MODELING AND SIMULATION OF AGENT BEHAVIOR IN A GOAL FINDING APPLICATION FOR EVACUATION

A Dissertation Submitted to the

Faculty of Graduate School

BOWIE STATE UNIVERSITY

In Partial Fulfillment

of the Requirements for the Degree of

DOCTOR OF SCIENCE IN COMPUTER SCIENCE

Department of Computer Science

By

Kolawole O. Ogunlana, Jr., M.Sc.

DECEMBER 2015

BOWIE STATE UNIVERSITY

THE GRADUATE SCHOOL

DEPARTMENT OF COMPUTER SCIENCE

DISSERTATION COMMITTEE

____

Sharad Sharma, Ph.D.
Chair

____

Sadanand Srivastava, Ph.D.

____

Jie Yan, Ph.D.

____

Joseph Gomes, D.Sc.

____

Jacqueline Le Moigne, Ph.D.
External Examiner

Candidate: Kolawole O. Ogunlana

Date of Defense: 10/29/2015

ABSTRACT

MODELING AND SIMULATION OF AGENT BEHAVIOR IN A GOAL FINDING APPLICATION FOR EVACUATION

Doctor of Science in Computer Science

Bowie State University, December 2015

Today it is expensive and time consuming for emergency personnel to perform multiple evacuation drills in real time for a building. We cannot gain knowledge to improve the design and layout of future buildings without running multiple drills. The purpose of this study is to investigate agent's behavior during emergency evacuation scenarios in a goal finding application. We implemented a goal finding simulation evacuation application (in C#) to help us run multiple drills and what-if scenarios. The first objective of this study is to investigate **agent's behavior** during emergency evacuation scenarios in a goal finding application. Second objective is to model **learning and adaptive behavior** which includes individual and collective behaviors. The adaptive behavior focuses on the individual agents changing their behavior in the environment. The collective behavior of the agent focuses on the crowd-modeling and emergency behavior in the goal finding application. The last objective of this study is to develop **new intelligent agent based characteristics** such as autonomy, social ability, cooperativeness, learning ability and level of panic which define their final behavior when trying to reach a goal. The contributions of this study are combining of Genetic Algorithm (GA) and Neural Network (NN), using fuzzy logic to model panic behavior for agents to simulate evacuation in a goal finding application. Result of this study is a C# application that is compared and validated to real-time data from an evacuation drill and commercial evacuation simulators like Pathfinder.

ACKNOWLEDGMENTS

I would like to express gratitude to my advisor, Dr. Sharad Sharma, for his extraordinary direction, assistance, and backing for my research dissertation work.

I also want to express appreciation to all my dissertation committee members: Dr. Sadanand Srivastava, Dr. Jie Yan, Dr. Joseph Gomes and Dr. Jacqueline Le Moigne for taking time out of their busy schedules to attend various dissertation meetings and providing constructive feedback during the period of my work.

I also want to acknowledge the Computer Science Department, Doctoral Program Coordinator: Dr. Hoda El-Sayed and Title III Grant. I would like to encompass my heartfelt thanks to Dr. Cosmas U. Nwokeafor for his supervision, direction, provision and wherewithal that he has conferred upon me during this academic journey.

I would like to extend thankfulness to Dr. Langdon for the HGBI fellowship funding to sustain my academic journey. I would like to acknowledge all of my fellow graduate students and evacuation study participants at the Department of Computer Science for their support and advice.

My wife, Dr. Sophia Ogunlana, also deserves special accolades for cheerleading and always providing a listening ear and support.

Finally, I am thankful to my parents: Mr. Kola Ogunlana, Sr., and Mrs. Olu Ogunlana, and sisters: Seun, Tobi and Sola for the many years of emotional and financial support.

DEDICATION

I would like to dedicate this dissertation to my father, Kolawole Ogunlana, Sr., my mother, Oludolapo Ogunlana, my in-laws, Carol and Kennett Johnson, and my dear wife, Dr. Sophia Ogunlana. Without the encouragement and drive instilled in me by my father and mother, I would not have had the grace that it takes to make it through this long journey. My wife provided constant advice, encouragement, help, guidance, and amazing insight, which truly aided me in the completion of this academic voyage. Finally, I would like to dedicate this dissertation to my toddler son, Joseph Bolutife. His curiosity in discovering and learning new things encouraged me in furthering my research for the dissertation.

Table of Contents

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Problem statement

A model or simulation provides us the ability to perform various tasks seen in the real world. For example without models, imagine how expensive and time consuming it would be to simulate an emergency evacuation in a plane or building. The objective of this thesis is to propose a model for simulating agent behavior in a goal finding application using an algorithm based on genetic and neural networks that includes agent behavior. The purpose of the research is to examine the behaviors that people acting as agents' exhibit during emergency evacuation situations. In those situations, the goal is to find the nearest exit. Furthermore, it sought to model learning and adaptive behavior, by focusing on individual agents changing their behavior as they receive external stimulus from the environment, and collective behavior such as crowd-modeling and emergency behavior. In addition new intelligent agent based characteristics such as autonomy, social ability, cooperativeness, learning ability and level of panic were examined as important factors to consider as the agents attempted to reach the exit goal. To simulate evacuation, the needs include paying for access to a real size plane, recruiting sample passengers, pilots, flight attendants and crew members. Sharma (Sharma S. , 2012) explained the renewed use of Agent-Based Modeling and Simulation (ABMS). Intelligent agents can be used by students interested in other applications like teaching kids how to safely cross busy intersections, providing online instructions in a university and giving military and medical personnel valuable experience they may face in performing their professions. Using an accurate simulation model is very important especially in emergency evacuation scenarios (Sharma, Otunba, Ogunlana, &

Tripathy, 2012). The accuracy is even more important in a simulation where results can change as various parameters are changed. For example psychological parameters of anger, stress, and panic, would have to be accurately observed in an emergency evacuation in a world. The idea of finding goals by searching for sub goals while running a simulation is used to provide an algorithm for moving an agent closer to an exit in disaster evacuation multi-agent settings.

This research examines how to simulate evacuation of a building structure observing the perception and cognition of the intelligent agents in a goal finding application. In the beginning of a project for example the designer of a building usually has to plan and strategize how to make the space usable and efficient for the people using the facility. The decisions made by the designer will affect the future behavior of the people that will move through the building or airplane structure to be constructed. The designer would benefit from having an application that would help to connect the way people behave and the factors considered in construction of a building structure. How do we test whether this application precisely simulates the navigation of people when developing a structure? Consequently, an investigational device that can be used to model people's progress through a path and the way they behave in various scenarios they face is projected and to be developed. This device will provide wise methods for re-tooling the pattern and the layout of building meeting places and will direct designers on achieving a higher probability of efficient and maximized use of these structures. A prototype application was developed that uses an existing building layout to demonstrate the simulation. In addition more prototype applications showing the simulation functionalities of the planned evacuation representation to display the way people behave and their thought process for a building evacuation is also shown.

Simulating how intelligent agents can learn from the model environment to find the nearest exit in an emergency evacuation is very important. Currently, there are many goal finding applications that exist (refer to Table I) to simulate agent behaviors, using cost effectiveness with past experiences and fast communication as a model for agent behavior. Nevertheless, these goal finding applications have disadvantages that make it difficult to work with them since the set up time and expense of setting up the simulation can be high and constant change of user role experienced by the agent. In the goal finding application to be developed, we will utilize an easy to set up simulation that is cost effective. Each individual agent will be assigned to a type role that does not change. For example in a simulation with three agents, the first agent could be assigned to type 'calm', second agent assigned to type 'hostile' and third agent assigned to type 'selfish'. The intelligent agent will experience various emotions such as anger, stress and panic as they try to find their way out of a building. This simulation will provide useful training and education. The simulation starts with assigning the intelligent agents a goal that they need to fulfill. Other agents may be given secondary goals that would help them achieve their main goal. Obstacles are also present throughout the building and the animated agents will try to avoid them. To supplement and support the accuracy of the model results from the simulation of the application and the data it generates, a comparison would be done to the data from an actual evacuation from an airplane. For the period of the evacuation, data will be collected to form a dataset from the behavior of the people to serve as input for the simulation application. The results would illustrate the way humans behave in the model proposed in this thesis and provide a dependable and truthful conclusion matching real-time simulation scenarios.

Research Questions and Hypothesis

Research Questions

a. How can intelligent agents learn from their environment in a goal finding application for evacuation simulation?

b. What adaptive behavior and collective behavior are found in goal finding application for evacuation simulation?

c. Which agent-based characteristics affect the speed of finding exits in a goal finding application for evacuation simulation?

Hypotheses

Hypothesis 1: There will be a positive correlation between the faster evacuation time and smaller size of occupants.

Hypothesis 2: There will be a positive correlation between the faster evacuation time and the number of runs.

Hypothesis 3: There will be a positive correlation between faster evacuation time and the type of behavior exhibited by occupants.

Goal and Objectives

Goal

The study of how people behave when evacuating involves a broader method in disaster scenarios in a goal finding application. Many goal finding applications are available that simulate agent behaviors that are cost effective, built on past experiences and configured for fast communication between agents (refer to Table I). The planned goal finding application combines both genetic algorithm and neural networks to explore how agents can learn from the environment and look for exits during an evacuation. Agents are divided into intelligent agents and steering agents. A fitness function is used in the genetic algorithm to evaluate and give the agents the ability to learn while the neural network is used for classification, noise reduction and prediction.

The research work also proposes the modeling of adaptive behavior and collective behavior of agents. The adaptive behavior focuses on the individual agents changing their behavior in the environment and formulating their response by learning from the dynamic factors in the environment. The collective behavior of the agent focuses on the crowd-modeling and emergency behavior in the goal finding application. Agents are likely to cooperate with each other in order to reach a goal. Some of these agents have a secondary goal that leads them to group together as they move towards the goal.

The development of new intelligent agent based characteristics such as autonomy, social ability, cooperativeness, learning ability and level of panic which define their final behavior when trying to reach a goal. The three major characteristics to be defined are reactivity, proactively, and social ability. An intelligent agent will be able to perceive their environment and respond in a timely manner, take initiative to reach their design objectives and interact with other agents to satisfy their collective goals.

Objectives

1. Investigate agent's behavior during emergency evacuation scenarios in a goal finding application.
2. Model learning and adaptive behavior which includes individual and collective behaviors.
    - The adaptive behavior focuses on the individual agents changing their behavior in the environment.
    - The collective behavior of the agent focuses on the crowd-modeling and emergency behavior in the goal finding application.

3. Develop new intelligent agent based characteristics such as autonomy, social ability, cooperativeness, learning ability and level of panic which define their final behavior when trying to reach a goal.

Contributions

1. Combining of Genetic Algorithm (GA) and Neural Network (NN) for learning and adaptive behavior to simulate evacuation in a goal finding application.

2. Using fuzzy logic to model panic behavior for agents to simulate evacuation in a goal finding application.

3. Developing an agent-based evacuation application (C#) that can help plan emergency evacuation scenarios, run numerous event-driven evacuation scenarios, support research in the areas of agent behavior, and model the movement of responders and security personnel.

Benefits of Research

a. Costly full-scale evacuation drills can be reduced by running multiple building evacuation drills on an agent-based evacuation application. The cost of conducting full-scale evacuation drills can go as high as millions of dollars.

b. Time consuming full-scale evacuation drills can be shortened by running multiple building evacuation drills on an agent-based evacuation application. It can take a full day to run a full-scale evacuation drill. Additional set up and planning time can also be needed to ensure that the evacuation drill goes successful.

c. The injuries of participants during full-scale evacuation drills can be avoided by running multiple building evacuation drills on an agent-based evacuation application. Injuries such as broken bones and emotional distress have occurred in past drills especially those that include environmental elements like smoke and fire which may cause panic in participants during a full-scale evacuation drill.

d. Constricted optimistic scope of emergency circumstances mandated by safety regulations for a full-scale evacuation drills, can be broadened on an agent-based evacuation application. Safety regulation discourages using minors and having environmental conditions like fires and smoke in a full-scale evacuation drill which is unrealistic to simulate a real emergency.

e. Successful evacuation of a building relies on more than the movement rates validated in full scale evacuation drills. Elements in the result of a real emergency evacuation of a building include: first responders and emergency staff abilities and training, building integrity and layout of rooms; occupants and obstacle attributes; and real calamity settings like fire and smoke. Agent-based evacuation application can accommodate for these short comings in full scale evacuation drills allowing variable settings of responders, type of buildings, occupants (male/female gender and old/young age) and fire level in an evacuation simulation.

f. Full scale evacuation drills only provide a standard for constant assessment of various building layout and exit configuration. Compliance with regulations do not realistically measure evacuation proficiencies that can be seen in an agent-based evacuation application. Agent-based evacuation application also encourages optimization of evacuation systems because it is prone to subjective decisions made in full scale evacuation drills.

g. Agent-based evacuation application encourages implementation of new technology for increasing how long occupants can survive after an emergency with environmental factors like a fire.

h. Building owners will be able to implement the knowledge gained from the simulation to structure the layout of various rooms to minimize the evacuation time of occupants during real emergencies.

i. Architectural educational institutions will be able to take advantage of the simulation tool to provide training materials for designers of building to test the safety of their blueprints before it is built.

Target Audience

- Researchers interested in visualizing evacuation time and what-if scenarios by incorporating data on emotions and movements by incorporating agent-based evacuation modeling.
- First responders and security personnel who need to help people in evacuating safely from buildings.
- Scientists who want to study event driven evacuation scenarios for decision making strategies.
- Government agencies who want to support research in human behavior and emotional behavior modeling in disasters.
- Regulators that create safety standards for building codes and user manuals for evacuation drills.

- Designers and architects interested in creating buildings of the future that are resistant to the bad effects of emergency scenarios.
- Education institutions that train emergency personnel who need new information for their curriculum and course materials.
- Building owners of new and existing buildings interested in keeping their occupants safe and reducing insurance premiums.
- Building insurance companies and adjusters interested in correctly assessing the risks of damage when emergency situations occur in the buildings they insure with a policy.

Summary of Methodology

The methodology explained further down in Chapter 3 will involve using the goal finding application to create a sampling of agents looking for a goal and how long it took them to find the goal. Then compare it with commercial evacuation simulators and real life data from an actual evacuation drill at Bowie State to validate the authenticity of our simulation. Once validation is complete, examine the goal finding application to test the various research questions mentioned above in section 1.2 of the dissertation.

Limitations of the study

- For safety reasons the evacuation drill that will be carried out at Bowie State Computer Science Building Lab, will not be able to simulate panic behavior real time. This is because we do not want to endanger the lives of volunteers in accordance with the IRB approval for the study.
- No validation of the panic behavior in the simulation with the evacuation drill.

- Evacuation drill did not incorporate children, blind people, and handicap people.
- Students with disabilities were not included due to potential conflict with HIC (Human investigation committee) approval.

Arrangement of dissertation

The dissertation is organized as follows: Chapter 1 focuses on the introduction overview of dissertation by identifying the problem statement, goals and contributions of thesis work. Chapter 2 presents a review of relevant literature dealing with various models of group behavior and the comparison of goal finding application. Chapter 3 presents modeling of favorite goal and sub-goal. It details what has been done so far with results and what will be done next in the study. Also an evaluation of the research will also be provided in this section. Implementation of the goal finding application is detailed in Chapter 4 of the dissertation. The research outcomes, in Chapter 5, shows the demographic and descriptive results, quantitative and qualitative results, device effects, research questions and hypotheses results, and hypotheses results, and statistical conclusions. Chapter 6 contains the conclusions and recommendations, and future work of the current research study.

CHAPTER 2

LITERATURE REVIEW

The learning of agent behavior is thought-provoking and has led to numerous studies by various researchers trying to understand agent behavior. In trying to understand and model agent behaviors, researchers have built various simulations to mimic the numerous emotions experienced by humans, especially when they are attempting to reach a goal. EvacSim, buildingExodus and Simulex are some of the examples of simulators that exist today to model the behaviors of people when evacuating buildings. Poon's approach using EvacSim models the evacuation of high rise buildings with the ability to scale to a large number of people while analyzing their individual behaviors (Poon, 1994). EvacSim users can select the type of behaviors that will be associated with a particular building by choice or by chance using probability. Once the behaviors are selected, simulated people in EvacSim can interact with other people and other stimuli in their environment. The simulation starts with a warden or fire alarm alerting the people in the simulation of the need to evacuate the building. Input data that is received by the people which controls their behavior when evacuating, include low warning leading to people seeking ways to put out the fire or warning others to evacuate. Medium warning adds a protection behavior for people to protect doors. High warnings lead to people seeking exits in the building. Each of the individuals in the simulation is assigned a maximum horizontal surface speed (1.4m/s), maximum speed going down stairs (0.9m/s) and the space occupied by the people (0.3 m2) (Poon, 1994).

The speed of the individual is affected by how large a crowd is moving in the same direction the individual is trying to evacuate in the building (density). An individual is able to travel at the maximum speed as long as the density is less than a particular minimum (Dmin). The speed decreases to zero when it reaches a maximum density (Dmax). People are placed in various spaces in the building which they may or may not be very familiar with and this affects the exits they are able to find when evacuating. The more familiar an individual is with a building layout, the more choices they can select from the exits and also communicate this information to other people. Other factors that determine which exit an individual selects are how crowded the exit is, how far the exit is from them, and whether obstacles are blocking the exit. The evacuating software similar to EvacSim is buildingEXODUS. It is capable of simulating people looking for exits. The buildingEXODUS software focuses on the interfacing of people, how people are organized, and the way people are located in their world (Galea, 2013). It can scale up to thousands of people in a large area and provides the option to add environment elements like smoke, heat and toxic gases which inhibit them from evacuating successfully (Galea, 2013). It has the ability to be visualized in a 2-D or 3-D space allowing the simulator modeler to interact with the people as they try to find their way out of the building. The simulation is built on five interacting elements: Occupant, Movement, Behavior, Toxicity, and Hazard (Galea, 2013). These sub models all interact in a grid-like world with a tracking of simulation time with a clock. The grid is entered in by the user through DXF files created in AutoCAD software. People can be added to the simulation using menu items or through selected input files with the ability to specify the gender, age, and physical abilities of the participants.

The participants also have the flexibility to display individual or group behaviors such as going for the nearest familiar exit or taking an elevator. This behavior variable allows a user to get different results anytime they run a simulation. The results are more noticeable when coupled with environmental elements like smoke and fire. When these elements are introduced to the environment, the participant's speed slows and they may not be able to complete their evacuation. Simulex is software that is able to provide the path and distance of a participant when evacuating a building (Simulex User Guide 6.0, 2012). Similar to buildingEXODUS, Simulex provides a 3-D world entered in through a virtual environment generated by AutoCAD. The main distinction that Simulex has over the other simulation software is that it allows the user to set the location of the exits outside the building. Participants are placed in the building individually or in groups with a possible path to follow before the simulation is allowed to run. When the simulation is completed, a saved version can be re-run again at a later date to provide more analysis to the researcher. The researcher can compare and analyze the performance of the algorithm controlling the participants in the simulation based on real life evacuation data. The participant characteristics can also be added such as physical types like office staff, commuters, shoppers, school population, gender (male or female), age group, speed, and country of simulation. Unlike buildingEXODUS, each participant is randomly assigned a walking speed between 0.8 and 1.7 m/s which may be slower if the area is crowded with other participants or they are walking up (0.35 times) or down (0.5 times) the stairs (Simulex User Guide 6.0, 2012). The participant is represented by three circles, a body with radius R (t), two shoulders with R(s) radius and R (b) which is the radius when a participant touches another. The radius are all different for various body types like male, female, child and old who are also affected by psychological and response time entered into the simulator. You can also add a floor; add a staircase to the building and a link between various floors and staircases in the model of the environment.

The environment would also contain distance maps which are the possible paths the participants can use to evacuate a building. Also when goals are assigned to participants, they can change their path when blocked by a large number of people on the way to their exit. By this way participants in simulators like EvacSim, buildingEXODUS and Simulex display autonomous ability which is the key to providing life like behaviors in modeled environment (Dzima, 2001). The life like behaviors is very important in path-finding especially using better understanding plots, determining the position of one point in space relative to another for arrangements, and moving actions for vehicles (Verth, Brueggemann, Owen, & McMurry, 2000). Path-finding is also important when participants in a simulation have to follow the direction of a leader on the way to an exit (Mamdouh, Kaboudan, & F.Imam, 2012). On this journey the participant can construct the potential path to the exit by assigning low weights to locations in the environment that it should go and higher weights to locations it should avoid in the world (Baert, 2000). This provides an incentive for participants to find the exits and evacuate the building faster.

Modeling and Simulation

Historically there have been many types of model simulation developed as a tool to analyze difficult realistic problems like evacuating a building. These models and simulators aid the researchers in testing different conditions multiple times that would either be too expensive or unsafe to do so in the real world (Still G. K., 2007). The results gleaned from the simulations could act as a guide when designing exits in a building but should not be taken as a fact or out of context in making a conclusion. To properly use the results of a simulation, it is important to know that it can only prove/disprove some theory or to shed some light on a possible issue (Still

G. K., 2007). The issue mostly focused on is reducing the evacuation time of participants in a building while realistically modeling the behaviors of people in such situations. Procedures used in modeling and simulation are either a top down approach or a bottom up approach (Still G. K., 2007). The top down approach uses facility rules, broad directions, length measurements and exit widths to show obedience to building codes (Still G. K., 2007). On the other hand, the bottom up approach utilizes systems agent's representatives that are able to make decisions in a reproduction of the compound world (Still G. K., 2007). It is recommended that the top down approach be used first to get a broad understanding of the environment and then use the bottom up approach for further analysis of the evacuation problem. In either of these approaches, researchers would have to take into consideration the response time of people to start evacuating and the actual time it takes to exit the building. The response time of the people is directly related to how fast they are alerted to evacuate the building and should be included in an accurate simulation (Still G. K., 2007). Such an accurate simulation should not be complex to build, change, read up on, and explain to others. It should use the most affordable up to date technology to build its models (Still G. K., 2007). These model simulations include Monte Carlo (sampling) methods, discrete-event paradigm, object-oriented and web-based simulation paradigms (Abu-Taieh & Sheikh, 2010).

The probabilistic fire simulator is an example of an application that implements the Monte Carlo method in a fire evacuation emergency (Hostikka, Korhonen, & Keski-Rahkonen, 2005). The method can show a way to predict how long it would take for various components to stop working in a fire including the time it takes to discover the emergency. During a discrete-event like an emergency, it is important to take into account how people will behave when designing a simulation. One such simulator that takes behaviors into account is the multi-agent simulation system for egress (MASSEgress) (Pan, Han, Law, & Latombe, 2006). It consists of an environment layout, participant's producers, a large database to store simulation information, playback device, graphical user interface and an individual perception, behavior and movement controller (Pan, Han, Law, & Latombe, 2006). The controller helps to provide social behaviors such as bi-directional flow, competitive motion, and queuing driven rules which affects individual movement such as steering, obstacle avoidance and exiting a building. Models of emotions which may be driving the movement of the agent during evacuation include Disposition, Emotion, Trigger, Tendency (DETT), Belief, Desire and Intention (BDI) and Openness, Conscientiousness, Extraversion, Agreeableness and Neuroticism (OCEAN) (Minh, et al., 2010). These models are utilized in simulations with emotions like fear which can be increased or decreased and can be passed on to other agents to change their behaviors and make them more realistic. The fire response performance model (FRP-model) that can be used to realistically simulate fear consists of human features (individual, social, and situation), building features (engineering, situation) and fire features (perception, fire, smoke, heat) (Kobes, Oberije, Post, & Weges, 2007). These human features are important to the social force model that makes it possible to predict how simple behaviors can be used in simulations (Helbing & Molnár, 1995). Another example of object-oriented paradigms like agent-based modeling (ABM) was developed to alleviate the difficulty in modeling agent behavior in simulations.

