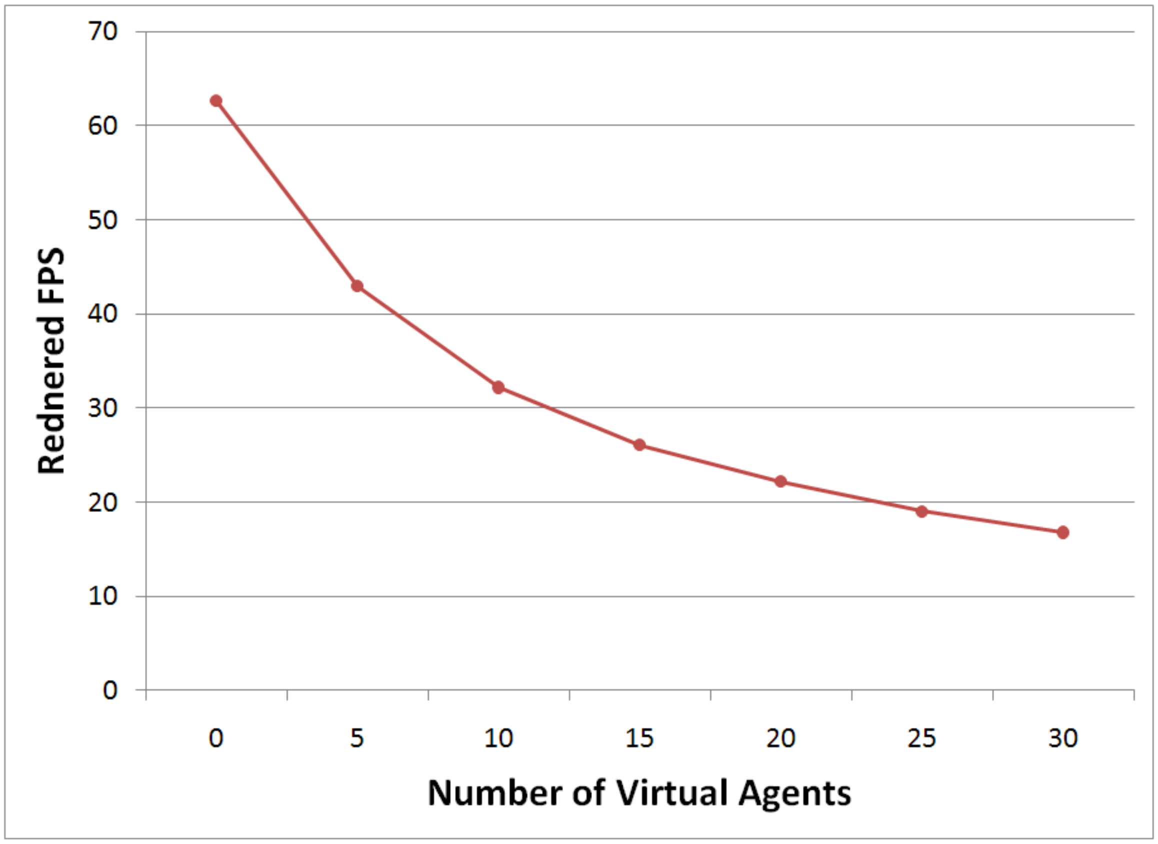


Figure 3.6: A graph of the average rendering rate of the simulation vs. the number of virtual agents that were controlled by the framework. The data most closely fits a power curve as the number of virtual agents is increased.

As seen in Figure 3.6, the graph shows framerate of the simulation with the varying number of agents. The first 30 seconds of the simulation were excluded to prevent starting positions of the agents from biasing the results and only recording typical performance under that agent load. The curve that is formed as the number of agents is increased most closely fits a power curve. The power curve is due to the fact that with each additional agent, the number of additional polygons that need to be drawn in order to render the entire simulation become more and more insignificant with respect to the number of polygons needed to draw the rest of the scene. The performance drop that is experienced as the number of agents is increased is mainly due to the number of triangles that have to be calculated and drawn by the graphics framework for every agent that is added to the simulation. Performing this analysis by abstracting out each agent in the simulation to a sphere would not be an accurate analysis because the framework is meant to be used in virtual environment simulations. The framework depends on some input that can only be provided by a detailed and complex environment and detailed, non-abstracted agents like foot slippage synchronization.