The key aspect of agent based modeling lies in understanding the mechanism by which autonomous agents interact among themselves and how to validate the accuracy of the simulation in the scenario studied. The study of ABM was drawn from researchers who were looking to find out the definition of evolving and composite actions seen in nonlinear systems (Abu-Taieh & Sheikh, 2010). Researchers of ABM included Adam Smith, Donald Hebb, and

Darwin; all focused on various theories such as Invisible Hand in Economics, Cell Assembly and Evolution respectively. All of these theories were founded on basic single agents communicating with each other to combine and form a novel compound event (Abu-Taieh & Sheikh, 2010). The process of forming the compound event was made easier with the invention of computers to study natural systems by Von Neumann. Neumann contributed to the study on DNA and creation of genetic algorithm searching by computers. The approaches to agent modeling frameworks include:

1. Geometrical Approach: This approach attempts to solve the difficult task of an agent avoiding collision with obstacles having different location and speed in an environment (Sharma S. , 2006). The coordinate system utilized in this approach is an X and Y space measured against time of movement of the agent. Agents avoid obstacles in their path by plotting a way around them either by passing before or after the obstacle in a particular position to reach their goal (Sharma S. , 2006).

2. Cellular Automata (CA): Ulam and Neumann further developed the study of natural systems by creating a Cellular Automata (CA) methodology. This methodology was based on an ordinary corresponding system with individual cells having the ability to make independent decisions on their own at the same time. Also localized behavior by each cell was combined to form global behaviors (Abu-Taieh & Sheikh, 2010).

3. Visibility Graph: This approach focuses on the degree to which a location in a three-dimensional system can be seen from another place in the environment (Sharma S. , 2006). When locations cannot be seen from another place, the length of a network of positions is computed to find out the number of intersecting positions that are needed for the location. The analysis of visibility graph allows for a network of positions evenly spread out in an environment to be taken as input and used to produce a visibility graph (Sharma S. , 2006). Important attributes of visibility graph can be used to discover the way people take in public observable and reachable areas.

4. Path-velocity decomposition: This approach inherits the attributes of visibility graph and builds onto it the ability to anticipate dynamic obstacles (Sharma S. , 2006). Visibility graph is used to find the direct way to a goal by identifying obstacles that are not moving in the environment. Collision with dynamic obstacles are avoided by creating a movement outline using linear time increase, speed steadiness and a range of maximum speed (Sharma S. , 2006). This approaches focuses on calculating the path of an agent using power, rotation, quickening, speed and inactivity.

5. Density correlation: This approach utilizes velocity and movement of agents based on the mass of the environment (Kuligowski E. , 2004/6). Fruin, J. J., Pauls, J. and Predtechenskii, V. M. & Milinskii are some of the researchers that have performed occupant movement studies related to the mass of an environment.

6. User's choice: This approach focuses on the user allocating the velocity, direction and mass values to particular areas of the environment (Kuligowski E. , 2004/6).

7. Inter-person distance: This approach focuses on providing each agent in the environment a 360-degree field of view that restricts other agents, obstacles from coming close and stopping their progress in the path (Kuligowski E. , 2004/6).

8. Potential: This approach assigns each grid cell in the environment a particular numerical value (Kuligowski E. , 2004/6). This value is calculated relative to an existing position in the environment that would enable the agent to navigate in a certain direction. The agent makes use of a diagram with the plan to decrease their potential value from moving one grid to another in the environment (Kuligowski E. , 2004/6). Other attributes that influence the agent's movement in the environment include demeanor of the agent, importance of an exit and agent past knowledge of the environment.

9. Emptiness of next grid cell: This approach focuses on the rule that an agent would not move to a surrounding grid until it is empty of the agent occupying it in the environment (Kuligowski E. , 2004/6). The model would have to decide which agent moves into an empty grid if there is more than one agent waiting to move into the same location in the environment.

10. Conditional: This approach relies on circumstances, configurations, agents and other environment factors like smoke to decide the movement of the agent (Kuligowski E. , 2004/6). The approach does not take into account the overcrowding of the environment.

11. Functional Analogy: This approach uses speed equation designated by subject areas like fluid movement or pull in an environment (Kuligowski E. , 2004/6). The equations may also rely on the mass of the area in the environment.

12. Other model link: This approach inherits the functionality of a different model that is connected to the evacuation model used in the environment (Kuligowski E. , 2004/6).

13. Acquiring knowledge: This approach is based on the prior knowledge gleaned by the agent in the environment (Kuligowski E. , 2004/6). There is an absence of movement algorithm because this approach ignores evacuation time and focuses on areas of overcrowding and queuing in the environment.

14. Unimpeded flow: This approach focuses on areas in the environment where the agent cannot make progress in the path of their goal (Kuligowski E. , 2004/6). Delays and improvement time are adjusted from the evacuation time calculated in the environment.

The behaviors were further studied by Wiener who developed cybernetics the knowledge of managing and messaging in the creature and the device (Abu-Taieh & Sheikh, 2010). This knowledge led to the discovery of Complex Adaptive Systems (CAS) of Aggregation, Nonlinearity and Diversity. Aggregation was made up of tagging agents to have the ability to identify each other and building blocks made of subgroups formed from multipart systems (Abu-Taieh & Sheikh, 2010). Nonlinearity based on the division of labor by individual parts produced greater yield than individual parts doing it alone. Diversity was achieved when each agent focused on its own target and behavior which could be adjusted, adapted, and modified based on the situation. In other words, there is no central command and control structure that would govern the behavior of these agents or system operation. There is a link between an actual system, a theory/model, and a simulation as shown in Figure 1 (Abu-Taieh & Sheikh, 2010). The simulation assumes the responsibility of producer, intermediary, or forecaster as the researcher continues to learn more about the system. This process of learning is seen in neural simulations on machines, computer vision and robotics. More powerful computers have made it possible to create simulations that mirror the way the human brain works with spiking neurons (Abu-Taieh & Sheikh, 2010). Spiking neurons corresponds to the upward or downward propagation of signals from one neuron to the next neuron which corresponds to how to model human cognitive personalities such as thoughtfulness and feelings by an agent (Abu-Taieh & Sheikh, 2010). SpikeStream is a type of hybrid simulator which is built to provide the ability to generate genetic algorithms that can provide neural networks adept at performing particular tasks (Abu-Taieh & Sheikh, 2010).

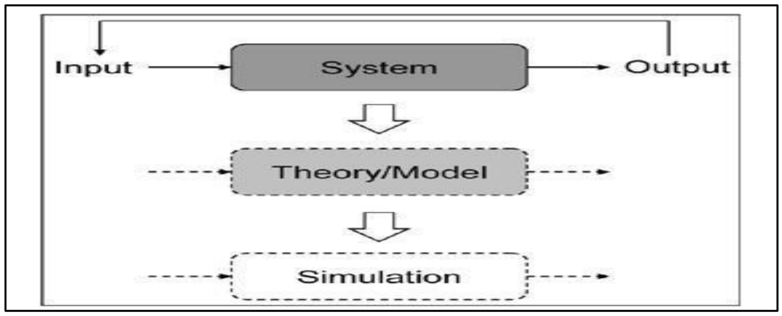


Figure 1: Diagram of link between a system, theory/model and simulation

There are two ways to build simulators using a theoretical and conceptual framework (Barjis, Rychkova, & Yilmaz, 2011). Both frameworks focus on creating vigorous simulation models as shown in Figure 2. This is accomplished by the strength, elasticity, flexibility, and changeability of the proposed models. The strength of the model is encountered when a unique event occurs in the simulation and the experiment can continue without stopping from reaching its goal. On the way to reaching its goal, elasticity refers to how fast the simulation can recover from unique events that may stop progress. Flexibility is reflected in the strength of the model and changeability allows new goals to be created when old ones are not reachable.

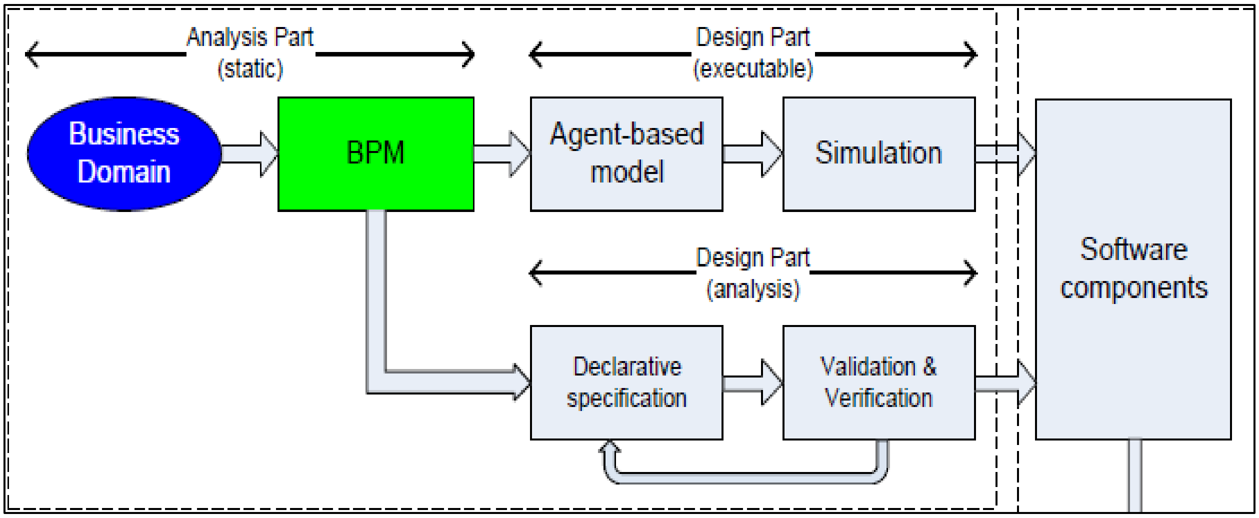


Figure 2: Diagram of framework for software application

Agents come from different backgrounds with the flexibility to adapt their behavior under different environment and formulate their response by learning from the environment. They are often independent to make decisions and are considered diverse, heterogeneous and dynamic. Every agent has its own set of behavior rules and protocols for interacting with other agents. They are capable of learning from the environment and adjust their behavior based on their experience. Collectively, agents are capable of exhibiting intelligence and are classified into two types steering agent and intelligent agent. Steering agents use the action selection and the steering behavior. On the other hand, intelligent agents are autonomous and acquire decision making ability from experience.

Agent

An agent is something that has the ability to perform various tasks (Russell & Norvig, 2003). Computer agents in simulations have defined parameters such as an independent regulator, ability to receive stimulus from their world, having memory over time and variable actions especially when changing goals (Russell & Norvig, 2003). It is assumed that an agent who is rational would take actions to move it closer to achieving its goals. The attributes of rational agents include ability to grade actions that would meet its goals, to remember the existing world, to know the actions that it can take and to match those actions to the stimulus received from the world (Russell & Norvig, 2003). The part of the world the agent cannot see presently is anticipated and tracked by a model-based reflex and stored in its memory (Russell & Norvig, 2003). This model-based reflex distinguishes the agent from human behavior which is able to adapt to a particular environment using its velocity and radius size through a complex changing process that is not perfect (Russell & Norvig, 2003). Each agent has its position and its orientation used to calculate a target position and its orientation (Ian Millington, 2009).The simulation can calculate how quick the agent orientation is moving by using its radius per second that it is change as shown below in figure below (Ian Millington, 2009). The agent in simulation has a horizontal radius of 6 pixels and vertical radius of 14 pixels as shown below in Figure 3. Agent is depicted as two dimensional coordinate point called X and Y affected by gravity holding them to the ground and restricting their movement. The direction of the agent has one positioning value that is an angle from a location alignment (Ian Millington, 2009). This alignment is in anticlockwise angle in radians from the positive Y-axis. The movement of the agent is tracked using the velocity relying on the change in the direction and location alignment values. The position can be calculated by multiplying the velocity and time with the addition of the steering direction. The location alignment can be calculated using the multiplication of the rotation and time with the addition of the steering angle. Velocity is the multiplication of the linear steering and time (Ian Millington, 2009).

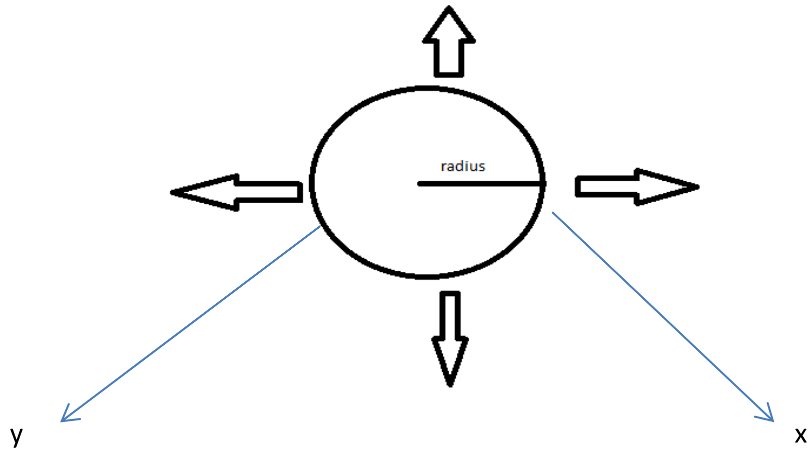


Figure 3 Diagram of agent radiuses in simulation with two dimensional coordinate point X & Y

The different types of agents broken down by attributes include:

1. Human agent: Have attributes like eyes, ears, and other abilities that enables it to receive stimulus from its environment (Russell & Norvig, 2003). This agent uses its hands, legs, mouth and other parts of its body to carry out actions in its environment.

2. Robotic agent: has attributes like cameras and ultraviolet range detectors to receive stimulus from its environment (Russell & Norvig, 2003). This robotic agent uses its various engines to carry out actions in its environment.

3. Software agent: Have attributes like keyboard, graphical user interface, and network packets to receive stimulus from its environment (Russell & Norvig, 2003). This software agent (software robots or softbots) uses monitor screen, output files, and sending packets through the network to carry out actions in its environment.

4. Rational agent: performs the action that would enable it to reach its goal all the time without making a mistake in the environment (Russell & Norvig, 2003). The ability of the rational agent is based on its knowledge of the past experiences saved in memory and input or output effect in the environment.

5. Simple reflex agent: performs the actions based on the present sensory input received from its environment without any recollection from the past (Russell & Norvig, 2003). For example a vacuum cleaner that decides on its next action based on its present sensory input of clean or dirty without checking if it has already visited the location previously in the environment.

6. Model-based reflex agent: performs the actions while accommodating for future sensor input it may receive in the environment (Russell & Norvig, 2003). This ability to anticipate future changes is achieved by keeping a log of inputs received in the past and consulting them when it needs to make a present decision in the environment.

7. Goal-based agent: performs the actions that would lead it closer to reaching saved goals (Russell & Norvig, 2003). The ability of the goal-based agent is achieved by perceiving a reward for some actions that encourages it to follow those actions in achieving its goals.

8. Utility-based agent: performs the actions that are faster, harmless, more dependable or inexpensive that would lead it closer to reaching saved goals (Russell & Norvig, 2003). A utility function provides a tangible real number to attach to various actions that the agent can take in the environment.

9. Learning agent: are able to observe the environment and learn new actions that make it better to reaching its goals (Russell & Norvig, 2003). The learning agent is made up of a faultfinder, knowledge component, and problem initiator and performance component.

10. Hostile agent: are agents that act in ways that stop other agents from reaching their goals.

11. Non Hostile agent: are agents that act in ways that encourage other agents in reaching their goals.

Behavior

A. Steering Behavior:

Agents exhibiting steering behavior are known as steering agents. In a three dimensional space based framework, an agent moves from one state to another for achieve its goals and progresses towards its target that is governed by rules of behavior. While doing so the agent takes into account the presence of other such agents and therefore is "steered" to move in a direction by factoring in the behavior, movement, speed and location of its neighbors. In a space based framework, an agent has three most important attributes namely its current location, velocity and acceleration. The acceleration of the agent signifies the sum total of all external forces acting on the agent that helps in determining the new position, speed, and orientation of the agent. The steering behavior is caused by an external force and causes the agent to move in a trajectory. The agent displays both scalar and vector properties. The scalar properties associated with the agent would be its mass, speed, and rotation. Whereas, the vector properties associated with the agent would be its position, velocity, heading (or orientation), and maximum force.

According to Craig Reynolds (Reynolds, 2000), there are three most important and distinct type of steering behavior namely alignment, cohesion and separation. If an agent moves toward the average position of its local neighboring agents then such a behavior is termed as cohesion. If the agent is steered toward the average heading of its local neighboring agent then such a behavior is termed as alignment. If the agent is steered away from its local neighboring agents in order to avoid flocking or crowding in space then such a behavior is termed as separation. Steering behavior is represented as force vector. Other common steering behaviors are seek, flee, arrive, wander, and path following. If the behavior result in a force that steers the agent with maximum speed towards a target position then such a behavior is termed as seek. In this case the target remains static. On the other hand, if the behavior result in force that steers the agent in full speed away from a target position then such a behavior is termed as flee. Here the target could be either static or dynamic and the fleeing behavior ensures that the agent would steer away from the static or the predicted location of target. Path following is a type of behavior that steers the agent along a particular predetermined path. This is an important aspect of steering behavior. In this case the agents have to move through a series of checkpoints. It is similar to real life where people have to go through predefined paths to reach the goal during an emergency. Wander behavior steers the agent randomly with a random steering force. There is no target to seek and the behavior results in a random walk. All these behavior are improvisational behavior of an autonomous agent.

B. Intelligent Behavior:

Intelligent agents are autonomous agents. They adapt to learn and improve using the knowledge of the environment. By doing so, they acquire decision making ability from experience. The three major characteristics that define an intelligent agent are:

• Reactivity: It is the ability of the agent to perceive an environment and respond to it in a timely manner.

• Pro-activeness: It is the ability of the agent to take initiatives in order to achieve design objectives.

• Social ability: It is the ability of the agent to interact with other agents in order to satisfy design objectives. The agent calculates the path to move towards a goal by calculating its distance, direction, and steering force by avoiding obstacles in between. The calculated distance, direction and steer force is feed into the neural network. The agent checks if the goal is reached and stops.

Modeling Agent behavior

Modeling agent behavior by steering agents and intelligent agents can be done with computers. Sun (Sun, 2009) explained that it involved static and dynamic methods of cognition written in computer programs that produce models that can be run by researchers. Many of these models were discovered in the field of artificial intelligence but did not have the full backing of validation with real live data (Sun, 2009). Psychologists were able to provide models such as Anderson's Ham which were validated with real live data. The popularity of neural networks in the 80s brought about the discovery of other computer models that were built on easy to understand parallel algorithms validated with real live data. This later led to the discovery of hybrid models that utilized both neural and artificial intelligence methods which provided a connection between computer models and agent behavior. These models include behavior outcome models (identical behaviors as humans), qualitative models and quantitative models. The identical behaviors of humans usually occur in dense crowded areas which may make it difficult to simulate in real time without pre-planning the movement of agents (Loscos, Marchal, & Meyer, 2001). Artificial intelligence allows for the simulation of dense crowded areas but may need a large amount of computer processing power to be realistic. An alternative way to model areas with large number of people that have set goals involves a two dimensional point to point layout of the environment (Loscos, Marchal, & Meyer, 2001). Graphs are used to locate the goals and to move the humans towards them in the environment. The rendering of the environment in the graphical interface can also be challenging with a large number of people but it is important so the researcher can accurately observe the behaviors in the simulation (Tecchia, Loscos, Conroy, & Chrysanthou, 2001). These behaviors usually begin as group behaviors as the agents interact with each other (Musse & Thalmann, 1997). This interaction can be made easier if the agent can decide before hand what obstacles in the environment they plan to avoid (Feurte, 2000).

The dynamics that plays out in the interaction of crowd agents when trying to avoid obstacles is that everyone tries their best to evacuate in an orderly manner to avoid panic (Tran, 2013). This gives credence to the use of physics principles like force and statistics in modeling agent behavior. Modeling individual behavior as a way to understand the dynamics of crowd behavior is encouraged through the creation of microscopic models (Henein, 2008). It allows the researchers to study the quality of behaviors witnessed in a crowd and the effects of individuals during simulation. The common emotions experienced by humans during simulations when attempting to reach a goal include happy, sad, angry, stressed and calm emotions. Anger and stress are some of the emotions that significantly cause behaviors such as panic in humans. It can also disrupt the cognition process when trying to reach a goal such as evacuating an airplane. The cognition process unfortunately is very complex and cannot be easily explained because of the large number of variables that are at play when humans need to make a decision. Computer scientists and researchers working in fields like support systems, adaptive systems, intelligent systems and authoring tools have suggested numerous ways of categorizing emotions (Aleven, McLaren, & Sewall, 2009). A large number of these categorizations are based on the agent involvement with the implementation of the genetic algorithm and vehicle routing algorithm in their environment (Aleven, McLaren, & Sewall, 2009). This has led to an increasing emphasis and focus on genetic and vehicle routing algorithms. The emphasis on these algorithms has attempted to show the relationship of emotions to behaviors of the various agents. Also knowledge is gained from observing agents in a simulation in a goal finding application for evacuation.

Multi-agent systems have been extensively used in numerous regions of study such as robotics (Sharma, Singh, & Prakash, 2008), individual performance (Sharma, Singh, & Prakash, 2008) (Sharma & Singh, 2006) (Sharma S. , 2009) (Sharma, Singh, & Gerhart, 2007), agent based modeling (Wooldridge, 1999), trouble solving (Wooldridge & Jennings, 1995), and kinetic plan blueprint (Sharma S. , 2010). Modeling individual actions throughout evacuation in a disaster setting is a difficult assignment. Sharma (Wooldridge, 1999) (Wooldridge & Jennings, 1995) (Sharma S. , 2010) has developed a multi-agent system AvatarSim (Sharma S. , 2010) (Sharma S. , 2009) (Sharma & Gifford, 2005) (Sharma S. , 2010) for simulating individual activities by means of fuzzy logic for disaster scenarios such as evacuation.

Agent-based Modeling and Simulation (ABMS) has directed the development of multi-agents systems for evacuation. (Sharma S. , 2010). ABMS addresses difficulties in important sections such as social sciences, natural science, psychology, and supply chains. An example of the negative impact of evacuation without a plan is seen in the mass rush of people inspired by terror which leads to causalities as people get compressed or crushed over. These phenomena are seen in life frightening situations such as an inferno in crowded buildings or in spaces where people dash towards the exit. Herding actions leads to dangerous, overcapacity, slower flight actions, plus increase in fatalities. Simulation outcome (D. Helbing, 2000) (Helbing, Farkas, Molnar, & Vicsek, 2002) (Still, 2005) propose ways of minimizing the unexpected conditions in disaster scenarios in alarm and the increasing a most favorable getaway plan for evacuation. Helbing et al. (D. Helbing, 2000) have replicated pedestrian association according to pull and repulsion. The pedestrian respond to obstacles and other pedestrians according to forces of attraction seen in the simulation. Their study shows filing formation and terror actions (Helbing, Farkas, Molnar, & Vicsek, 2002). Computer scientists in support systems, adaptive systems, intelligent systems and authoring tools have expressed a keen interest in genetic algorithm and vehicle routing algorithm and modeling it through agent behaviors and emotions. Vincent Aleven (Aleven, McLaren, & Sewall, 2009) proposed a way to break down the process of creating an intelligent system capable of teaching and training students in reaching a goal solution of various problems. This was done through the use of authoring tools that are capable of creating intelligent systems that would be useful as a tool to diverse students.

Sharma (Sharma S. , 2012) explained the renewed use of Agent-Based Modeling and Simulation (ABMS). Animated agents can be used by students interested in other applications like teaching kids how to safely cross busy intersections, providing online instructions in a university, and giving military and medical personnel valuable experience they may face in performing their professions. Using an accurate simulation model is very important, especially in emergency evacuation scenarios (Sharma, Otunba, Ogunlana, & Tripathy, 2012). The accuracy is imperative in a world where results can change as various parameters are modified.

For example psychological parameters of anger, stress and panic, would have to be accurately observed in an emergency evacuation in a specific environment. Simulating how animated agents use their environment to find the nearest exit during an emergency evacuation is very critical. The animated agent would experience various emotions such as anger, stress and panic as they attempt to exit the building. Agents would also react to environmental factors; smoke for example, exacerbates the animated agent's speed and direction. The agents have characteristics such as:

1) Attributes: An agent is a discrete individual and has mass, position, velocity, force, and speed.

2) Emotions: An agent has emotions such as level of panic and stress attributes.

3) Memory: The agents are goal oriented. The agents have a list of goals that constantly keeps increasing or decreasing depending upon its interaction with the environment.

4) Rules of behavior: An agent has the ability to learn and adapt its behaviors based on experience in the environment using a genetic algorithm and neural network.

5) Decision making capability: An agent is autonomous and can function independently in its environment while interacting with other agents (dynamic obstacles) and environment (static obstacles.

Modeling Evacuation Agent behavior

The ability to model agent behavior in the decision making ability of agents when they are evacuating is very important, especially when creating an environment that represents the real world. Three ways that were used to represent the real world by other researchers include coarse networks (Pedroute), fine network (buildingEXODUS), and continuous (Simulex) (Chooramun, Lawrence, & Gale, 2010). Each of these methods of representing the real world has their advantages and disadvantages that led to their combination in the hybrid spatial discretization (HSD) method (Chooramun, Lawrence, & Gale, 2010). The coarse networks increase the pace of the agents in the vague or less detailed areas that are needed in the modeled environment. Fine networks pickups up the load for rendering areas in the environment where there is a lot of interactions between agents. Lastly, the continuous is used where more detail is needed to be captured in the agent to agent interactions in the environment. Once a method such as HSD has been selected for modeling the physical environment, the next step would be the select the way the results of the evacuation would be analyzed by the researcher. The three ways of analyzing evacuation are through simulations, optimization and risk assessment (S. Gwynne, Lawrence, & Filippidis, 1999). Simulations, on one hand, focus on modeling the point to point movements of the agents and their behaviors as they search for exits in the evacuation. Unlike simulations, optimization is built on the premise that agents would move as a group to evacuate together in one direction without attempting to notice other obstacles and parts of the environment (S. Gwynne, Lawrence, & Filippidis, 1999). Risk assessments take a different approach with agents affected by environmental factors like fires, smoke and reduced visibility of risk of not finding the exit. The behavior of the agents in each of the analyses may be affected by artificial intelligence, rules, implicit, and functionality (S. Gwynne, Lawrence, & Filippidis, 1999).

Artificial intelligence tries to replicate human intelligence in the evacuation (Kuligowski E. , 2005). Rules use “if-then” conditions of the environment faced by the agent to control their reactions. Implicit matches' identifiers are assigned to agents to control how fast or slow their movement is while evacuating (Kuligowski E. , 2005). Lastly, functional uses physics equations, for example, to control the movement of the agents. The movement of agents can be taken into account when designing buildings. The safety of a building design can be measured against the number of fires in similar designs against the number of people that were hurt in such an incident (Korhonen, Hostikka, & Keski-Rahkonen, 2005). The data when plotted on a graph is referred to as the F-N plot. The F-N plot usually captures the three kinds of evacuation crowd behavior which are individual, individual to individual and groups (Winter, 2012). Individual behavior is based on following past knowledge of where exits are located when evacuating. It also involves other individual following the instructions of a leader who they perceive has knowledge of exits during evacuation (Winter, 2012). The probability of individual to individual interaction through pushing or a stampede increases with panic. The negative effect of panic also increases as the size of the group that is trying to evacuate grows in a building (Winter, 2012). The perception of panic starts the behavior process of individuals when deciding what situations should cause them to evacuate and what steps to take to evacuate and then to actually evacuate (Kuligowski E. D., 2009). The time it takes to begin the evacuation is important to track and involves predetermined movement time, taking a walk calculating speed, passenger attributes, taking steps and choosing the exit (Shi, et al., 2009). Exit choice decisions are usually based on the particular locations that individuals have knowledge of before the emergency (Benthorn & Frantzich, 1996). Sometimes the exit chosen by the individual trying to evacuate may not be the closest but another better one may be picked if the individual can see beyond the exit (Benthorn & Frantzich, 1996). Also verbal evacuation directions are better than alarm rings when trying to get people to start moving towards an exit. Other psychological factors that affect an individual's ability to find an exit after receiving verbal warning are worrying, overburdened with too much task, task difficulty, significant task, tiredness and worldly factors (Pires, 2005). These factors that affect the speed of individuals as they move from one area to another towards an exit can be combined into a space compressed object (SCO) (Shen & Chien, 2005). The SCO would involve utilizing powerful computers to track the various factors in the simulation (Quinn, Metoyer, & Hunter-Zaworski, 2003).

Modeling Evacuation Learning and Adaptive Agent behavior

Learning and adaptive agent behavior in modeling evacuation is important in improving the speed at which individuals are able to find an exit out of a building. One such system is the reinforcement learning implemented using different pathways for transmitting behaviors that interacts with a rules engine (Pyeatt & Howe, 1998). This system is seen in action in the robot automobile racing simulator (RARS) consisting of multiple agents reacting to sensory inputs in their environment while attempting to achieve a task like finding an exit (Pyeatt & Howe, 1998). The agent is able to finish its tasks such as finding an exit by learning to change direction, increase their speed and navigate around other agents by using a reinforcement learning neural networks. This network allows the agent to learn a new behavior, get good at using that behavior and then move on to learning the next behavior in its list. Ultimately the agent would learn how to interact with other agents in their environment (Pyeatt & Howe, 1998). Though RARS is simple, reinforcement learning can be applied to getting solutions faster for more involved problems. This is achieved by two main implementations of reinforcement learning: adaptive heuristic critic (AHC) learning to Sutton and Q-learning to Watkins (Lin, 1992). Both implementation of reinforcement learning have three variations each that shorten the learning cycle by repeating experiences, observing steps in models for strategy, and learning (Lin, 1992).

Both variants utilize the process of breaking a large problem into smaller tasks that can be easier to solve and then combined to solve the overall problem. This process is called a behavior based system and was utilized in a robot named OBELIX given positive reinforcement for every action that draws them closer to finding the solution to a problem (Mahadevan & Connell, 1992).

When trying to find a solution to a problem like looking for an exit, it is important to understand the common behaviors that are inherent in every human in an evacuation. These behaviors include the preference that individuals lean towards moving in the left direction especially when they have to change directions in taking a shortcut (Park, Kim, Whang, Parl, & Lee, 2007). The direction of the individual can be modeled with a magnetic force where the wait time of people in queues formed when evacuating can be investigated and analyzed in the simulation (Okazaki & Matsushita, 1993). The analyses done in a simulation can be investigated using a two connectionist network made up of an action and evaluation network (Anderson, 1987). The action network creates the system behavior which is the decision point used to match an action to a given state (Anderson, 1987). Evaluation network provides a fitness function to evaluate the various states (Anderson, 1987). The various states can be achieved through the reactive knowledge gained from sensors in the environment of autonomous robot architecture (AuRA) (Arkin, Integrating Behavioral, Perceptual, and World Knowledge in Reactive Navigation, 1990). This is achieved through breaking down larger tasks into smaller more manageable sections. The attributes of AuRA include behaviors as the foundation (obstacle avoidance), reaction to sensory input from their environment, mimicking animal like behaviors, and practical application leading to measurably results (Arkin, Reactive Robotic Systems, 1995). Furthermore adding artificial intelligence to AuRA provides a hybrid system that gives the best of both world of reacting quickly to stimuli from the environment when navigating the agent

(Arkin, 1995). Navigating the agent who may be mimicking animal like behaviors produces applications like “Qualitative human pre-testing of simulated environments and tasks”, “Quantitative comparison of human- and animal-generated trajectories”, “Human-generated motion trajectories as training exemplars for evolution/learning”, and “Extracting cues for animate motion perception through iterative experiments” (Blythe, Miller, & Todd, 1996). These various applications can all be grouped under the term of synthetic learning providing “pattern classification, prediction, and the adaptive control of dynamical system” (Barto, Sutton, & Watkins, 1989).

An example of synthetic learning is ALECSYS which is a combination of learning through classification and advanced genetic algorithm (Dorigo & Colombetti, 1993). This is achieved by having both a simulated and real life agents (situated agents) learning together through studying the goal to be achieved and being supervised as they make progress (Dorigo & Colombetti, 1993). These situated agents are usually placed in “adaptive intelligent systems (AISs)” which is a balance between the controlled environment in AI systems and the non-bounded capabilities of humans (Hayes-Roth, 1995). This compromise allows the agent to adapt to the myriad of situations that it would face and allow it to still accomplish its tasks. Two additional methods that aid in developing an agent that can effectively accomplish its tasks include finding a way to consolidate the count of inputs needed for the agents to function and also merging multiple similar actions into one allowing for a simplified architecture for learning (Long-Ji, 1990). This architecture can be simple as a trial and error process of learning new behaviors by an agent without human intervention with a “performance feedback function as reinforcement (Mahadevan & Connell, 1991). The new behaviors can be improved by fine-tuning neural network inputs (Cazenille, Bredeche, Hamann, & Stradner, 2012). The output from the past of the neural network for example can be used to predict the future behavior of the agent as a way to improve learning in a simulation (Sutton, 1988). Other improvements in learning can be achieved by using genetic programming to try to surpass agent behavior in controlling a robot auto racing simulator (Shichel & Sipper, 2011). Agents in such environment should also be able to discard sub tasks that are no longer relevant to them achieving their overall goals (Wilkins, Myers, Lowrance, & Wesley, 1994).

Learning and Adaptive Agent behavior

To reach their goals, some agents rely on learning and adapting agent behavior by focusing on avoiding obstacles along their way (Stentz & Martial, 1995). This is called navigating locally in an environment. Other agents focus on behaviors such as wandering, path following, seeking or a combination of these three behaviors (Stentz & Martial, 1995). This is called navigating globally in an environment. Unfortunately in a dynamic environment, local or global navigation are not as effective in helping an agent to reach it goals. The combination of local and global navigation is seen as the solution to enable the agent to survive in a dynamic environment. This is achieved by the agent using local navigation to search for obstacles along its field of view to navigate around it and then passing that information to its map storage system. This system is used to create a step by step method to direct the agent to get closer to their goal (Stentz & Martial, 1995). As the agent moves closer to its goal, it can utilize a central system to monitor the information received from the local navigation and a remote system to fine tune the navigation of the agent to the goal (Chatila, Devy, Lacroix, & Herrb, 1994).

On the way to reaching the goal, it is important for the agent to create sub goals as it tries to navigate around obstacles present in the environment (Feng, Singh, & Krogh, 1990). The agent steers and aims for each of its sub goals until it has successfully navigated around the obstacles (Feng, Singh, & Krogh, 1990). In some cases it would be beneficial for the agent to back track and try another direction if it cannot go around an obstacle in its way (Gat, Slack, Miller, & Firby, 1990). Especially in a grid like environment, it is important for the agent to be able to process pixels in order to reduce delay in detecting obstacles in its path (Hebert, 1994). There is always a central unit that sends out commands letting the agent know what next action it should take after receiving feedback from the environment (Langer, Rosenblatt, & Hebert, 1994). Reflexive high-fidelity visualization can be used to provide feedback when detecting obstacles in the environment (Matthies, 1992). Also agents can learn how to detect and avoid obstacles by observing human agents navigating their way through similar paths (Pomerleau, 1991). Using GPS capabilities, satellites can also guide an agent around obstacle in an undefined path (Singh, et al., 1991). To accomplish its navigation, the agent must simultaneously use path following, obstacle discovery, and obstacle aversion. Path following involves combining static and dynamic satellite information to change the direction and speed of the agent. The goal is to minimize the error of the agent to stop them from going in the wrong direction. Along the directed path of the agent, the field of view of the agent is constantly scanned to find any obstacles by comparing the presence or absence of objects in a path. Finally, the agent performs obstacle aversion by reviewing the angles that will steer around the object in its path.

The factors that influence the ability of an agent to learn include which modules of the performance features are to be learned, what reaction is offered to study these modules and what illustration is used for the modules (Russell & Norvig, 2003). The modules of the agents are the link from its present state to movements, stimulus from the environment, how it changes and actions that would speed up meeting the simulation goals (Russell & Norvig, 2003). Modules can be learnt by the agent through suitable reactions to their movements in the environment. In the environment of the agent, supervised, unsupervised, and reinforcement learning are triggered by the type of reactions received by the agent (Russell & Norvig, 2003). Supervised learning occurs when an agent has sample inputs and outputs though it may or may not be able to see the reaction to its movement in the environment. Unlike supervised learning, the agent is not provided outputs for unsupervised learning in the environment but uses a probability system (Russell & Norvig, 2003). In reinforcement learning, the agent learns from the reward it receives from the environment when it takes an action that moves it closer to the goal. Past knowledge also helps in helping the agent to learn in the environment (Russell & Norvig, 2003). The various types of learning algorithm available to the agent include:

1. **Inductive Learning:** this algorithm focuses on learning that is supervised where the outcome of the action and next state relies on the current state of the environment (Russell & Norvig, 2003). This form of learning can be complex because of the guessing involved in providing outputs to the agent.

2. **Decision Trees Learning:** this algorithm is a less complex way of learning that involves attributes set as a input and attempts to forecast the value of the output (Russell & Norvig, 2003). Learning by decision tree can be broken down into classifying and regression tasks in the environment. Other types of decision learning are Naïve Bayes, reinforcement learning and neural networks.

3. **Ensemble Learning:** this algorithm focuses on using multiple estimates in the learning process of the agent in the environment (Russell & Norvig, 2003). An example of ensemble learning is using multiple decision trees to decide on the most likely output for the agent environment which may or may not involve enhancing the algorithm.

4. Online or Off Line learning: online learning occurs while the agent is present in the environment while offline learning is done before the simulation is run the environment (Ian Millington, 2009). Off line learning is used more than online learning because it allows for generation of validated path finding and movement data.

5. Intra-Behavior Learning: this algorithm focuses on a narrow part of an agent's behavior in the environment (Ian Millington, 2009). Modification of the total agent behavior is avoided which simplifies the pace of the learning and testing of the algorithm.

6. Inter-Behavior Learning: this algorithm is an experimental way in which agents can learn from the bottom how to act in an environment (Ian Millington, 2009). Intra-Behavior learning can be combined with Inter-Behavior learning in the environment.

7. Parameter Learning: this algorithm focuses on computing the value of one or more restrictions (Ian Millington, 2009). The numerical (fitness) values are used for navigation computation, cost tasks for route finding, weights and possibilities in making assessments.

8. Hill Climbing: this algorithm builds upon parameter learning by moving up in changing parameters that increase the likelihood of the agent getting to the goal (Ian Millington, 2009). This algorithm is very quick and produces accurate outcomes. The algorithm has many types such as momentum, adaptive resolution, multiple trials, annealing, direct methods, Boltzmann probabilities

Avoiding obstacles can be difficult for a single agent in an environment. Modeling complexity increases in the environment when you have multiple agents trying to navigate together as a group towards a common goal or exit. The simulation would have to synchronize the direction, speed and behavior that the agents exhibit in the environment. Flocking is one of the behaviors seen in birds that can be imitated by a group of agents (Crombie, 1997). Most specifically the split-up behavior of flocking allows agents to have dynamic obstacle avoidance with other agents in the environment (Crombie, 1997). In addition the sticking together and alliance properties allows agents to reach the goal together as a group (Crombie, 1997). To reduce the time it takes for the agents to complete the simulation, their individual speed and direction would have to be changed in a non-static way as more input is received from the environment (Brumitt & Stentz, 1996). The input received by a group of agents cooperating with each other to reach their goals, allows them to learn the behaviors needed to be successful in the environment (Moreira, 1995). Agents that are successful often use algorithms like genetic and neural networks; adapt their search methods and utilize specialists systems (Filho & Treleaven, 1994). Other algorithms can also make the agents movement to be stable enough to increase the speed at which they go towards their goals (Alonzo, 1996).

Evacuation

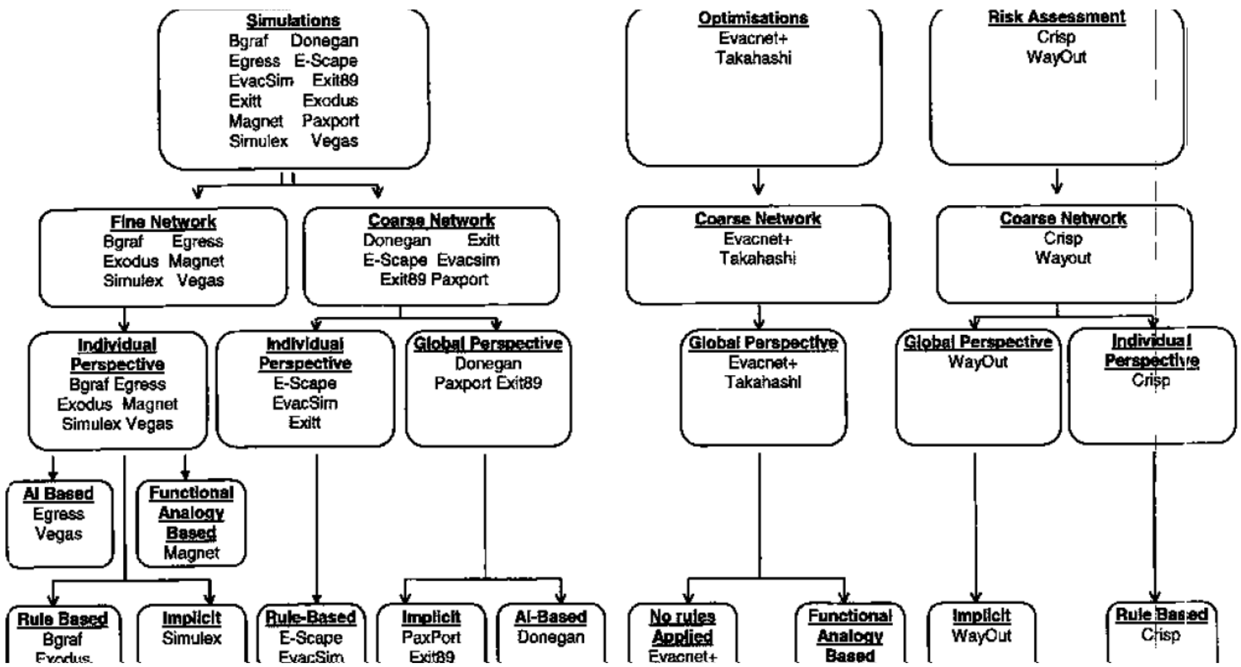
During an evacuation which is the mass departure, migration or flight from a situation, the main goal of occupants is to escape from danger to a place of safety in a short amount of time. There are various types of evacuations including building evacuation, airplane evacuation and train/subway evacuation. The various evacuations described below have various simulation tools that aid in running the drills without risking the life of volunteers.

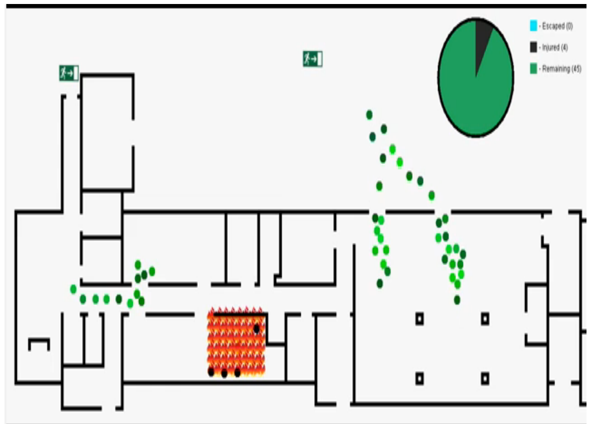
Building Evacuation

Floods, Fires, and Explosions are some of the causes of emergencies that can necessitate the need for evacuation from a building. The Occupational Safety and Health Administration

(OSHA) recommends an emergency action plan that includes a quick way of notifying authorities of fires, developing an evacuation policy and procedure and displaying in prominent places the work space map and safe areas (Chao & John L. Henshaw, 2001). The emergency action plan should include how to alert employees to an emergency, evacuation policy and procedures, conditions that necessitate evacuation, role of coordinators and evacuation wardens during emergency. It also should include evacuation routes and exits, tracking for employees after an evacuation, planning for rescue operations, medical assistance and role of employees in the emergency action plan (Chao & John L. Henshaw, 2001). Emergency action plan are also important in the home because fires caused 83% of deaths and 78% of injuries to civilians (Research, November 2013). Human attributes such as being asleep, physical disability and unconsciousness are some of reasons that increase the probability of death in home fires (Evarts, June 2011). The diagram below shows the various tools that are currently used for building evacuation (S. Gwynne, Lawrence, & Filippidis, 1999). In the top level of diagram, the tools are broken into simulators, optimization and risk assessment applications. Bgraf, Egress, EvacSim, Simulex are some of the examples that are found under simulation. Evacnet and Takahashi are tools that improve upon the study of evacuation. Crisp and WayOut are tools that focus on analyzing and assessing various risks found in evacuation. The second level of the diagram divides the tool into how they are rendered in the simulation environment fine network versus coarse network. Bgraf, Exodus, Egress and Simulex are example of fine network. Evacnet, Evacsim, Takahashi, Crisp and WayOut are examples of coarse network. The third level in the diagram focuses on the individual perspective versus global perspective. Bgraf, Exodus, Simulex, Egress and EvacSim are example of individual perspective. Evacnet, Takahashi and WayOut are examples of global perspective. The last level of the diagram highlights the AI

Based versus Functional Analogy Based versus Rule-Based versus Implicit. Egress and Vegas are example of AI Based and Magnet is an example of Functional. Rule-Based include E-Scape and EvacSim.





<u>Figure 4:</u> Diagram of Evacuation Tools

Airplane Evacuation

The National Transportation Safety Board (NTSB) recommends the action plan to be used when evacuation of commercial planes is needed in an emergency. Emergency studies have been conducted by the NTSB to collect information from passengers, airplane workers including pilots and airplane flight attendants and fire fighters (Board, 2000). The reasons for these studies are because over the last 10 years there were many incidents on planes that involved emergency evacuation. The issues reviewed by the safety board includes the problems in certifying airplane evacuation, how effective equipment are in evacuation, the depth of aircraft rescue and firefighting (ARFF) units procedures and communicating effectively during evacuation (Board, 2000). The most important part of a flight is the takeoff, approach and landing sections though emergency evacuation can be required when the plane is at a gate or taxing (BRIEFINGS, 2006). Before each flight, attendants should review standard operating procedures including commands that are effective in assertively communicating to passengers during evacuation and how to open emergency doors (Services, FLT_OPS – CAB_OPS – SEQ 12 – REV 01 – NOV. 2006). Also it is important to review incidents that on a flight in case procedures have to be updated or attendants retrained for future flights. Diagram below shows AirExodus the simulation tool. (Galea, 2013). AirExodus tracks the elapsed simulation time and the total number of occupants that have successfully evacuated the plane. The simulation can be paused, rewind and replayed back to see various important elements the researcher is trying to study during the simulation. The 3D environment in the simulation shows the various exits in the plane cabin with the various obstacles that the occupants would have to navigate to get out of the plane. There are various layouts representing different plane types that can be selected before the simulation is executed. The researchers is able to observe bottlenecks and what conditions cause queuing that could be resolved by providing additional training to airline personnel.