Furthermore, this performance measurement is taken from a birds-eye perspective of the environment. However, in virtual environments, the common view of interaction is the first person perspective. In those cases, the framerate for the simulation with 10 agents was well over 100 frames per second because of the possibility of occlusions, frustum culling and other optimizations that are achieved from the introduction of the first person perspective.

# CHAPTER FOUR

# APPLICATION AND ASSESSMENT

We needed to be able to quantify the effectiveness of the framework in creating believable, realistic and engaging scenarios that could be accepted by the user of the framework as an acceptable portrayal of reality. The application that we built was a pedagogical simulation built to train healthcare workers the Five Moments of Hand Hygiene. The Five Moments of Hand Hygiene is a protocol that was developed by the Center for Disease Control (CDC) along with World Health Organization (WHO) in order to mitigate the spread of infections in hospital settings [15].

## The Five Moments of Hand Hygiene Application

The training simulation can be divided up into three different phases. In the first phase, the introduction phase, the user is introduced to the Five Moments of Hand Hygiene procedure and given instructions about the simulation controls and directions on the evaluation criteria for the performance of the user within the simulation. In the training simulation, the user plays the role of a health inspector in a hospital ward where the user is instructed to follow various virtual healthcare workers in the environment and record their compliance with the Five Moments of Hand Hygiene procedure. The users are instructed to record as many infringements to the procedure as possible in the scenarios that are presented by the simulation.

Once all the instructions have been disseminated to the user, the interactive phase begins. In the interactive phase, the user is allowed to freely roam the hospital ward in the virtual environment. However, as stated earlier, the objective for the trainee is to record as many infringements to the compliance of the hand hygiene procedure as possible within a five minute period. Since the users are told to follow various agents, a large portion of the time that the users spent in the simulation is spent watching the low level behaviors that are generated by the framework as the virtual agents are traveling to their next destination to interact with a virtual patient.

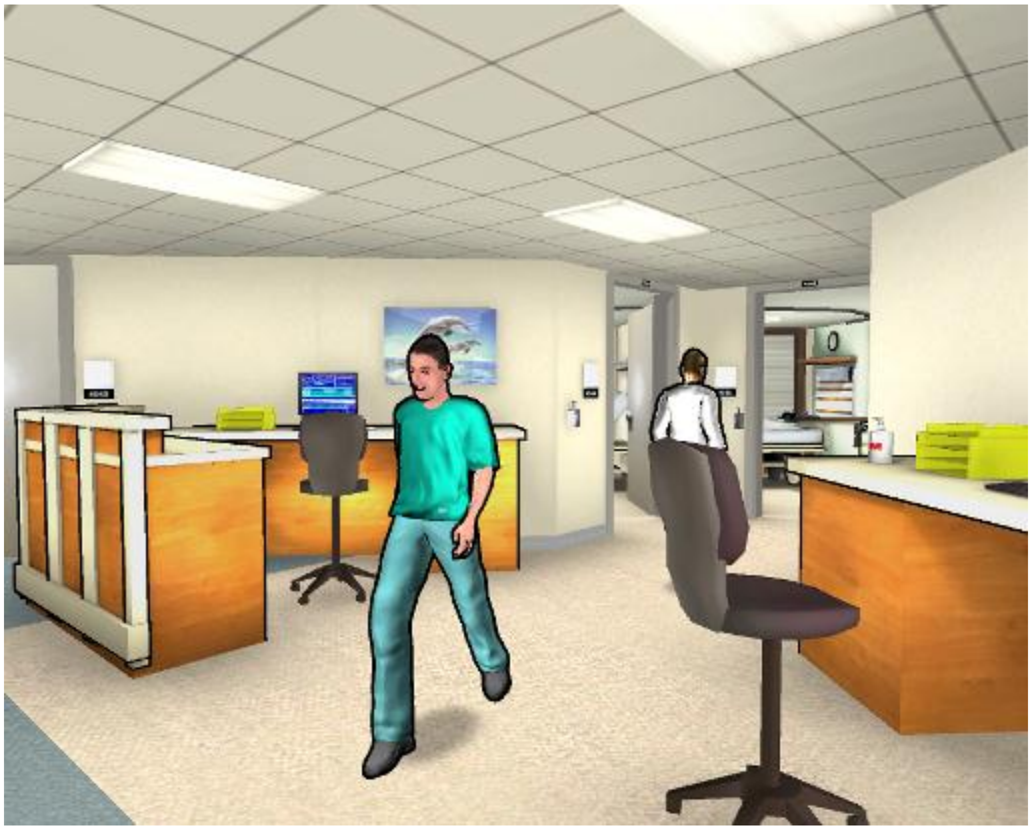


Figure 4.1: Once the user starts the simulation and enters the interactive phase, the user is allowed to follow any health care worker and observe its compliance with the hand hygiene protocols.