Figure 5: Diagram of Airline Evacuation

Train/Subway Evacuation

Chicago Transit Authority (CTA) has procedures for evacuation when train is completely in station, train is partially in station, train is out of station with another train on the same track, and train is out of station with or without another train on adjacent track (transitchicago). Irrespective of the location of the train, passengers are encourage to stay calm and follow instructions of the personnel who have received training on proper emergency evacuation procedures. Passengers are also encouraged to be familiar with the location of emergency exits, operator call buttons to communicate with train drivers and not to open rail car doors without instructions from emergency operators. It is important that trains are equipped with tool kits which contain fire extinguisher, pry bar, a hacksaw, a glow stick, and a first aid kit (Express, 2014). These tools will assist personnel to have the right equipment to use during an emergency evacuation. A map of each station can be included so personnel will know the approximate walking distance to the next metro station from an exit during an evacuation (Map, 2014). The diagram below shows the flow that passengers would need to take when evacuating a train based on its position of the station platform. When the train is at the station platform, the passengers can evacuate based on the closest exit to them on the platform. When the train is partially in the station platform, the passengers would have to move through the train until they reach an exit that is closest to the platform so they can evacuate safely from the train. In the circumstance where there is no platform closest to the train exit, the passengers may have to move to the next train in the adjacent track or move to the emergency walkway on the way to the station platform.

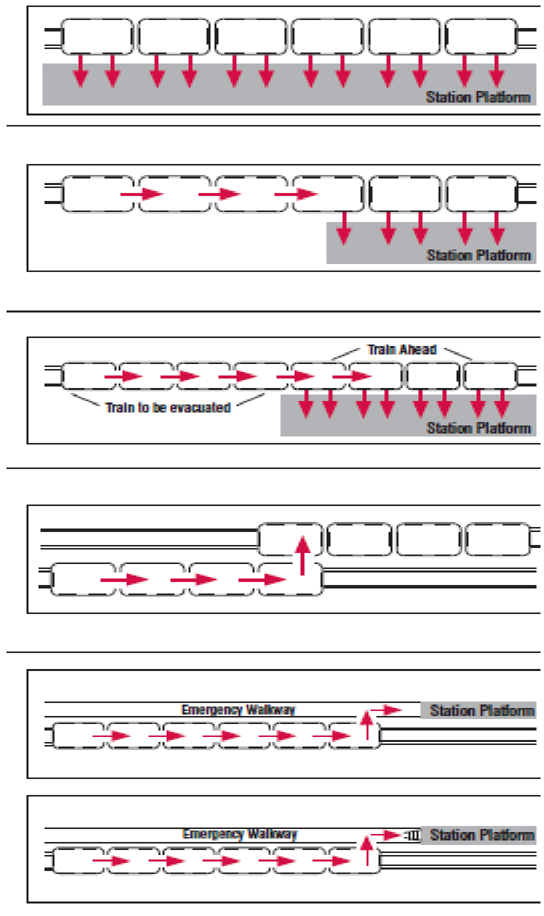


Figure 6: Diagram of Train Location

In summary, building, airline and train evacuations are very important parts in the study of how to evacuate people safely in an emergency, but the focus of this dissertation is building evacuation.

Multi-Agent System

Reinforcement Learning Algorithm

Reinforcement learning is the methods for learning that rely on past knowledge (Ian

Millington, 2009). It has three parts which includes a search approach for attempting various movements in a game, a way to grade the effectiveness of each movement and a learning rule that connects the two together (Ian Millington, 2009). The various parts have many ways of implementing and optimizing the methods depending on the kind of application. Reinforcement learning is very popular in artificial intelligence games and also used in developing the games of the future (Ian Millington, 2009). The problem that reinforcement learning is trying to solve is the ability of an agent to make smarter decisions as time goes by in a simulation. Developers of simulations may find it challenging to decide on the parameters to use to judge what qualifies as a smarter decision in reinforcement learning. Smarter decisions may rely on the actions of the agent in the environment or unplanned behavior that cannot be anticipated by the developer of the simulation. In either case the agent should be allowed to select the behavior at all situations and to figure out the most advantageous behaviors for any given state. The downside to the free will of the agent is that instant reaction may not be possible to find out whether the behavior selected was a smart decision. The outcome that lead to the agent reaching their goal can also be a result of numerous behaviors which individually may not receive a good reaction when done in the environment. Still the agent has to understand that though a good reaction may not be received for an individual behavior, combination of behaviors in a particular order can lead to a good reaction in the end. Reinforcement learning uses a state machine to manage the game environment and agents.

Yan-hua et al (Yan-hua & Xue-ren, 2011) focused on the importance of education and training of learners in a developing country like China. The lack of knowledge of what people should do during an emergency exerted a higher cost than even the actual disaster. This knowledge could be gained through short-term training that would help to reduce the response time to a disaster. The effect of the reduction in response time to disasters would lessen the impact of the crisis on large population size countries like China and would increase the quality of life of the people. The state in Q-learning algorithm contains all the information about the agent's world and memory of inputs and outputs received in the simulation (Ian Millington, 2009). The different states the agent may transition to throughout the simulation are the only ways they may learn using Q-learning algorithm. In the simulation, the agent states are created from different parameters such as direction, speed, location, coordinates of hostile and non-hostile agents. Q-learning algorithm does not analyze to understand the various parameters but assigns a numerical value to each one of them in the simulation. The simulation analyzes the various parameters to convert the state into a value that the Q-learning algorithm can implement in the environment. This is an advantage of Q-learning algorithm that makes it not difficult to apply over other algorithms like path finding where the algorithm has to convert the state to a value that the simulation can use in the environment. Q-learning algorithm does not try to simulate how the environment is structured in the simulation but has to assign a set of behaviors to each state.

In more difficult scenarios, it may be hard to get access to a particular action because it is only activated when an agent is in a particular state in the simulation. The reinforcement learning algorithm is activated after the agent has fulfilled a particular behavior in the environment and a good or bad or neutral reaction would have to be provided to the agent between the values of -1 and 1 (Ian Millington, 2009). This range of values represented by the reaction received by the agent may not be the same with similar behavior and state in the simulation. The agent moves to a new state after completing a behavior in the simulation. This new state may not match every time because there a myriad of variables that makes each new transition to a new state unique in the environment. Another advantage of Q-learning algorithm is that it can thrive in such a setting of ambiguity in behavior state transitions. This setting is called the knowledge record consisting of the start state, behavior taken, reinforcement value and next state (Ian Millington, 2009). The start state and behavior taken are search parameters in the quality information knowledge base and reinforcement value and next state are used to update the knowledge base. Reinforcement algorithm also provides a probing plan to deciding what behaviors is assigned for a particular state which may be random in the case where there is more than one particular behavior assigned to a state (Ian Millington, 2009). The Q-learning algorithm would have completed all learning after many repetitions and no new update of the quality information in the knowledge base. Performance of the Q-learning algorithm is big O (i) is the count of repetitions when all states and actions visited or big O (as) where "a" is behavior and "s" is count of states per behavior (Ian Millington, 2009). The algorithm has a learning rate, discount rate, randomness for exploration and length of walk parameters that match the learning rule and outcome in the environment. Reinforcement learning algorithm can also be broken down into passive in a fully observable environment with set strategy and active with the ability to decide on what behaviors to implement (Russell & Norvig, 2003).

Sharma, et al. (Sharma & Otunba, 2012) proposed that the damage of airline disasters could be reduced through proper study of emergency evacuation in a virtual environment. Users in such environment are totally immersed in the virtual world created that they are able to respond in ways they would have done, if they were actually put in that real-life situation. The virtual world included real-life aircraft, airport, runways and a control tower. The computer controlled the agents in the aircraft and the user controlled the agents that were trying to find the nearest exit of the plane (refer to Figure 7).

Lozano et al (Lozano, Gil-Gomez, Alcaniz, Chirivella, & Ferri, 2009) proposed a virtual cognitive system (VIRCOG) with two virtual environments that helped users trying to regain the use of their body functionality. The users faced walking in the street and shopping in a supermarket which provided realism to the environment created virtually. These worlds were similar to the virtual world in a restaurant and involved navigating their way in the environment.

Figure 7: Agent seek behavior (Sharma & Otunba, 2012)

Virtual Reality Intelligent Simulation System

The research shows that there are animated agents in augmented reality. These animated agents can be evaluated using various guidelines which can affect their social behavior. Also route finding algorithms can be used for evacuation behavior simulations which are also dependent on the cognitive architecture of the agent. WU, et al. (WU & Lin, 2009) investigated pedestrian evacuation behavior in relation to a multi-agent based paradigm that simulated behavior on a microscopic scale. The simulations demonstrated a crowd behavior by following the path of the agent from their current location to their chosen destination. Also there was an integration of individual agent behavior pattern. Chertoff et al (Chertoff, Vanderbleek, Flore, Gallagher, & M, 2009) developed a cognitive architecture for perception-reaction intelligent computer agents (CAPRICA). This library consisted of ideas of theory of mind, episodic memory and embodied cognition. It enabled them to study complex agent-agent and agent-human interactions. Polani (Polani, 2011) did some work on the decision making process of agents in the study of artificial life. Agents in artificial life usually made decisions based on their utility or goals to be achieved in the world (refer to Figure 8).

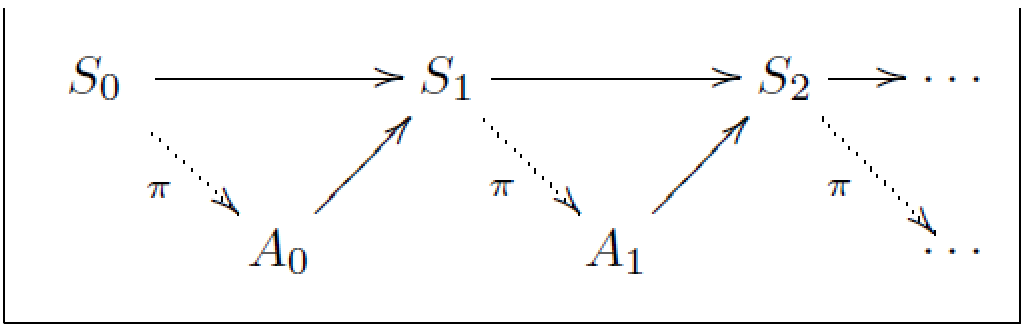


Figure 8: Graph of utility or goals (Polani, 2011)

There was a requirement of information processing which proved to be costly to the agent. Therefore the agent found ways to relieve itself, which was very important in the simulation of artificial life. Signoretti et al (Signoretti, et al., 2011) expounded on ways to reduce the computer intensive task of simulating multi-agent environments. The task involved making better, efficient way the agents react to change in the environment. This was done by controlling the agent's perception in an impromptu manner, to only important changes in their environment.

Johnson et al (Johnson, Thompson, & Coventry, 2010) moved on to evaluate how to measure which virtual environment a user will find useful in a simulation. The important criterion used in the selection process was the visual methods used in the virtual environment to arouse the emotions and mental state of the users to meet their goals. In addition to the visual methods, users also relied on their past experience when viewing the environment in the simulation. During the simulation, the users had built in expectations of what was going to occur next in the environment based on their past experiences (refer to Figure 9). Bruder et al (Bruder, Steinicke, & Wieland, Self-motion illusions in immersive virtual reality environments, 2011) explained how to identify superior virtual environments. These environments were known by their visual manipulation of the sight of the user (illusion) viewing the simulation in the real world. This meant that as the user moved in the simulation, their movement would have to be adjusted to match the pace of their movement in the real world. Adjustment would continue until the user's motion in the simulation closely matched their motion in the real world. The user motion is usually viewed in the simulation FOE by using HMD devices. Sobota et al (Sobota, Hrozek, Korecko, & Szabo, 2011) provided a laboratory to perform research on studying connections between interfaces communicating with an information system (LIRKIS). Improved realism structures can be used in cognitive learning, that yield production or design. The information and cognitive message systems can also be used as boundary VR technologies that aid in learning, design, medicine or robotics (Sobota, Hrozek, Korecko, & Szabo, 2011).

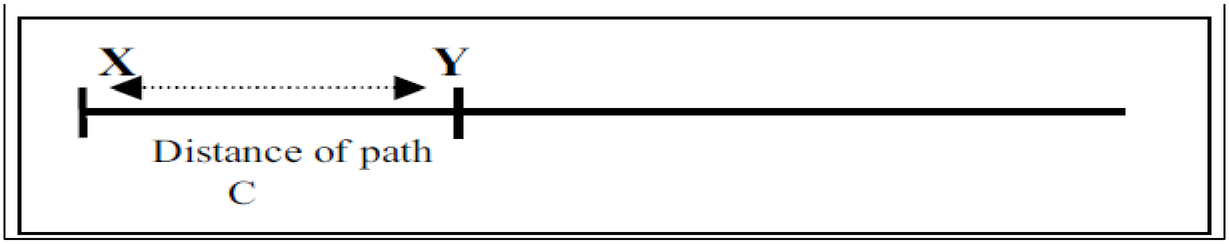


Figure 9: Image of visual path (Johnson, Thompson, & Coventry, 2010)

Chakareski (Chakareski, 2011) worked on a virtual technology used to reach a goal by improving the behavior of the user when sharing content on online community systems. The method of achieving the goal was done by first identifying factors that influence the public chart of information flow from the users to the online community (Chakareski, 2011). Next, the online community information was then added together using the influences of the starting place of the author of the content (Chakareski, 2011). The last step was achieved by coming up with the best path to take in reaching the goal of communicating the content in the online community. This resulted in the important feat of achieving reduced communication rate and content release difficulty. The online community that was very socially conscious had many agents (Ferrari & Zhu, Enabling Dynamic Roles For Agents, 2011). The agents were represented by users who acted as agents (MASs) playing various roles (RBC) as they communicated with other agents (Ferrari & Zhu, 2011). This meant that the agents on the way to achieving their goals would switch from one role to another as they interacted with other agents in their environment (refer to Figure 10).

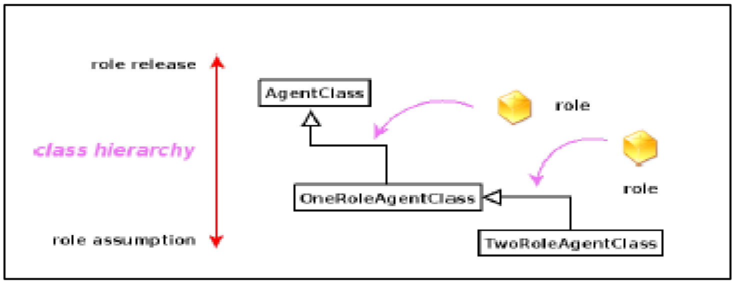


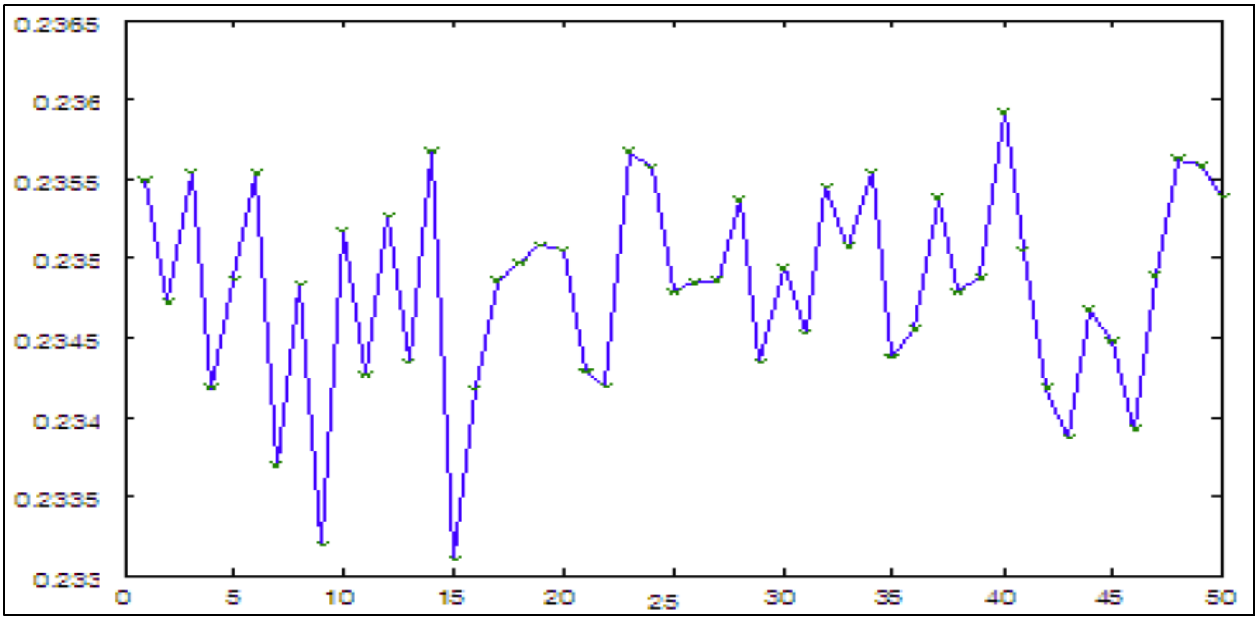
Figure 10: Graph of agent role (Ferrari & Zhu, 2011)

This would serve as a survival mode to act as a friendly agent when interacting with an agent trying to help them to achieve a goal and then switching to a hostile agent when interacting with agents trying to impede their progress in achieving their goals. To switch between the various roles, agents must have had an understanding of the logical classification that regulated the behaviors. The behaviors exhibited in the scenarios they face and must be able to work together and harmonize the actions they would take on the way to achieving their goals (Ferrari & Zhu, 2010). The agents also needed to be able to recognize the roles that other agents were presently playing in the environment without consulting an independent source of the knowledge of all roles. The roles were saved in storage areas to be made available when needed by the agents. They were governed by rules that managed the pick up and put down of roles by the various agents in the environment (Ferrari & Zhu, 2010).

Genetic application

Yi (Yi, 2010) put forth a VRP solution that used a modified GA to find the solution to search for a goal. This solution improved the time it took for a narrow, best possible answer to be achieved by the agent. It involved maintaining agents that performed well in reaching the goal while providing adequate interaction with other agents in the environment. This provided an advantage in using the improved GA and would make it easier to apply it to other scenarios.

These scenarios differed based on the circumstances and the environment in which the search of the goal was conducted (refer to Figure 11).



<u>Figure 11:</u> Graph of genetic algorithm (Yi, 2010)

Mehra (Pankaj, 2012) explained that circumstances shaped human understanding when trying to achieve a goal. The circumstances included emotions & outcomes; network of people; network of information; network of possessions; instance & location and contracts & obligation. Emotions & outcome encompassed family; network of people comprised professional and social networks; network of people constituted competency, parenting and interests (Pankaj, 2012). Network of things incorporated customer service and product ownership; contracts & obligation combined project; collaboration and vacation and instance & location included event, trip, task and meeting (Pankaj, 2012). Distance and direction information were also used in EA as a way to search for a goal (Thangaraj, Pant, Chelliah, & Abraham, 2012). It used a select number of variables to achieve the result of getting quickly to the goal. The increased speed occurred because a large variety of agents were available in the environment, all working together to obtain the goal. OCDE algorithms used OBL rules to achieve the goals (Thangaraj, Pant, Chelliah, & Abraham, 2012).

Artificial Life Simulation Methods

Choi et al (Choi & Zhu, 2012) proposed a way for assigning goals to multiple agents and the path the agents would take to reach the goals. Tasks were assigned to agents based on a public sale system where agents submitted a bid of cost to achieve a goal to the central agent. The central agent then assigned the goals to the agents based on the lowest cost bid submitted to reach the goal (Choi & Zhu, 2012). Also agents could submit a bid on more than one goal at the same time to the central agent but the single goal bid was highly encouraged because of its simplicity. Next the agents, after winning a bid for a goal, would have to chart the route from their present location to the finish goal. The environment for the agent could be static where things such as layout, obstacles and path direction were not changing nor be dynamic where those things were constantly changing (Choi & Zhu, 2012). The dynamic environment closely resembled the real world in which, to reach a goal, the agent would be operating. The agent would have to take in consideration how to navigate toward the goal while avoiding colliding with agents and obstacles on its path. BDI built on the behavior of agents in dynamic environments who reacted to events with broad plans that changed as they faced new scenarios in the process of reaching their goals (Scerri, Hickmott, & Padgham, 2012). It allowed the user to watch what decisions agents were taking and why they were taking them in the simulation. Also, users were able to make changes to the simulation, see the effect of these changes and give feedback in the creation of the simulation (Scerri, Hickmott, & Padgham, 2012). McIntire et al (McIntire, Havig, & McIntire, 2009) laid out important factors that need to be taken into consideration when creating simulations with agents acting as humans. The agents acting as humans would have to be assigned to tasks and goals that are easy for humans to solve but not computationally difficult for the agents to solve. This means that the agents would have to be able to pass the CAPTCHAs tests to mimic the behaviors of humans with the agents (McIntire, Havig, & McIntire, 2009). The behaviors that the agents are trying to learn from the humans can be achieved in a community environment (Penaloza, Mae, Ohara, & Arai, 2012). This would mean that agents in our simulation would imitate other agents that seem to know the path to get to a goal, such as evacuating a building in an emergency (Penaloza, Mae, Ohara, & Arai, 2012). The agent should be persuaded to follow an outsized number of agents going in the same direction of a goal, rather than a small number of agents going in the opposite direction of a goal (Penaloza, Mae, Ohara, & Arai, 2012). For this to occur accurately, the agent would have to be in a close range of field of view so they can identify the size of the group of agents that are heading in one direction and to ignore other agents going in the wrong direction (Penaloza, Mae, Ohara, & Arai, 2012). This is made possible by the SLT ability of the agents, which is also available in agent behaviors, similar to groups going in one direction or staying in one location (Penaloza, Mae, Ohara, & Arai, 2012). Liu et al [28] proposed a MacroAEM that uses three behaviors of rivalry, collaboration and self-centeredness as ways to improve the ability of agents to reach their goals. The goals are achieved in an environment of multiple agents interacting together using a hybrid of GA. To implement the algorithm, agents exhibit the behavior of self-centeredness when they move to capture resources in the environment that would help them in achieving their goal (Jing, Zhong, & Jiao, 2009). The agent would also exhibit the behavior of collaboration with other agents when they could all pool their efforts to achieve their goals together more efficiently as one (Jing, Zhong, & Jiao, 2009). Agents also need to be able to possess the ability to know how to pick an efficient solution in order to reach a goal (Cristina &

Santos, 2010). This ability is implemented using behaviors seen in strengthening the education and knowledge of the agents as they work to achieve their goal (Cristina & Santos, 2010).