In the final phase, the feedback phase, the user is given feedback about his or her performance in the simulation. The user is presented with the number of infringements to the hand hygiene procedure that they recorded correctly. The user is given the option to restart the simulation in case they want to revisit the training exercise for better performance.

## User Study

At the end of the training simulation, the users from the usability study are presented a survey that asks them to rate the simulation on the realism, engagement, believability, interactivity, ease of use and effectiveness in teaching the task. The survey also collected general comments and feedback regarding the user's experience in the each of the fields listed above. You can see all the questions that were posed to each user in the survey in Figure 4.2. The questions in the survey are adapted from the questions asked in a study to evaluate the engagement, responsiveness and humanness of an embodied conversational agent [16].

The simulation was evaluated by six full-time nurses and eighteen nursing students. We restricted the population of the usability study to only people in the medical field so that the hand hygiene procedure would not be as difficult to learn and the evaluation would reveal flaws in the behaviors generated by the framework. Figure 4.3 shows a summary of the responses on the questions answerable on the Likert scale.

As seen in the figure, the training simulation is rated above average in the qualitative aspects of the simulation like engagement, realism, believability, interactivity, ease of use and effectiveness of training. In particular, the low level behaviors generated by the framework in were believable for the users of the application. The users did not report any foot slippage nor any strange behavior exhibited by the virtual agents.

| | |
|---|---|
| 1. | On a scale of 1(not engaging) to 10(highly engaging), please rate how engaging the hand hygiene training simulation is |
| 2. | On a scale of 1(not realistic) to 10(realistic), please rate the realism of the training scenarios in the simulation |
| 3. | On a scale of 1(not believable) to 10(highly believable), please rate the believability of the scenarios consisting of the virtual healthcare workers and patients |
| 4. | On a scale of 1(not interactive) to 10(highly interactive), please rate the interactivity of the training simulation |
| 5. | On a scale of 1(very easy to use) to 10(very hard to use), please rate the usability of the training simulation |
| 6. | On a scale of 1(strongly disagree) to 10(strongly agree), please rate your impressions of how effective this training simulations of how effective this training simulation was in teaching proper hand hygiene protocols |
| 7. | What are some medical procedures and best practices besides hand hygiene that can be taught via such interactive virtual reality training simulation? |
| 8. | What did you like most about this training simulation? |
| 9. | What did you like least about this training simulation? |
| 10. | What are some ways in which this training simulation can be improved? |

Figure 4.2: The questions that were asked on the survey. With each question that asked to rate the scenario on a Likert scale, the user was allowed to leave feedback in a comment section associated with each question.

In addition to the scores in the various categories of usability, we also received many comments from the users of the study. The users mostly found the simulation engaging but something about the environment struck them as odd. Users commented that there needed to be more activity in the virtual hospital ward. One user commented, that we "Need more people in the nursing station, halls, and activity…perhaps a group of professionals making rounds, secretaries in nursing station, etc." Another commented that "the basic hospital setup seemed pretty real, but the simulation needed more people present. Many of the rooms were empty, there were no visitors, and there were few employees present." Furthermore, another user voiced that "You would not find a nursing station without folks milling around, secretaries, and a variety of staff…nutrition, housekeeping, laboratory…depending on time of day."