Prithviraj et al (Prithviraj, Cheng, & Li, 2009) put forth a way for agents to work together when trying to achieve a goal in an environment. This process involves mimicking the swarm behavior of birds or ants in nature when they all cooperate together to reach the goal of capturing their food. The environment contains a limited amount of resources to use to explore it and the agent has to search to reach their goal. The agents all fan out in the environment exploring it to see what paths they can all take to get to the goal. Once this spy mission is completed, the agents return to strategize to select the best course of action that they can all take to get close to the goal. This is a divide and conquers method of efficiently building on the large number of agents to provide a division of labor to get to the goal in a shorter amount of time.

Summary

The following table summarizes the pros and the cons of the goal applications presented.

Table I: Overview of the different goal applications

No.	**Goal Application**	**Object**	**Pros**	**Cons**	**Examples**
1	Virtual	Real-life aircraft, airport, runways and a control tower.	Cost effective to run simulation	Set up time of simulation	Example applications are AirExodus and buildingEXODUS that address the arrangement, setting, conduct and technique of the evacuation procedure.
2	Cognitive	Simulation user	Based on past experiences	Expensive to setup	CAPRICA and VIRCOG.

3	Online	Simulation user	Fast communication	Constant change of user role	AvatarSim and MAS.
4	Genetic	People, information and possession	Similar to human brain	Many input variables	VRP and Goal Finding Application.
5	Task assignment	Agent in dynamic environment	Works with multiple users	Set up time of simulation	Simulex, Egress and Path Finder.
6	Divide and Conquer	Swarm of birds or ants	Efficient system	Limited resources	Goal Finding Application and Egress.

Comparison and limitation of the goal finding applications

In conclusion, the above goal finding applications worked best in scenarios where it was cost-effective to run simulations, past experiences of people were taken in account; fast communication was utilized; mimicking of the human process of thinking was done and multiple users were involved in searching for a goal in an efficient system. Virtual goal finding applications focuses on using 3D technology to model objects that are seen in the real world. Objects include real-life aircrafts, airport runways and control towers. Using these object the researcher can manipulate their placement in the virtual world and observe the effect on the simulation. This advantage allows the simulation to cut the cost of running various drills and experiments that mimic the real world. The disadvantage of the virtual is the high set up time in the beginning of the simulation. The designer has to invest resources to accurately depict the environment they want to model to make sure that it can be validated in the real world. Cognitive goal finding application focuses on the thought process of the simulation user in the environment. It can take advantage of the past experiences of the user to aid in the simulation. The technology that would be used for the simulation can make it expensive to set up the cognitive application. Online application also focuses on the simulation user like cognitive application but it has the advantage of a fast communication with high speed networks but it suffers from constant change of user roles in the simulation. Genetic application relies on people, information and possessions as the object of the simulation. It utilizes the structure of the human brain with neurons communication to pass information in a fast manner. The disadvantage of genetic application is the high processing power needed for many input variables in the simulation. Task assignment has agents as the object in a dynamic environment. It allows for a multi-agent assignment but like virtual it suffers from high set up time of simulation. Lastly, divide and conquer is utilized by swarms of birds or ants and it is very efficient. The main disadvantage is the limited amount of resources available in the simulation. Based on the comparisons of the various types of goal finding application simulations, the research will show that when a smaller number of intelligent agents collaborate with each other, they reach more goals of finding exits in a shorter time frame, which will be discussed in detail in section 3.

CHAPTER 3

METHODOLOGY

Introduction

Previous research (Yan-hua & Xue-ren, 2011) has shown that goal finding applications work best in scenarios where it is cost effective to run simulations. Also, when past experiences of people are taken into account, goal finding applications are very successful in decision making strategies for evacuation (Johnson, Thompson, & Coventry, 2010). In addition, when communication is done quickly (Chakareski, 2011), among multiple human users (Choi & Zhu, 2012), goals can be found efficiently (Prithviraj, Cheng, & Li, 2009). These goal finding applications have limitations when agent behavior is involved. When a small group of agents collaborate with each other, they reach more goals of finding exits in a shorter time frame. This research combined GAs with NNs (Neural Networks). The focus was to create agents that can learn how to find a goal. We first created a lot of agents (using a typical agent model), gave them their brain (the NN, each agent has its own), fed the data about its current motion into the network, got the outputs, and used them to move them again (hopefully closer to the goal). We did this for some time, and then we saw which agent got to their goal using its brain (the NN). A GA fitness function was used for evaluation. After evaluation, we sorted them in some order, chose the best ones, let them be parents and have children. We looped until agents actually learn (refer to Figure 12).

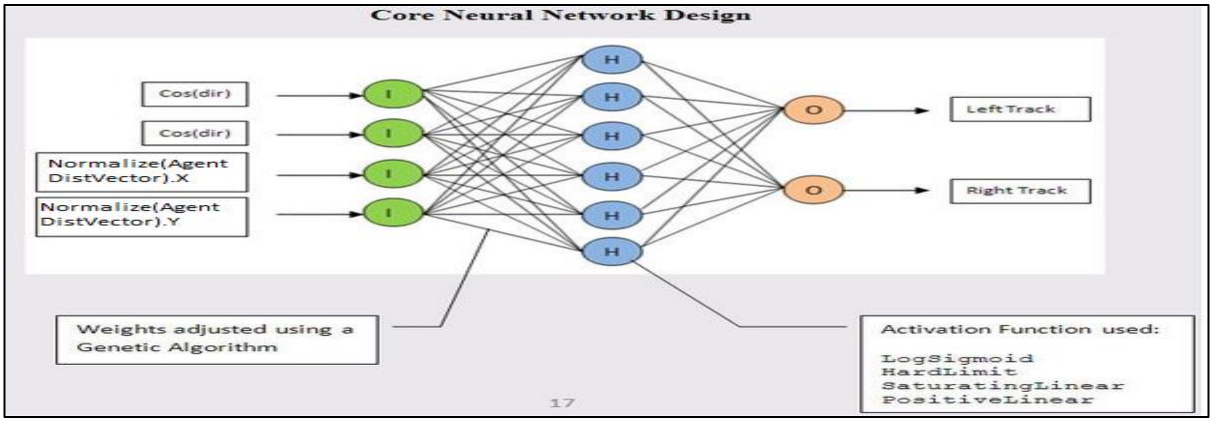


Figure 12: Graph of Neural Network

The NN was represented by a list of numbers (Notice that in EStrings, it was a list of characters, and here it's a list of numbers), these numbers make up the weight of the NN. The solution space was again the permutation of them, and the problem was to find the combination which would steer an agent towards the nearest goal (refer to Figure 13).

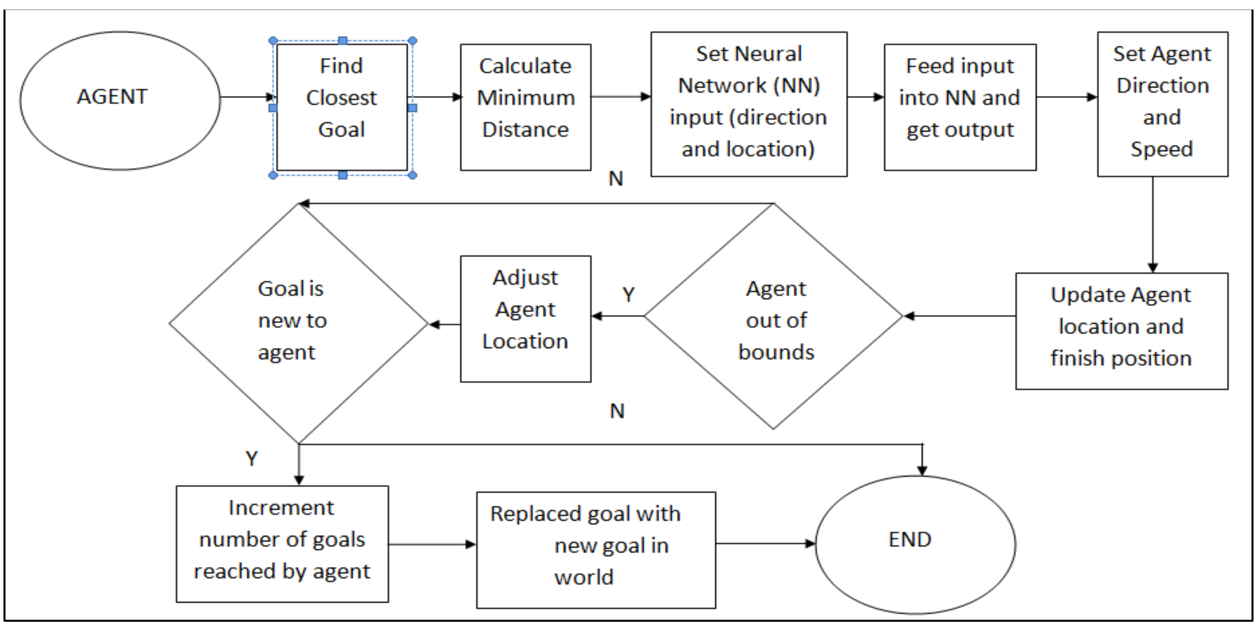


Figure 13: Diagram of agent goal activity

Figure 14 shows the combination of the genetic algorithm and neural network. The inputs we focused on were the direction and distance of the nearest 4 non hostile agents including the active agent in the simulation. These input along with their corresponding weights were fed into the neural networks and the output behavior of flee, seek, arrive, wander and cohesion were used to control the agent direction and speed as it moved closer to the goal.

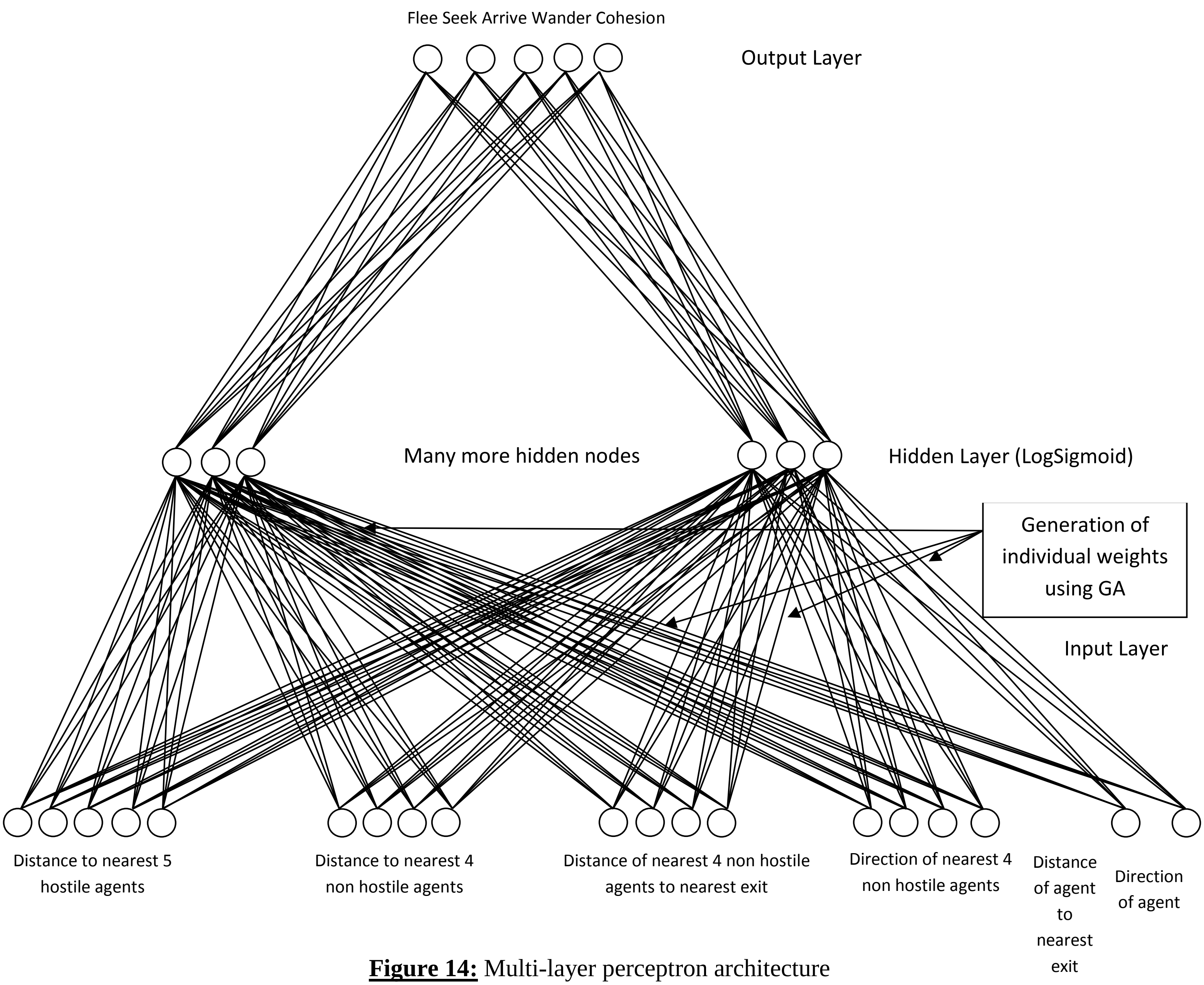


Figure 14: Multi-layer perceptron architecture

Figure 15 depicts the fuzzy logic model used to implement the panic behavior in the simulation. The input included the current panic level of the agent (low, medium and high) and distance from the exit in the room. Inputs were then scaled and using fuzzy rules contained in the knowledge base produced the direction, speed and behavior output of the agent.

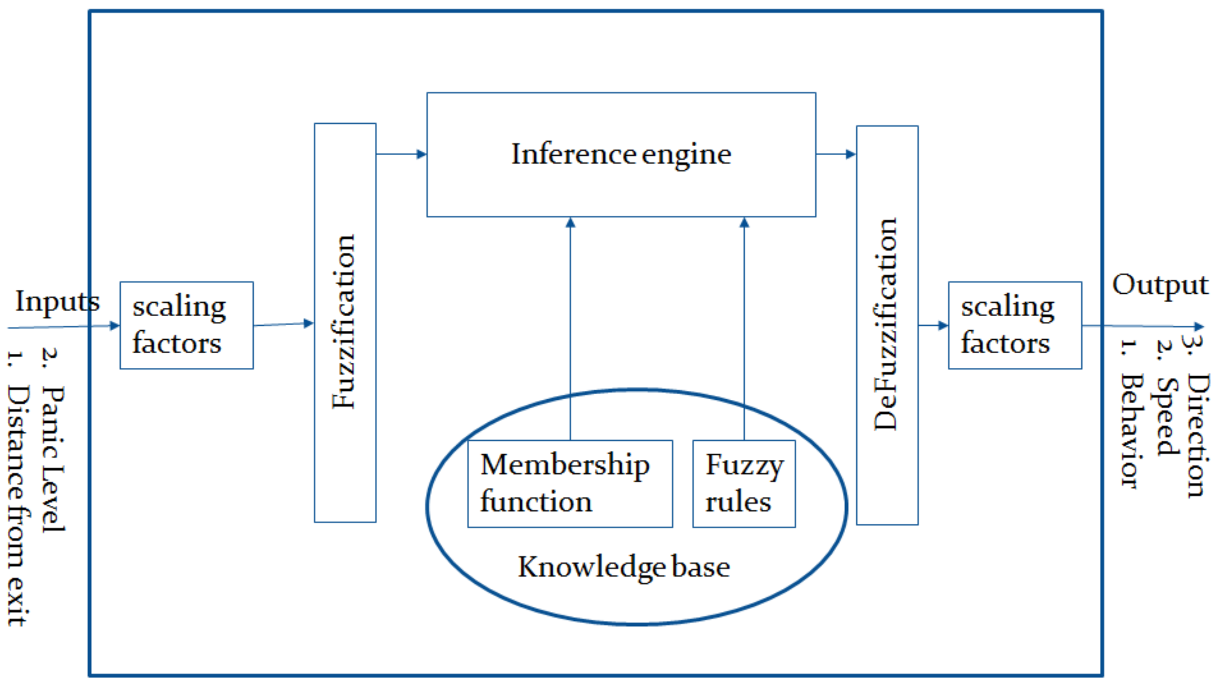


Figure 15: Fuzzy Logic Model—Panic Behavior

The combination of the genetic algorithm and neural network with panic behavior composed the implemented C-Sharp application.

Implemented System

The implemented system was a C-Sharp application, which simulated evacuating a room in a building by agents. Agents included steering and intelligent who were trying to navigate their way around static obstacles (tables) and dynamic obstacles (other agents) on the way to one of two exits in the room. The agents utilized neural network and genetic algorithm to guide them as they searched for exits during the simulation (Sharma S., 2013). The framework on which the simulation was built included an AI-controlled character that made use of 19 input values: the distance to the nearest 5 hostile agents, distance to the nearest 4 non hostile agents with their distance to nearest exit and direction values, and the distance to nearest exit and direction of the AI. The input values such as hostile and non-hostile were set in the input file using the keyword ‘type’ as shown in appendix D. We assumed that there were five different output behaviors agents could exhibit on their way to an exit: flee, seek, arrive, and wander and cohesion.

We used a network with three layers: input layer and output layer, plus an intermediate (hidden) layer. The input layer had the same number of nodes 19. The output layer had the same number of nodes as there were possible outputs: 5. Hidden layer matched the 24 count of the input layer.

Implemented Research Methodology

Methodology—Geometrical approach of developing goal finding application using C#. Intelligent agent was known for its variables which allowed it to maintain current state as it transitioned based on the input it received from the environment. The data of the agent was stored in the below variables:

i. X and Y coordinate of agent location
ii. dir was the current direction of agent
iii. myExit was closest point of exit to agent
iv. agentMaxSpeed can be increased or decreased
v. agentTopX and agentTopY for drawing agent circle shape on screen
vi. agentBottomX and agentBottomY drawing

vii. agentLeftX and agentLeftY drawing

viii. agentRightX and agentRightY drawing

ix. clientSize was the width and height of screen

x. indMyExit was for tracking index of exit reached by agent in an array

xi. exitCollected was the count of total number of exits reached by agent so far

xii. agentNumber unique identifier of ant

xiii. net was the private neural network of agent

xiv. myWorld was the agent's copy of the world environment in which the simulation took place.

xv. startPosition and finishPosition used to find the distance traveled so far by agent

xvi. myexitCollected array

Algorithm and Pseudo code

Agents had nine general categories of behavior: seek behavior where an agent was directed towards a target position; flee behavior was opposite of seek; arrive behavior was a variant of seek where agent decelerated towards a target position, pursuit behavior was where an agent went after another agent; wander behavior was a random walk through the environment; path following behavior moved agent through a series of waypoints; cohesion behavior kept a group of agents together, alignment behavior kept agent aligned with other neighbor agents and separation behavior was opposite of alignment. We focused on a subset of five behaviors in our algorithm which were seek, flee, arrive, wander and cohesion as a foundation to build the rest.

The agents adjusted their behavior based on their distance from the two exits in the world. Each agent calculated the distance of the closest exits from their current location. The agent transitioned to arrival behavior when it passed a threshold distance from an exit. When an agent was near a hostile agent, it entered a flee behavior while the hostile agent entered a pursue mode. If an agent threshold distance level dropped very low, it entered a random behavior. Otherwise, it would be in a seeking mode. Agents changed their threshold distance level by combining genetic algorithm and neural network with inputs current distance, direction and outputs new speed, direction and position. The world kept track of how many agents had exited the environment and how long the simulation was running. Agents kept track of visited places that could be shared with other agents that exhibited cohesion, alignment and separation behaviors.

Agents were symbolized independently, moving everywhere in their world freely. We needed an accurate physical model making sure that agents could not walk through walls in the environment and went around obstacles in their path. Also we needed full steering behaviors. Agents would often want head for one of two exits which required navigation through the world while avoiding static obstacles (walls and blocks) and dynamic obstacles (other agents) with path finding. In addition, we also fed the data about agent's current motion into the network. Input data fed include:

a. Math.Cos(dir), was sin as long as it's between 0 and 1

b. Math.Cos(dir)

c. (myExit.X - x) Normalized exit distance vector. Let agent know about nearest exit

d. (myExit.Y - y) Input data was weight adjusted using a genetic algorithm. Got outputs and used them to move closer to exit. Output included: Left Track and Right Track

Activation function used LogSigmoid, Hardlimit, SaturatingLinear and PositiveLinear

dir += (rightTrack - leftTrack) * (Cosmos.maxForce / 100); (refer to Appendix A for Cosmos definition)

curSpeed = (rightTrack + leftTrack) / 2;

x += Math.Sin(dir) * Cosmos.maxSpeed * curSpeed / 10;

y -= Math.Cos(dir) * Cosmos.maxSpeed * curSpeed / 10;

Pseduocode

```
For each agent initialize new network
       Set numInputs to 19.
       Set numHidden to 1
       Set perHidden to 24
       Set numOutputs to 5
       Set numNeurons by adding numInputs to multiplication of numHidden and perHidden and then
       adding the numOuputs
       Set numWeights to ((numInputs + 1) * perHidden) + ((perHidden + 1) * perHidden) *
       (numHidden - 1) + ((perHidden + 1) * numOutputs)
       Set numLayers by adding 1 to numHidden
       Create an array of Layers equal to numLayers

       Initialize a layer with initNumNeurons set to perHidden and initInputsPerNeuron
       set to numInput
       For each numHidden
              Initialize layer with initNumNeurons set to perHidden and
              initInputsPerNeuron  set to perHidden
       Initialize a layer with initNumNeurons set to numOutputs and initInputsPerNeuron
       set to perHidden

while simulation is running
       Update agent position
              Feed input into network and get output
                     For each numLayers
                            For each layer numNeurons
                                   For each layer neuron numInputs -1
                                          netInput is sum of netInput with curInput
                                          multiplied by layer neuron weight
                            layer neuron weight is subtracted from netInput
                            curOutput is set by transfer function return value
                                   if LogSigmoid
                                          return value
                                   else if HardLimit
                                          return value
                                   else if SaturatingLinear
                                          return value
                                   else if PositiveLinear
                                          return value
                                   else
                                          return 0
              Change agent behavior to flee, seek, arrive, wander or cohesion based on
              output
              Change agent speed, direction and finish position (x and y coordinates)

If maxIteration is reached
   Update
   CalculateFitness and set totalFit
   PrintBest totalFit
   NewGeneration
   generationCount is incremented
```

The novelty of the pseudo code above was in the combination of neural network and genetic algorithm to aid the agent in finding an exit. This was achieved by using genetic algorithm to dynamically adjust the weight of each of the inputs that were fed into the neural network to produce the output behavior of the agent. As the simulation was running, the inputs that were not helping the agent to reach the exit faster were given lower weights and higher weights were assigned to inputs that moved the agent closer to the exit. Grading of the inputs was done such as distance and direction of the hostile and non-hostile agents that were in the field of view of the agent who was trying to evacuate throughout the simulation. The adjustment of the weights mimics the real life behavior of an individual for self-preservation to invest in any opportunity that allows them to find an exit faster in an emergency situation.

Expansion of Neural Network and Genetic Algorithm for more people

Bottlenecks occurred as the people tried to evacuate at the same time and using the same exit and therefore their speed was reduced and resulted in slower evacuation. Identification of bottlenecks was important. Automated testing was done on the simulation where a researcher designed a building and then used the program to run a series of tests, varying the position and number of agents and obstacles. After a defined series of tests have taken place, the program would use the collected metrics to automatically determine optimal placement for exits, door and obstacles. The algorithm can be expanded for more people (500+) by increasing the number of agents in the input file that is fed into the simulation. The number of input variables stays the same irrespective of the number of agents in the simulation because the algorithm takes into account the field of view of the individual agent that is set to the same number of nearest hostile and non-hostile agents closest to the agent.