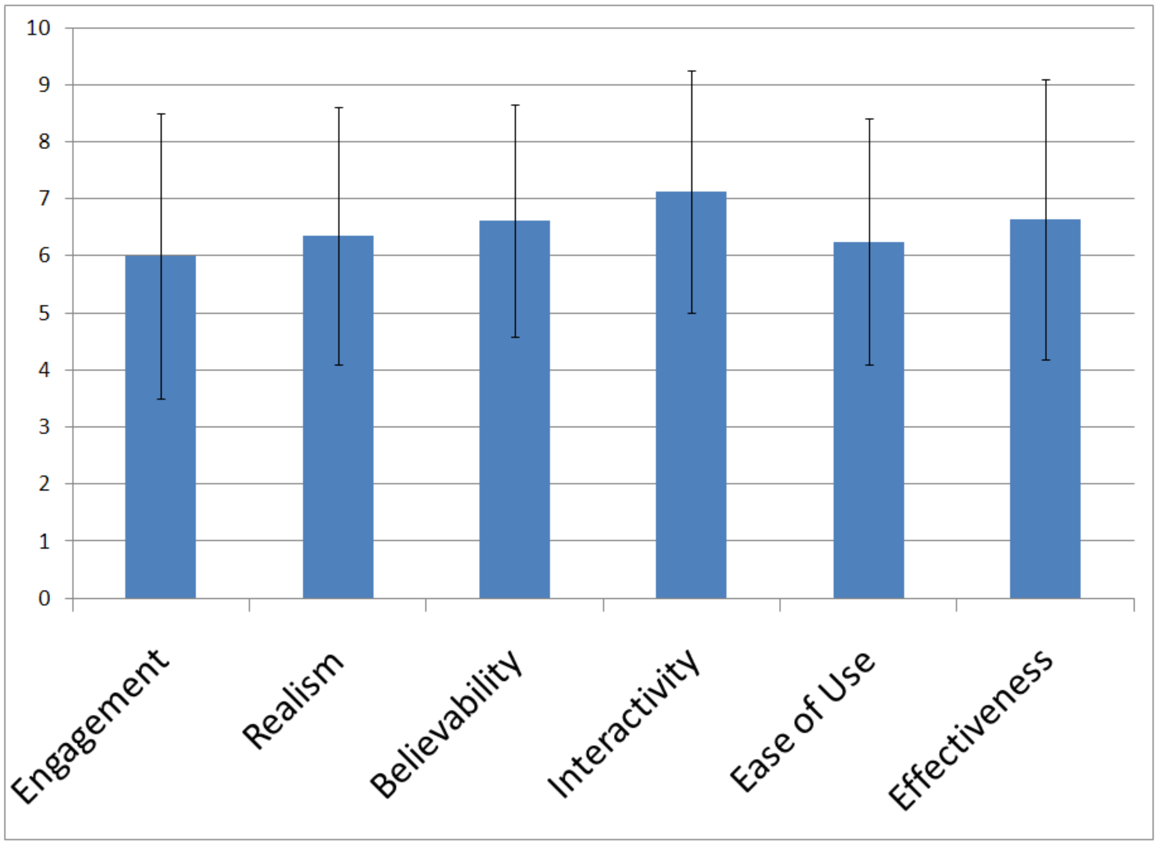


Figure 4.3: The results of the user study showing the mean and the standard deviation of the scores given in the various usability aspects of the training application.

The users of our simulation noticed a sparseness of virtual entities in the environment even though the number of agents in virtual environment is the same as the number of people that were recorded by the distributed sensor network. This sparseness is due to the fact that the distributed sensor network at the University of Iowa only recorded the position of only doctors and nurses. It does not record visitors and other staff in the hospital. It also only records activities, so, it is unable to record idling workers. Since this framework is based on sensor log data, we can either artificially increase the number of doctors and nurses in this environment to make the virtual ward seem more busy. However, the better solution is if we are able to track a higher percentage of the people that are present in the ward so that the traffic patterns generated by the framework remain realistic and are not simulated.

A difficulty that we noticed when the users were navigating the virtual environment was the users found it difficult to move around corners using the keyboard controls we provided them. Also, the users tended to get lost in the virtual environment because they failed to form a mental map of the hospital due to the difficulty in controls and the complex but symmetrical layout of the hospital. This loss of navigability resulted in the users finding it difficult to find healthcare workers that they could observe. Possible improvements to alleviate the problem of forming mental maps are suggested in the future work section.

However, overall, the responses from the users suggest that the simulation is realistic and believable which means that the low-level behaviors generated by the framework can be accepted by the users.

# CHAPTER FIVE

# CONCLUSIONS AND FUTURE WORK

We present an extensible framework for making multi-agent simulations. The research presented here comprises of at least two different key contributions.

## Conclusions

The key contributions of this research include a framework that can automatically generate low-level behaviors for a multi-agent simulation or a visualization. The behaviors generated include low level behaviors such as path planning, directional traffic rules, collision avoidance and overtaking other agents. The framework also provides ways to prevent foot slippage when the speed of agents can be varied. This framework provides the low level behaviors in a manner such that higher level interactive behaviors can be layered on top to give a better model of a person.

Furthermore, we present a method that allows for the visualization of human movements recorded in sensor log data by distributed sensor networks. The framework presented here allows for visualization of the captured data and is able to fill in the missing pieces in the sensor data in order to present an agent that smoothly travels from one destination to another, arriving at each destination on time. We develop an application using the framework presented here that trains users in the five moments of hand hygiene procedure. This application uses multiple agents that are driven by sensor log data captured at real hospital setting. We conducted an initial usability study to determine whether the behaviors generated by the framework were believable by a user.

The quantitative results of the study suggest that the low-level behaviors generated by the framework are plausible to most users of the system. Further analysis of the comments revealed some weaknesses of the environment that we used for the application and some weaknesses in the framework that can be addressed in future work.

## Future Work

The usability study of our framework revealed that there were users who found it difficult to form a mental map of the environment due to their troubles operating the navigation controls and other similarities in different parts of the environment. This lack of a mental map made it harder for these users to find virtual health care workers to observe. More work on this problem is being conducted by adding wayfinding aids to the simulation that present the immediate surroundings of the user in the virtual environment. In this wayfinding aid, the user is able to spot various healthcare workers, patients and also the layout of their immediate surroundings. The aid also contains a pointer that signifies the direction that the user is facing in the virtual environment. This approach to alleviating problems with mental map formation should enhance the user's spatial awareness and allow them to quickly spot areas that the virtual healthcare workers are visiting.

The users of the training application for the five moments of hand hygiene also complained about the lack of people in the hospital environment. The users felt that the environment felt empty and deserted. We plan to leverage the additional data that the University of Iowa hospital's evolving distributed sensor network is able to capture in order increase the amount of people present in the virtual hospital. In future work, we could also add sequestering algorithms so that healthcare workers, from the sensor log data, that take an extraordinary amount of time to reach a nearby destination, can idle nearby or perform some other task in the vicinity of the user.

The performance evaluation of the framework was conducted in the top down view with all the optimizations disabled. Another evaluation of the framework is needed to ascertain the effect of increasing the number of agents on the performance of the system. This evaluation would be performed from a first person perspective so that the performance characteristics collected would show the real world performance in the intended use cases of the framework.

The framework needs to be evaluated under high agent loads. Evaluation under high agent loads would identify any emergent behavior that is manifested as a product of the interaction between many agents with different emphases on the behaviors in the framework. High agent loads would also identify instances when the component algorithms making up the low level behaviors of the framework fail and cause undesired behaviors such as collisions between oncoming agents, collisions during overtaking, collisions with the environment or loss of pathfinding.

The framework currently has the ability to generate waypoint nodes and edges based on a given environment geometry and a file that describes the location of the waypoint nodes. Future work would automate the generation of the information dictating the location of the waypoint nodes so that simply the knowledge of the environmental geometry would be enough to find waypoint nodes in order to traverse the entire environment using the framework's path determination algorithm.