Genetic Algorithm & Neural Network Steps

1. For each agent, initialized new network with number of inputs equal to 19, number of hidden layer equal to 1, nodes per hidden layer equal to 24 and number of output equal to 5.

2. The number of neurons in the network was initialized to number of inputs + (number of hidden layer * nodes per hidden layer) + number of outputs = 19 + (1*24)+5=48.

3. Number of weights was initialized to ((number of inputs + 1) * nodes per hidden layer) + ((number of node per hidden layer + 1) * number of nodes per hidden layer) * (number of hidden - 1) + ((number of nodes per hidden layer + 1) * number of outputs) = ((20*24)+((25)*24)*0+((25*5))= 605.

4. The number of layers was initialized to number of hidden + 1 = 2

5. Initialize layers array of size 2. Layer 1 was initialized to 24 neurons and 19 inputs per neuron. Layer 2 was initialized to 5 neurons and 19 inputs per neuron. The weight for each input was initially generated between a range of 0 and 1 assigned at the beginning of the simulation.

6. The agent was assigned a unique number and given a start and finish position that was the same at the beginning of the simulation.

7. Fed input into the network.

8. For each layer, iterated over the number of neurons and the number of inputs in each neuron and calculated the netInput by adding up the sum of the individual weights multiplied by the input value.

9. Subtracted the last weight value from the net input.

10. The output of each neuron was calculated from calling the transfer function with the net input. The transfer functions that were used are LogSigmoid, HardLimit, SaturatingLinear and PositiveLinear. If none was selected it returned 0.

11. The output was returned to the agent to change their behaviors to flee, seek, arrive, and wander or cohesion which changed the direction, speed and finish position (x, y coordinates) of the agent.

Evacuation drill in Room 109 in Computer Science building at BSU

Layout of the room

The first floor computer lab in the computer science building at Bowie State University was used for the study. The room had two entrances that can be used to get into it from outside the hallway. There were also a row of tables and chairs in the center of the room and also on either side lining the walls of the room. Two elevators are seen on either side of the hallway to get to the other floors of the building. Video cameras are also located in the hallway of the computer science building to monitor the movement of students walking to and fro to the various classrooms in the building. Figure 16 represented the type of wander behavior of the students as they walk back and forth the hallway.

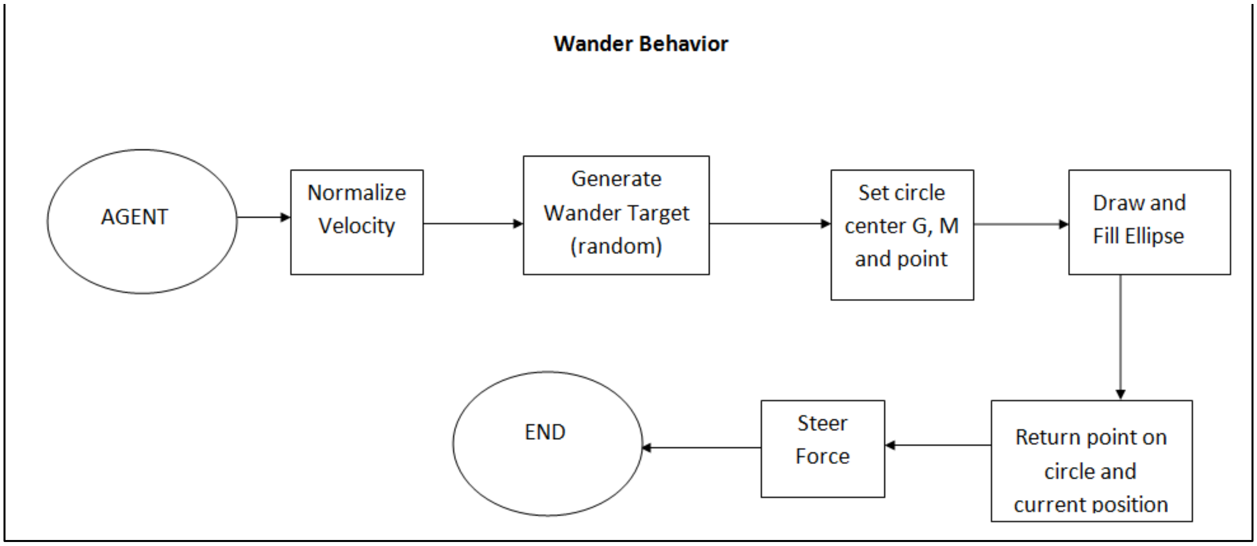
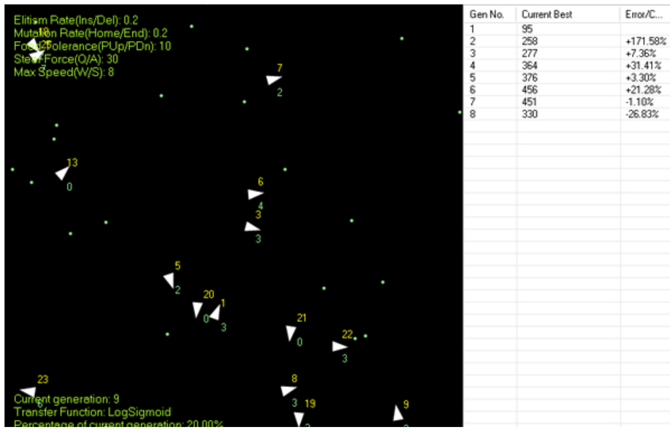


Figure 16: Diagram of wander behavior of students as they work back and forth the hallway

Population

The participants were all students. At least 7 students took part in the evacuation drill; out of them a minimum of 3 females. The homogeneous population of the college campus (age was between 18 and 30) was taken into consideration when generalizing to an average population. However, at the same time, we accounted for the age restriction by adapting the speed in our simulation (refer to Figure 17). The limitations of the study were the evacuation drill did not incorporate children, blind people, and handicap people. But the built application can include the different behaviors for people which were not validated by the real-time drill since we were not able to recruit such people for the drill. Students with disabilities were not included due to potential conflict with HIC (Human investigation committee) approval.



<u>Figure 17:</u> Screenshot of adapting the speed in our simulation

The green dots represented the goal, the green number with each agent was the number of goals collected, the yellow number was the agent number, and the list on the right showed the details of the previous generations. Results were made general to a large diverse group by changing the speed parameters. In addition the following rules were taken into consideration:

- There was no attempt to change the visibility of the room with hazards like smoke.
- Students were made aware of the number of exits in the room.
- Students were told to take extra precaution when performing the evacuation drill so they would not be injured.
- The room was not crowded when performing the evacuation drill.

Data Collection Strategies and Steps

After getting the IRB approval, real time evacuation drill was conducted in Room 109 in CS building at BSU.

The implemented C# application was compared with:

1. **Real-time data from the evacuation drill:** The purpose of the study was to collect data for students of different ages (above 18 years) who participated in the evacuation drill. The evacuation drill occurred in Room 109 in Computer Science building at BSU.

The drill was used to study building evacuations in multi user environment and could also be used as an education and training for emergency responders. Evacuation drills was used to study human behavior that could be evaluated in the real world. The data collected was used to validate goal finding application and safety recommendations to make emergency evacuation safer. We used 2 cameras to record and collect data on the route, time, path and which exit the students took during the evacuation.

The students were asked to participate in the evacuation drill scheduled for a 60-min session at their convenience. Upon arrival at the study the participants underwent informed consent procedures in which drill personnel explained to them the procedures, rules, and read the consent form with them. 10-28 participants were asked to participate in the multi-user environment. During the session the participants were placed at various locations in the room with computers to perform their normal daily tasks like checking their email and working on their class projects. When the timer started with the cameras rolling, they were given instructions to evacuate the room by avoiding obstacles on their way to reach the goal. The goal for building evacuation was to reach one of two exit doors. After the task was completed, the participants were given a survey questionnaire including questions on their experience. They were debriefed and given opportunity to ask questions or express any feedback they had.

The participants were not paid and their participation was voluntary. Adult participants (above 18 years of age) were selected from BSU campus. The participants were recruited through the use of announcements around BSU campus. Participants below 18 years of age were excluded from the study. We recruited 10-28 participants because the maximum room capacity where the evacuation drill took place was 70 students. One session had 10-20 students participating in a multi-user environment. We were able to have 2 sessions. Risks to the participants were minimal. There was a risk of fatigue from rushing to evacuate the room. To minimize the risk, the participants were given specific instructions before the evacuation drill and were also informed that they could leave at any time during the drill.

The consent procedures were conducted in the same room where the study took place so that the research staff could instruct them on how to stop participating in the drill. Research data would be retained for two years after the completion of study. This would allow sufficient time for analysis and publication of research data. After three years the research data will be destroyed. The survey given to the participants after the evacuation drill is shown below.

Survey questionnaire

Evacuation Drill

Please use the following values for your self-evaluation and when grading the group project:

5=outstanding

4=significant

3=average

2=marginal

1=almost none

1) How would you rate the overall quality of the evacuation drill?

2) Overall how would you rate the participant instructions given to you before the drill?

3) How realistic was the evacuation drill for you the participant?

4) Did you follow the drill personnel instructions (Look for the nearest exit when told to evacuate)?

5) Can this drill be used for educational or training purposes?

6) Is the layout of the room sufficient enough to learn how to evacuate a building?

7) How would you rate the contribution of the drill personnel?

8) Do you think participating in this drill would help in learning evacuation skills?

9) Recommendations/suggestions to improve on this evacuation drill?

The data extracted from observing the timed evacuation drill and from surveys of the students was compared and contrasted to other larger scale disaster studies.

2. Commercial evacuation simulator like Simulex and Pathfinder: The dependent research variable in simulator was evacuation time and independent research variable was number of people (occupants), number of simulation runs and type of behavior. Below is the algorithm used to calculate the number of time and goals reached by each of the agents.

```
private void Update() {
    int totalFit = CalculateFitness();
    PrintBest(totalFit);
    lastTotalFit = totalFit;
    NewGeneration();
    inCG = 0;
    generationCount++; }
  private int CalculateFitness()
  {
     int totalFit = 0;
     for (int i = 0; i < numAgents; i++)
     {
         totalFit += agents[i].GetGoalCollected;
```

```
}
return totalFit; }
```

Data analysis described the responses for the research questions; determined the overall trends, distribution of the data, described representative characteristics in samples mean, standard deviation, and range for continuous scaled variables, frequency, and percent for categorical scaled variables. Pearson correlation coefficient criterion/predictor variables contained continuous interval data. Two-tailed test to find inferences for the hypotheses.

Comparison of goal finding application and real fire drills

Figures 18 and 19 shows the comparison of three fire drills and goal finding application (Peacock, Averill, & Kuligowski, 2009). Figure 18 provides the variety of people movement rate for three fire drills. The camera location was on every other floor landing and captured people evacuating as they made their way down the building. The graph shows the mean speed of the people evacuating on the Y-axis and X-axis representing the floor where the people were first seen in the building. Total number of people in the drill at the six-story building was 277, 127 in stairwell A and 150 in stairwell B. Eleven-story building had 134 people participating during the drill. 727 people participated in the drill at the Eighteen-story building. The number of people evacuated in the time interval of 0.2, 0.4, 0.6, 0.8, 1.0, 1.2, 1.4. People speed in six-story building was .83m/s as compared to the .62m/s in the eleven-story building and .40m/s in the eighteen-story building. Thus, speed of people was the highest when evacuating the six-story building. The error rate in Figure 19 tracked was similar to the path of the graph in Figure 18.

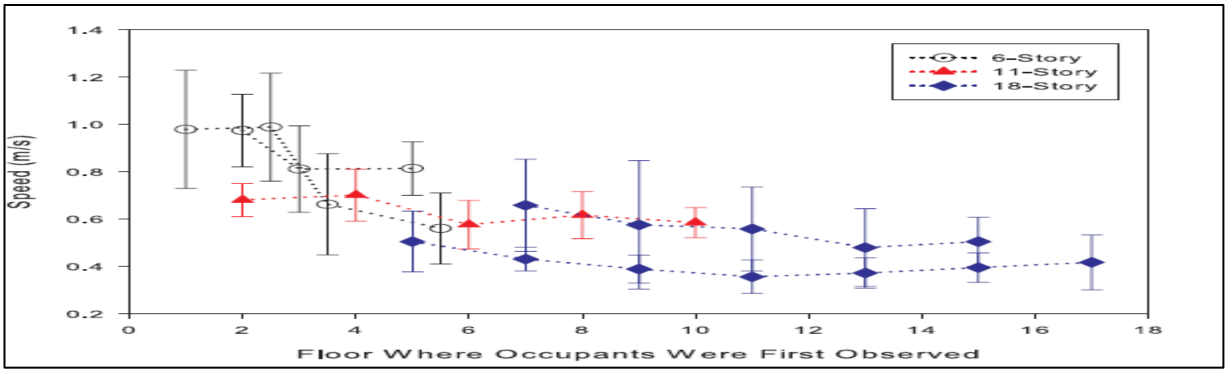


Figure 18: People progress speeds including standard deviation going down a stairwell during three fire drill evacuations (Peacock, Averill, & Kuligowski, 2009).

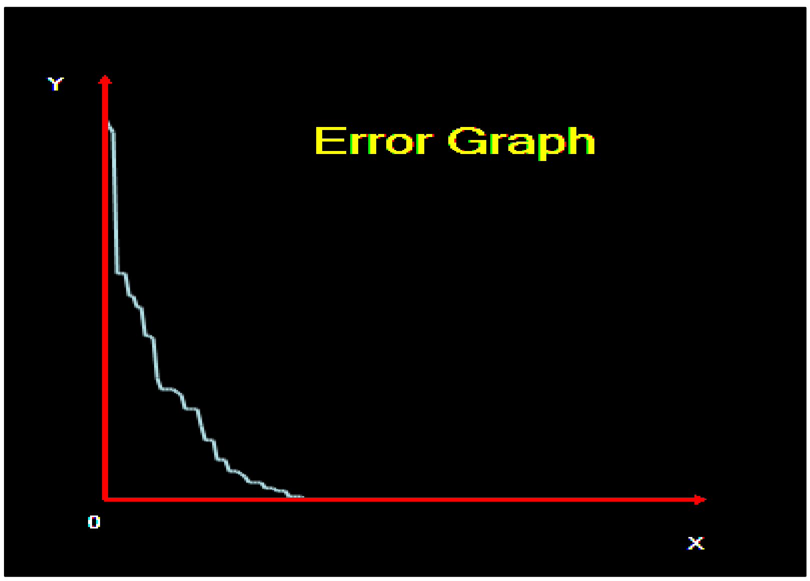


Figure 19: Error graph of NN implemented in goal finding application

Evaluation and Test of System Implementation

The goal finding application was built with functional independence relying on modularity and methods of abstraction and hiding of information. This was achieved by developing components with single focus function and avoiding many interactions with other modules. The focus was on designing components that met a specific requirement and had a simple interface when seen from other parts of the program structure. For example Cosmos component below focused on the specific requirement of modeling the environment in which the agent would exist and interfaced with the main form, network and agent components. This allowed an easier development of the application because of simplified interfaces and reduced difficulty in maintaining or testing because errors were reduced and components were reused across the application. It was a good design and important to developing quality applications. Independence was determined using two qualitative criteria which were cohesion and coupling. The application aimed for high cohesion between components such as Network and Neuron below and low coupling among components such as Cosmos and Steering Behaviors. Black box testing was done to examine the functions that were in the application without focusing on how the code was implemented. The black box testing involved running the simulation to unit test all the test cases matching the functionality shown below in the diagram below. White box testing was performed to test the functionality shown below in the diagram

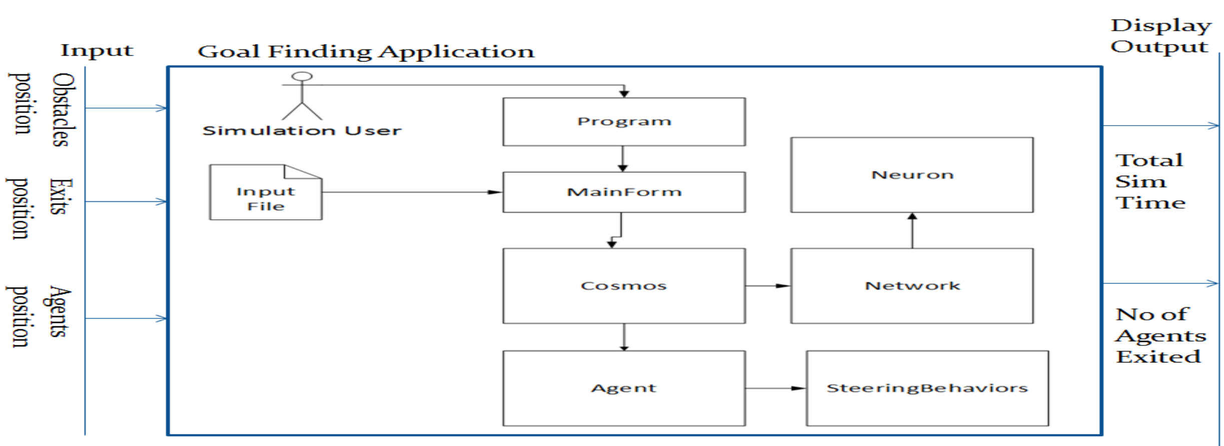


Figure 20: Goal Finding Testing

Timeline

Step	*Activity*	*Time to Commence*	*Deadline*
1	*Proposal hearings on Chapters 1-3*	*February 2014*	*April 2014*
2	*Submission of IRB packets to IRB*	*April 2014*	*May 2014*
3	*Approval received from IRB.*	*May 2014*	*June 2014*
4	*Recruiting of Evacuation drill participants*	*June 2014*	*October 2014*
5	*Neural Network and Genetic Algorithm implementation*	*June 2014*	*October 2014*
6	*Fuzzy Logic Implementation*	*June 2014*	*October 2014*
7	*Evacuation Drill Study in Room 109 Collection and analysis of data*	*November 2014*	*December 2014*
8	*Scheduling of Defense Hearing*	*September 2015*	*Second week September 2015*
9	*Defense Hearing*	*October 2015*	*Last week October 2015*
10	*Completion of all dissertation requirements for December Graduation*	*December 2015*	*December 2015*
11	*Graduation*		*December 2015*

Pilot Research and Preliminary Results

For preliminary validation of our system, we did a set of small, proof-of concept style experiments modeling an actual room used as a computer lab in our department building.

Setup

The intelligent agent application for emergency evacuation was written in Microsoft C-Sharp. It served a similar purpose as Pathfinder 2014 developed by Thunderhead Engineering for agent based egress modeling. The intelligent and Pathfinder applications both had a graphical user interface for simulation and design but intelligent was capable of 2D visualization for result analysis while Pathfinder had 2D and 3D (Figures 16 and 17). Output of the intelligent application was shown to the right in Figure 16 below as generation and current best. The length of a generation was 90 seconds after which the agents were counted that had successfully evacuated and were recorded in the current best. The movement environment in the intelligent application was a 2D rectangle (Figure 16) representing the dimensions of the room. Dimensions of the room were represented on the screen with a rectangle width 1015 and height 695 pixels. The total number of people modeled in the test simulation was about 45.

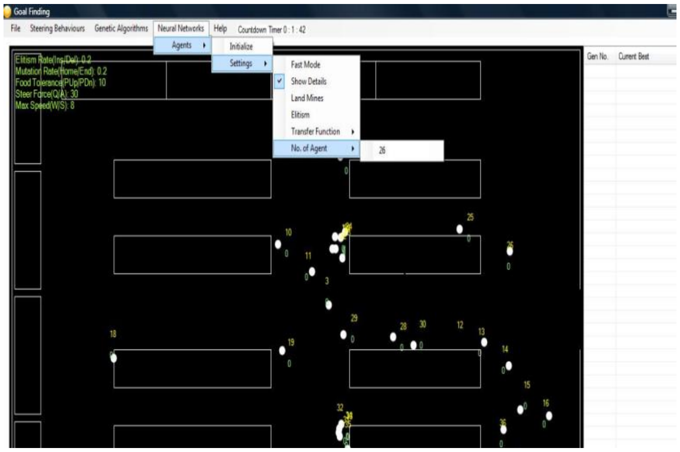
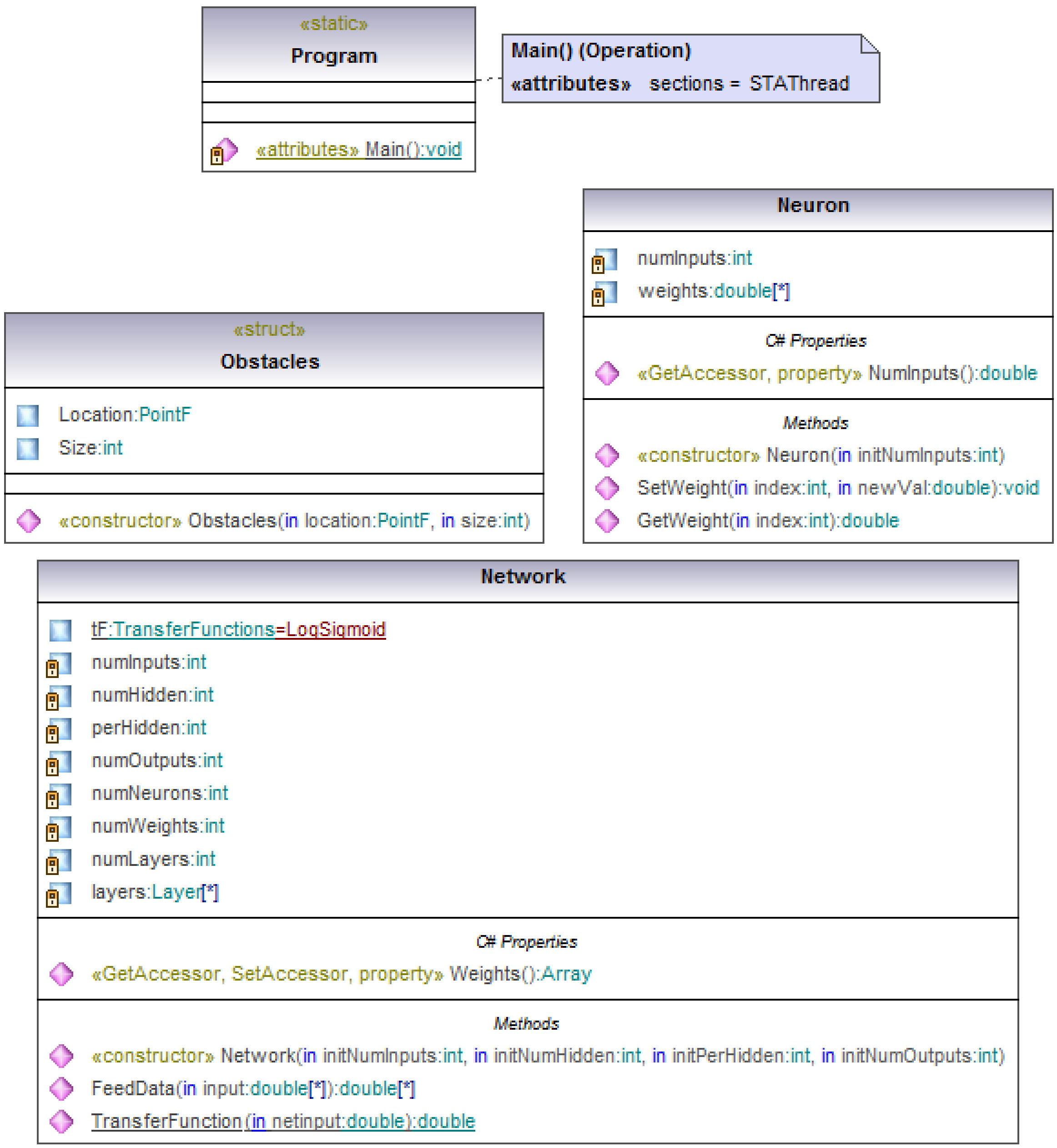
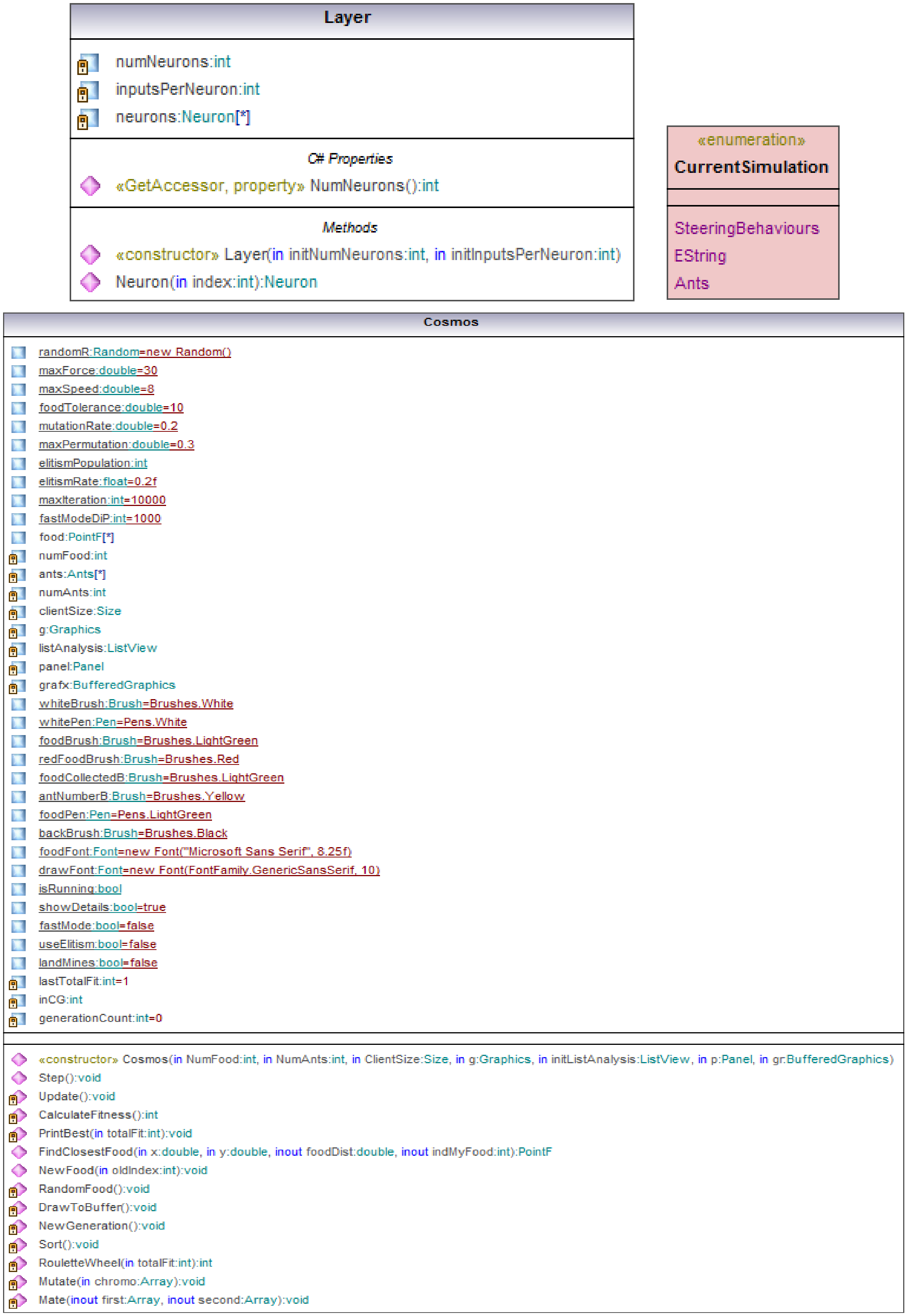


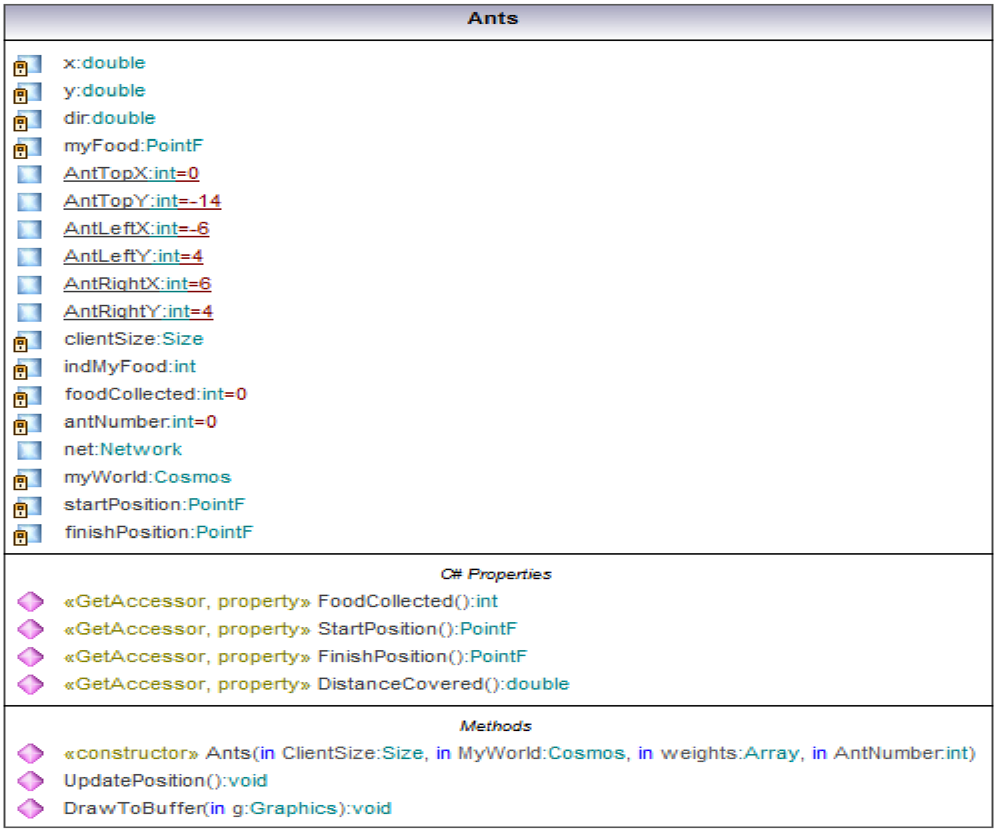
Figure 21: Intelligent Agent Graphical User Interfaces

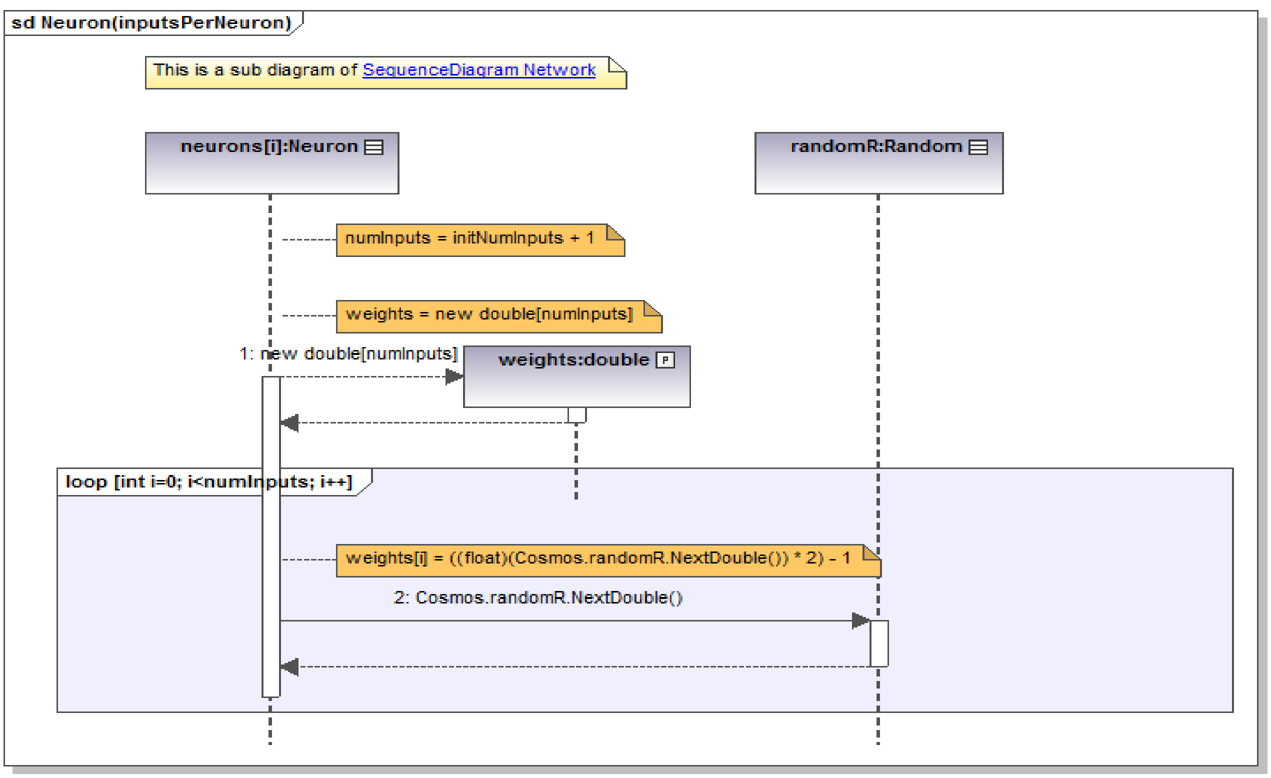
Walls and other obstacles in the intelligent application were shown as rectangles that could not be passed through by the agents. Doors were represented as gaps in the wall. The parameters for drawing the walls were passed to the intelligent simulation through an input text file specified by the user of the application. The UML diagrams of the intelligent application were shown below that represented the program code.

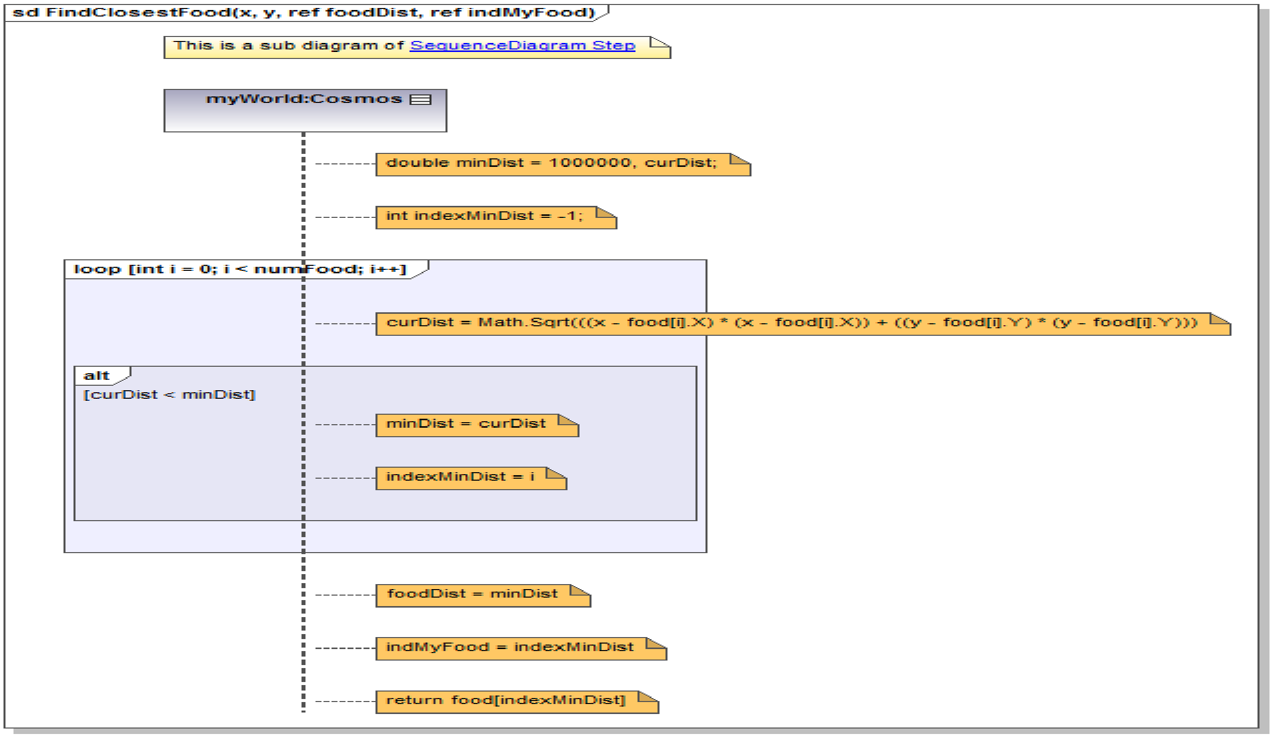
UML Diagrams

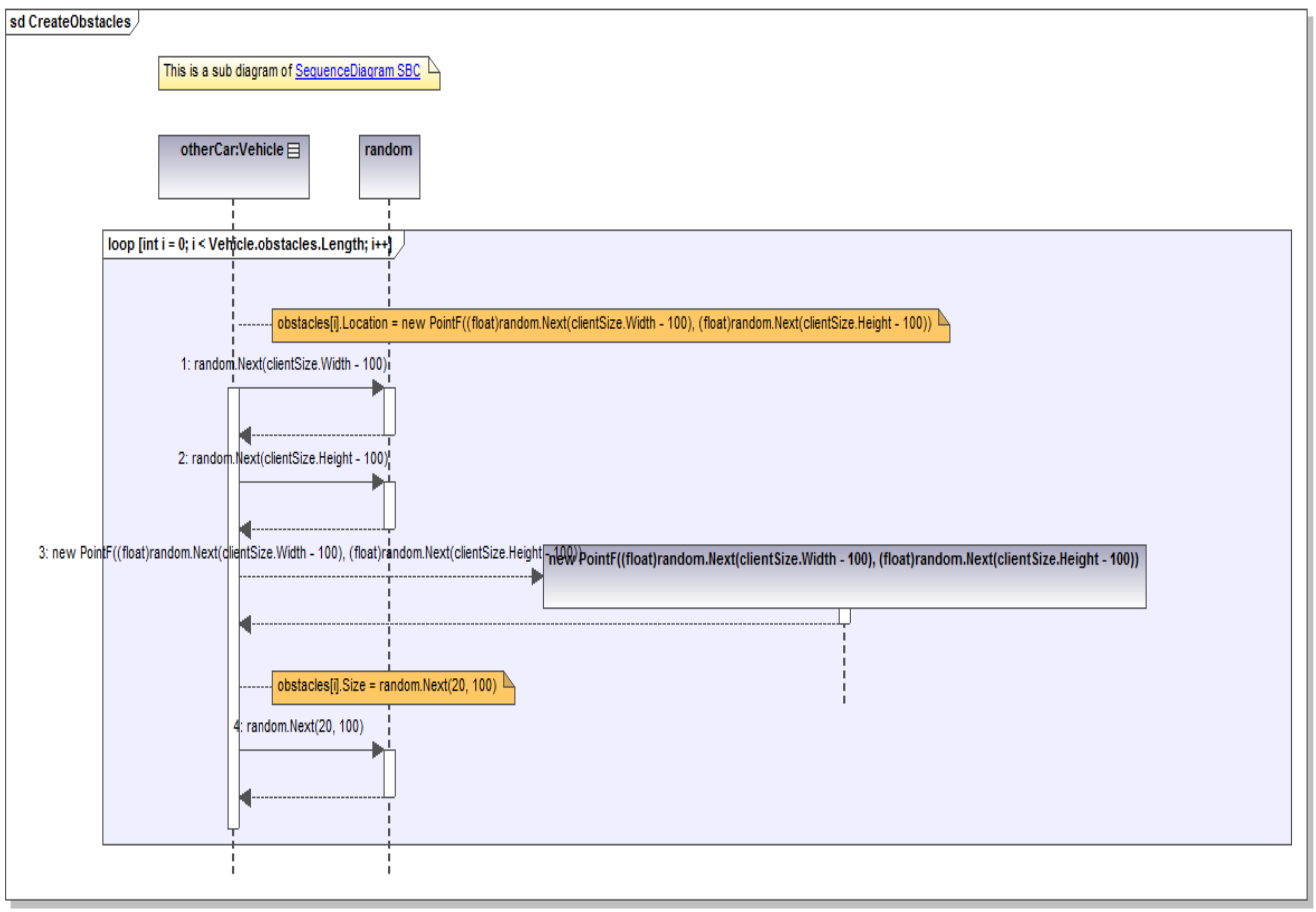


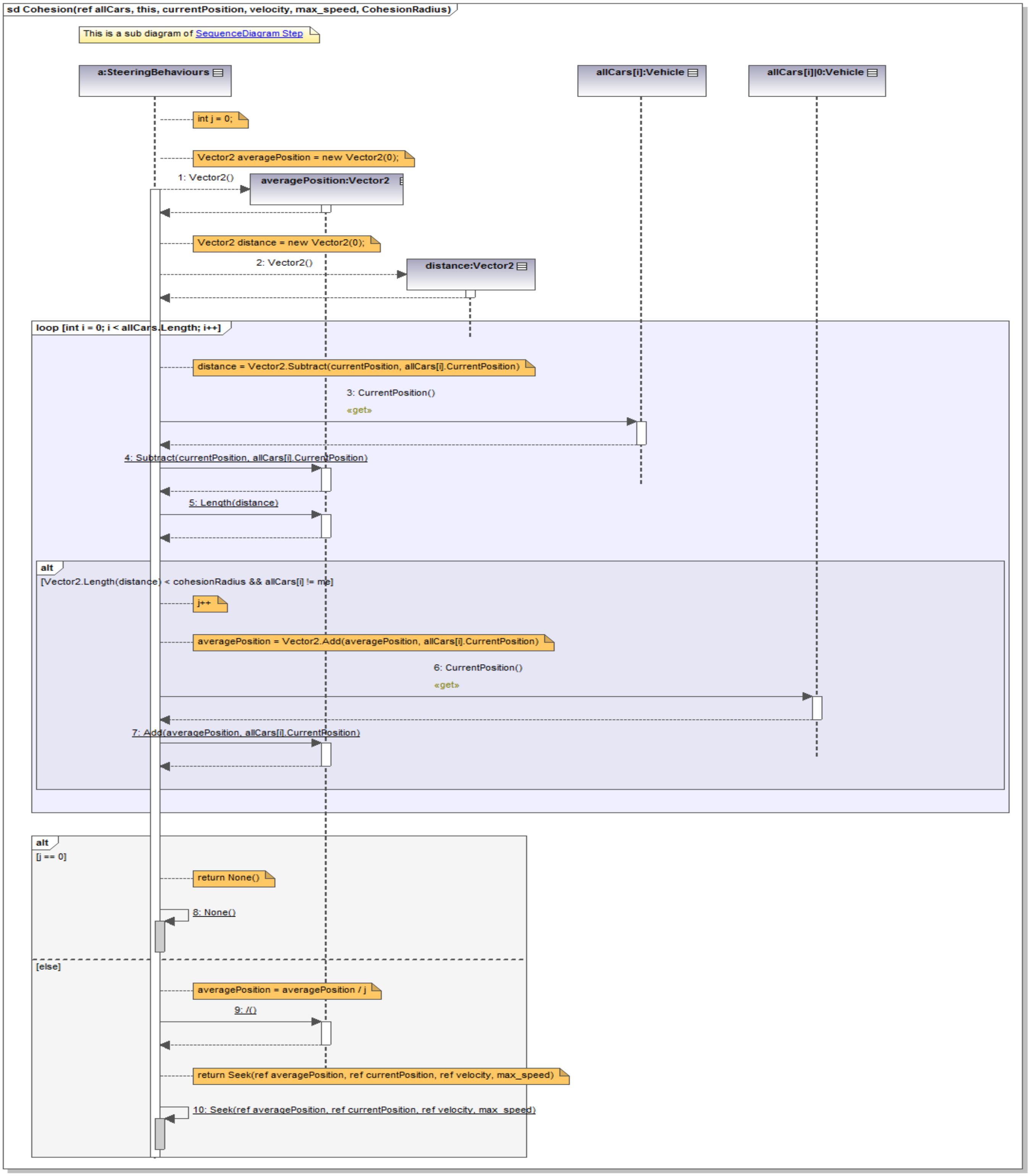

Layer
numNeurons:int
inputsPerNeuron:int
neurons:Neuron[*]
C# Properties
«GetAccessor, property» NumNeurons():int
Methods
«constructor» Layer(in initNumNeurons:int, in initInputsPerNeuron:int)
Neuron(in index:int):Neuron
«enumeration»
CurrentSimulation
SteeringBehaviours
EString
Ants
Cosmos
randomR:Random=new Random()
maxForce:double=30
maxSpeed:double=8
foodTolerance:double=10
mutationRate:double=0.2
maxPermutation:double=0.3
elitismPopulation:int
elitismRate:float=0.2f
maxIteration:int=10000
fastModeDiP:int=1000
food:PointF[*]
numFood:int
ants:Ants[*]
numAnts:int
clientSize:Size
g:Graphics
listAnalysis:ListView
panel:Panel
grafx:BufferedGraphics
whiteBrush:Brush=Brushes.White
whitePen:Pen=Pens.White
foodBrush:Brush=Brushes.LightGreen
redFoodBrush:Brush=Brushes.Red
foodCollectedB:Brush=Brushes.LightGreen
antNumberB:Brush=Brushes.Yellow
foodPen:Pen=Pens.LightGreen
backBrush:Brush=Brushes.Black
foodFont:Font=new Font("Microsoft Sans Serif", 8.25f)
drawFont:Font=new Font(FontFamily.GenericSansSerif, 10)
isRunning:bool
showDetails:bool=true
fastMode:bool=false
useElitism:bool=false
landMines:bool=false
lastTotalFit:int=1
inCG:int
generationCount:int=0
«constructor» Cosmos(in NumFood:int, in NumAnts:int, in ClientSize:Size, in g:Graphics, in initListAnalysis:ListView, in p:Panel, in gr:BufferedGraphics)
Step():void
Update():void
CalculateFitness():int
PrintBest(in totalFit:int):void
FindClosestFood(in x:double, in y:double, inout foodDist:double, inout indMyFood:int):PointF
NewFood(in oldIndex:int):void
RandomFood():void
DrawToBuffer():void
NewGeneration():void
Sort():void
RouletteWheel(in totalFit:int):int
Mutate(in chromo:Array):void
Mate(inout first:Array, inout second:Array):void