## APPENDICES

## Appendix A

## Sample XML File for Inputting Waypoint Locations into the Framework

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<waypoints name="Hall" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="WaypointDescription.xsd">
	<position>
		<x>12.34626</x>
		<y>-3.092425</y>
		<z>6.179554</z>
	</position>
	<waypointSettings>
		<radius>1.0</radius>
		<radius>1.04</radius>
		<radius>1.13</radius>
		<radius>1.25</radius>
		<radius>1.5</radius>
		<twoWayHallThreshold>1.2</twoWayHallThreshold>
	</waypointSettings>
	<waypoint>
		<waypointName>Waypoint1</waypointName>
		<position>
			<x>2.209666</x>
			<y>0.0</y>
			<z>-12.15279</z>
		</position>
	</waypoint>
	<waypoint>
		<waypointName>Waypoint2</waypointName>
		<position>
			<x>-24.99734</x>
			<y>0.0</y>
			<z>-12.15279</z>
		</position>
	</waypoint>
</waypoints>
```

## REFERENCES

[1] Bertrand, J., Babu, S.V., Polgreen, P., and Segre, A. Virtual Agents Based Simulation for Training Healthcare Workers in Hand Hygiene Procedures in *Lecture Notes in Computer Science: Proceedings of the 10th International Conference on Intelligent Virtual Agents (IVA 2010)*, Springer Verlag Berlin/Heidelberg, Vol. 6395, pp. 125-131.

[2] Bertrand, J., Babu, S.V., Gupta M., Segre, A.M., and Polgreen P. A 3D Virtual Reality Hand Hygiene Compliance Training Simulator in *Proceedings of the 2011 Annual Scientific Meeting of the Society of Healthcare Epidemiology of America*.

[3] Gupta, M., Bertrand, J., Babu, S.V., Polgreen P., and Segre, A. An Evolving Multi-Agent Scenario Generation Framework for Simulations in Preventive Medicine Education in *Proceedings of the 2nd ACM SIGHIT International Health Informatics Symposium (IHI 2012)*, ACM, New York, NY, pp. 237-246.

[4] Narain, R., Golas, A., Curtis, S., and Lin, M. C. Aggregate Dynamics for Dense Crowd Simulation. *ACM Transactions on Graphics*, Vol. 28, pp. 122:1-122:8.

[5] Shao, W., and Terzopoulos, D. Populating Reconstructed Archaeological Sites with Autonomous Virtual Humans. *Proceedings of the 6th International Conference on Intelligent Virtual Agents*, pp. 420-432.

[6] Rizzo, A., Parsons, T.J., Buckwalter, J.G., and Kenny, P. A New Generation of Intelligent Virtual Patients for Clinical Training. *Proceedings of the IEEE Virtual Reality Conference*, 2010.

[7] Gomez, L., Laube, A., and Ulmer, C. Secure Sensor Networks for Public Safety Command and Control System. *IEEE International Conference on Technologies for Homeland Security*, pp. 128-136.

[8] Sud, A., Andersen, E., Curtis, S., Lin, M., and Manocha D. Real-time Path Planning for Virtual Agents in Dynamic Environments. *2007 IEEE Virtual Reality Conference*, pp. 91-98.

[9] Ming, L.C., Sud, A., Van den Ber, J., Gayle R., Curtis, S., Yeh, H., Guy, S., Andersen, E., Patil, S., Sewall, J., and Manocha, D. Real-time Path Planning and Navigation for Multi-agent and Crowd Simulations. *Motion in Games*, LNCS 5277, pp. 23-32.

[10] Karamouzas, I., and Overmars, M. A Velocity-Based Approach for Simulating Human Collision Avoidance *Lecture Notes in Computer Science: Proceedings of the 10th International Conference on Intelligent Virtual Agents (IVA 2010)*, Springer Verlag Berlin/Heidelberg, Vol. 6395, pp. 180-186.

[11] Van der Berg, J., Patil, S., Sewall, J., Manocha, D., and Lin, M. Interactive Navigation of Multiple Agents in Crowded Environments. *ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, 200.

[12] Herman, T., Pemmaraju, S., Segre, A., Polgreen, P., Curtis, D., Fries, J., Hlady, C., and Severson, M. Wireless Applications for Hospital Epidemiology. *Annual Scientific Meeting of the Society for Healthcare Epidemiology of America*.

[13] Dijkstra, E. W. A Note on Two Problems in Connexion with Graphs. *Numerische Mathematik*, Vol. 1, pp. 269-271.

[14] Helbing, D., Buzna, L., Johansson, A., and Werner T. Self-organized pedestrian crowd dynamics and design solutions: Experiments, simulations and design solutions. *Transportation Science*, Vol. 39, pp. 1-24.

[15] Pittet, D., Allegranzi, B., and Boyce, J.M. The World Health Organization Guidelines on Hand Hygiene in Health Care and Their Consensus Recommendations. *Infection Control and Hospital Epidemiology*, Vol. 30, pp. 611-622. (2009)

[16] Babu, S., Schmugge, S., Barnes, T., and Hodges, L.F. "What would you like to talk about?" An Evaluation of Social Conversations with a Virtual Receptionist. *Lecture Notes in Computer Science: Proceedings of the 6th International Conference on Intelligent Virtual Agents (IVA 2006)*, Springer Verlag Berlin/Heidelberg, Vol. 4133, pp. 169-180.

[17] Garau, M., Slater, M., Pertaub, D.P., and Razzaque, S. The Responses of People to Virtual Humans in an Immersive Virtual Environment. *Presence: Teleoperators and Virtual Environments*, Vol. 14, pp. 104-116.

[18] Jörg, S., Hodgins, J., and O'Sullivan, C. The Perception of Finger Motions. *Proceedings of the 7th Symposium on Applied Perception on Graphics and Visualization (APGV 2010)*, pp. 129-133.