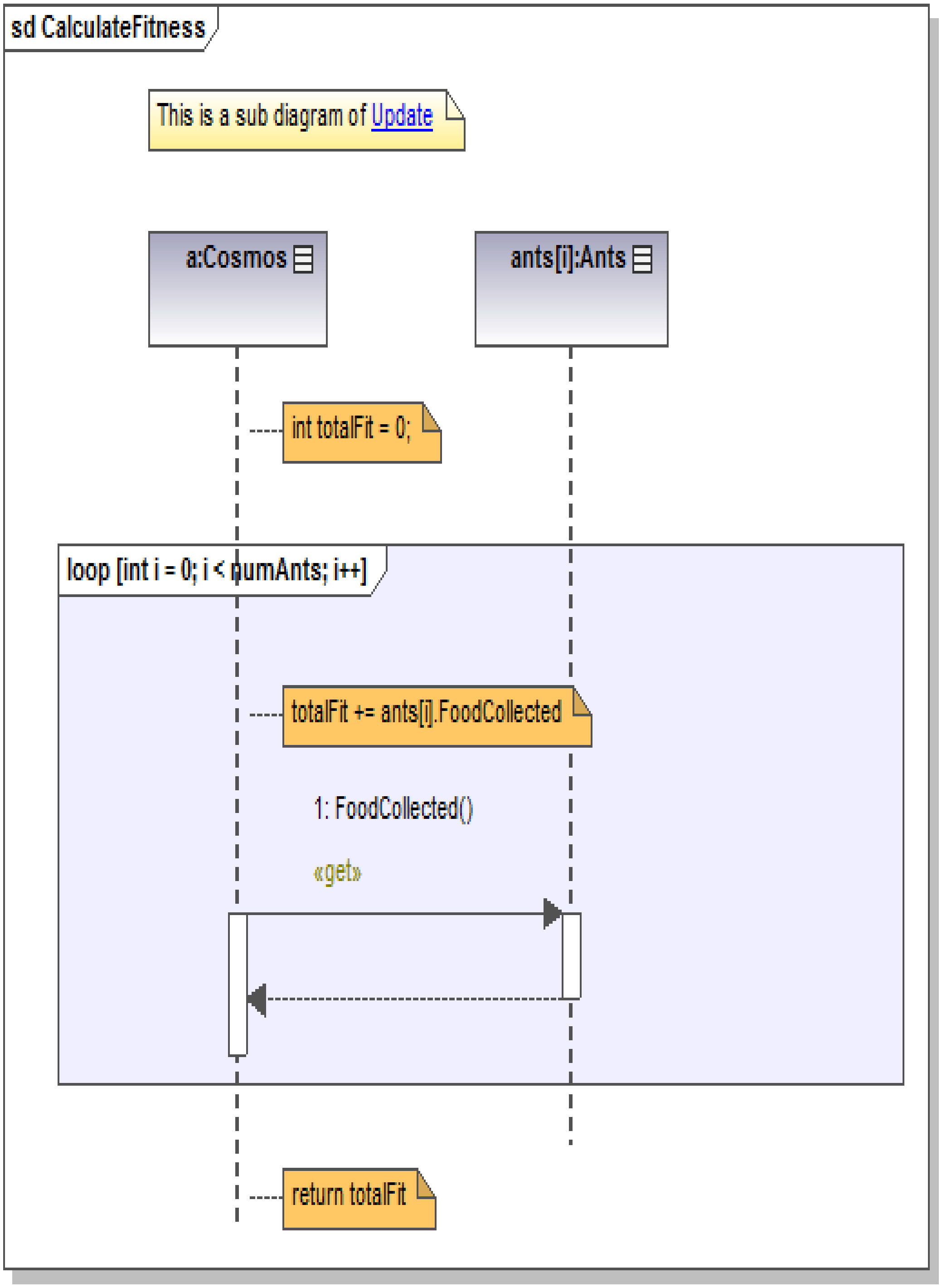

Ants
x:double
y:double
dir:double
myFood:PointF
AntTopX:int=0
AntTopY:int=-14
AntLeftX:int=-6
AntLeftY:int=4
AntRightX:int=6
AntRightY:int=4
clientSize:Size
indMyFood:int
foodCollected:int=0
antNumber:int=0
net:Network
myWorld:Cosmos
startPosition:PointF
finishPosition:PointF
C# Properties
«GetAccessor, property» FoodCollected():int
«GetAccessor, property» StartPosition():PointF
«GetAccessor, property» FinishPosition():PointF
«GetAccessor, property» DistanceCovered():double
Methods
«constructor» Ants(in ClientSize:Size, in MyWorld:Cosmos, in weights:Array, in AntNumber:int)
UpdatePosition():void
DrawToBuffer(in g:Graphics):void

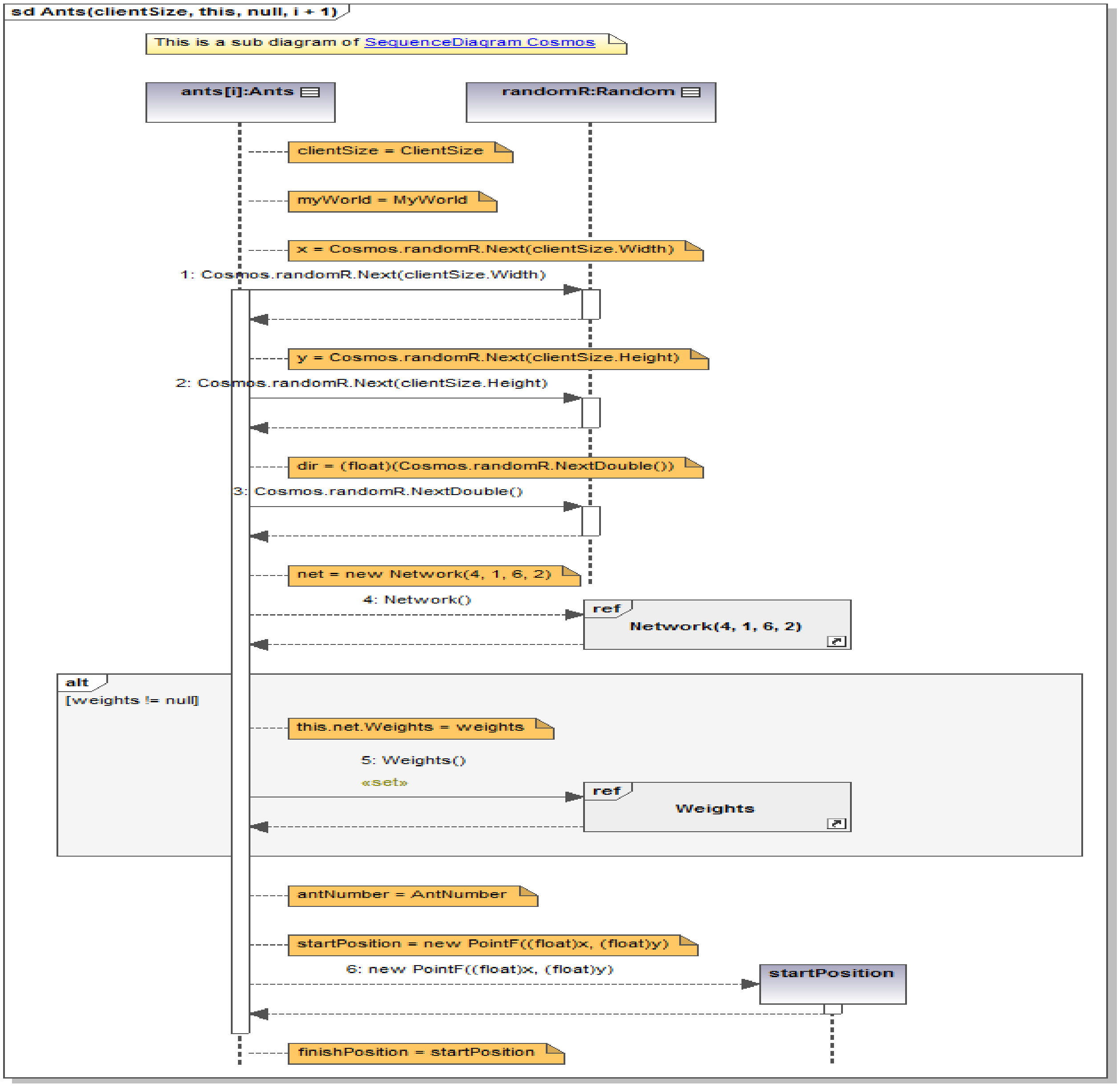

sd Neuron(inputsPerNeuron)
This is a sub diagram of SequenceDiagram Network
neurons[i]:Neuron
randomR:Random
numInputs = initNumInputs + 1
weights = new double[numInputs]
1: new double[numInputs]
weights:double
loop [int i=0; i<numInputs; i++]
weights[i] = ((float)(Cosmos.randomR.NextDouble()) * 2) - 1
2: Cosmos.randomR.NextDouble()

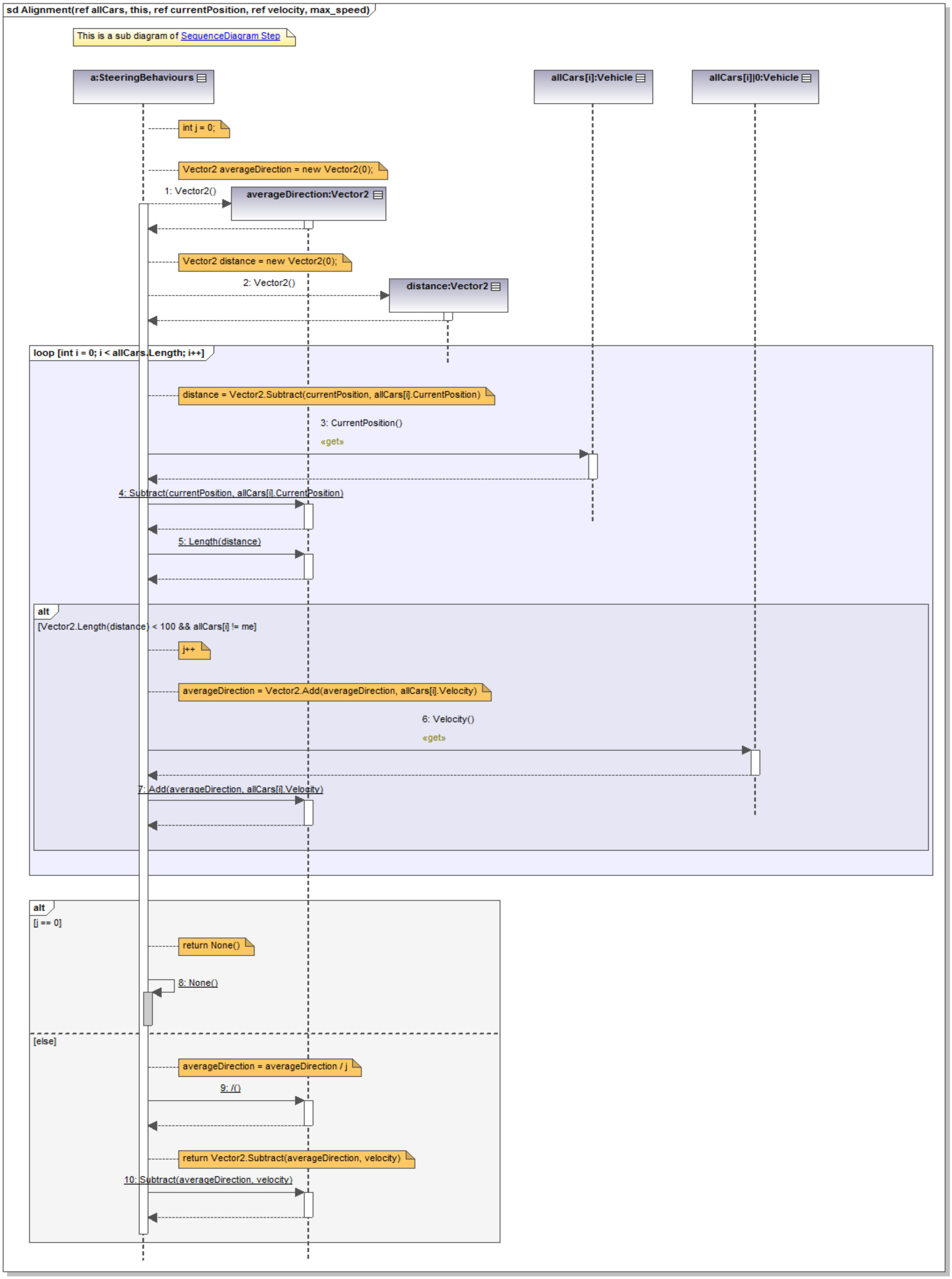

sd FindClosestFood(x, y, ref foodDist, ref indMyFood)
This is a sub diagram of SequenceDiagram Step
myWorld:Cosmos
double minDist = 1000000, curDist;
int indexMinDist = -1;
loop [int i = 0; i < numFood; i++]
curDist = Math.Sqrt(((x - food[i].X) * (x - food[i].X)) + ((y - food[i].Y) * (y - food[i].Y)))
alt
[curDist < minDist]
minDist = curDist
indexMinDist = i
foodDist = minDist
indMyFood = indexMinDist
return food[indexMinDist]

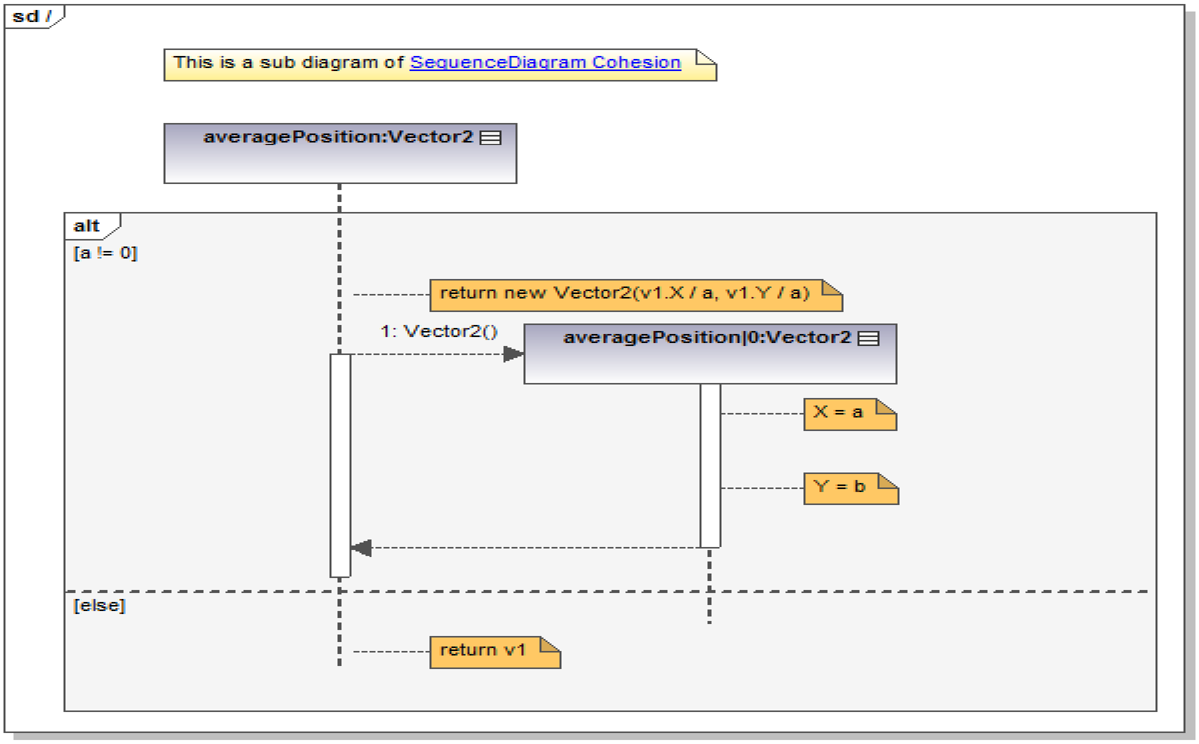

sd CreateObstacles
This is a sub diagram of SequenceDiagram SBC
otherCar:Vehicle
random
loop [int i = 0; i < Vehicle.obstacles.Length; i++]
obstacles[i].Location = new PointF((float)random.Next(clientSize.Width - 100), (float)random.Next(clientSize.Height - 100))
1: random.Next(clientSize.Width - 100)
2: random.Next(clientSize.Height - 100)
3: new PointF((float)random.Next(clientSize.Width - 100), (float)random.Next(clientSize.Height - 100))
new PointF((float)random.Next(clientSize.Width - 100), (float)random.Next(clientSize.Height - 100))
obstacles[i].Size = random.Next(20, 100)
4: random.Next(20, 100)

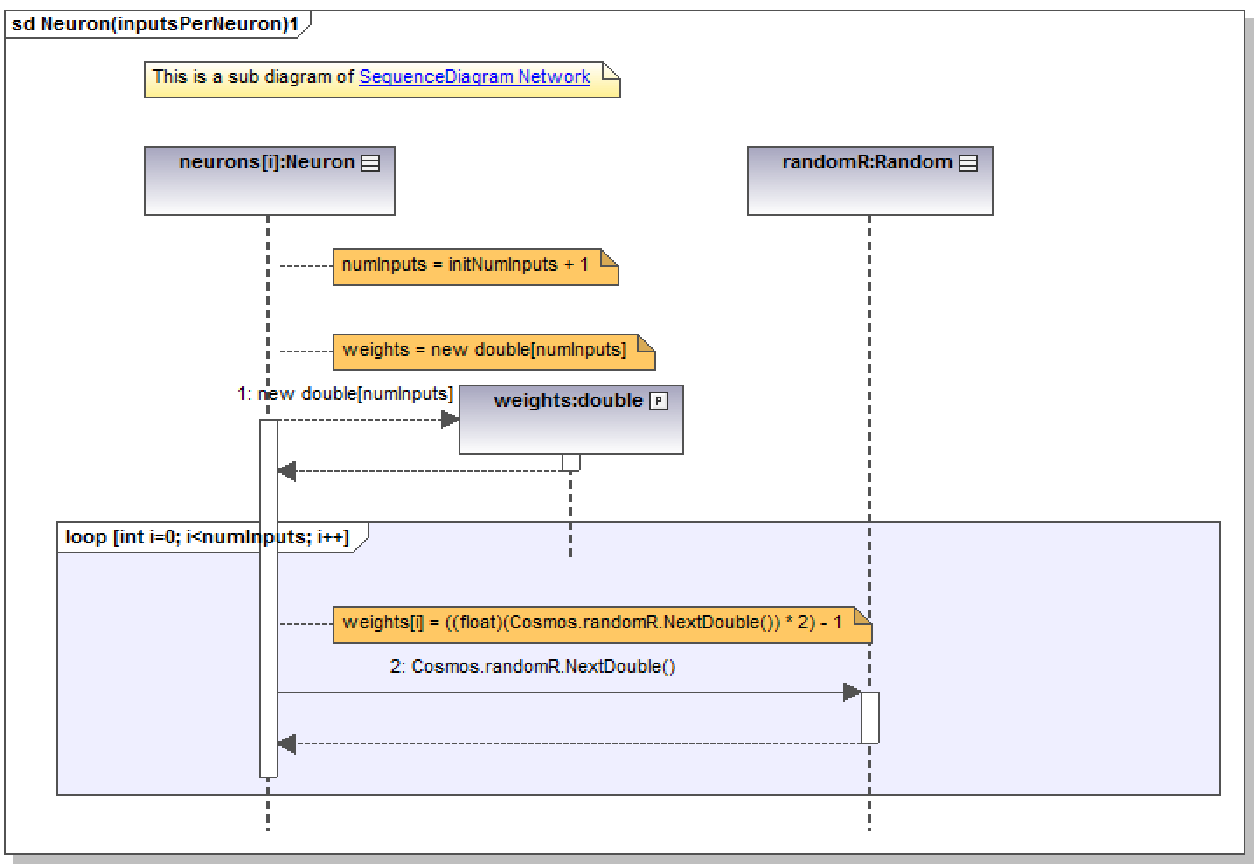

sd Cohesion(ref allCars, this, currentPosition, velocity, max_speed, CohesionRadius)
This is a sub diagram of SequenceDiagram Step
a:SteeringBehaviours
allCars[i]:Vehicle
allCars[i]|0:Vehicle
int j = 0;
Vector2 averagePosition = new Vector2(0);
1: Vector2()
averagePosition:Vector2
Vector2 distance = new Vector2(0);
2: Vector2()
distance:Vector2
loop [int i = 0; i < allCars.Length; i++]
distance = Vector2.Subtract(currentPosition, allCars[i].CurrentPosition)
3: CurrentPosition()
«get»
4: Subtract(currentPosition, allCars[i].CurrentPosition)
5: Length(distance)
alt
[Vector2.Length(distance) < cohesionRadius && allCars[i] != me]
j++
averagePosition = Vector2.Add(averagePosition, allCars[i].CurrentPosition)
6: CurrentPosition()
«get»
7: Add(averagePosition, allCars[i].CurrentPosition)
alt
[j == 0]
return None()
8: None()
[else]
averagePosition = averagePosition / j
9: /()
return Seek(ref averagePosition, ref currentPosition, ref velocity, max_speed)
10: Seek(ref averagePosition, ref currentPosition, ref velocity, max_speed)


sd CalculateFitness
This is a sub diagram of Update
a:Cosmos
ants[i]:Ants
int totalFit = 0;
loop [int i = 0; i < numAnts; i++]
totalFit += ants[i].FoodCollected
1: FoodCollected()
«get»
return totalFit


sd Ants(clientSize, this, null, i + 1)
This is a sub diagram of SequenceDiagram Cosmos
ants[i]:Ants
randomR:Random
clientSize = ClientSize
myWorld = MyWorld
x = Cosmos.randomR.Next(clientSize.Width)
1: Cosmos.randomR.Next(clientSize.Width)
y = Cosmos.randomR.Next(clientSize.Height)
2: Cosmos.randomR.Next(clientSize.Height)
dir = (float)(Cosmos.randomR.NextDouble())
3: Cosmos.randomR.NextDouble()
net = new Network(4, 1, 6, 2)
4: Network()
ref
Network(4, 1, 6, 2)
alt
[weights != null]
this.net.Weights = weights
5: Weights()
«set»
ref
Weights
antNumber = AntNumber
startPosition = new PointF((float)x, (float)y)
6: new PointF((float)x, (float)y)
startPosition
finishPosition = startPosition


sd Alignment(ref allCars, this, ref currentPosition, ref velocity, max_speed)
This is a sub diagram of SequenceDiagram Step
a:SteeringBehaviours
allCars[i]:Vehicle
allCars[i]|0:Vehicle
int j = 0;
Vector2 averageDirection = new Vector2(0);
1: Vector2()
averageDirection:Vector2
Vector2 distance = new Vector2(0);
2: Vector2()
distance:Vector2
loop [int i = 0; i < allCars.Length; i++]
distance = Vector2.Subtract(currentPosition, allCars[i].CurrentPosition)
3: CurrentPosition()
«get»
4: Subtract(currentPosition, allCars[i].CurrentPosition)
5: Length(distance)
alt
[Vector2.Length(distance) < 100 && allCars[i] != me]
j++
averageDirection = Vector2.Add(averageDirection, allCars[i].Velocity)
6: Velocity()
«get»
7: Add(averageDirection, allCars[i].Velocity)
alt
[j == 0]
return None()
8: None()
[else]
averageDirection = averageDirection / j
9: /()
return Vector2.Subtract(averageDirection, velocity)
10: Subtract(averageDirection, velocity)


sd /
This is a sub diagram of SequenceDiagram Cohesion
averagePosition:Vector2
alt
[a != 0]
return new Vector2(v1.X / a, v1.Y / a)
1: Vector2()
averagePosition|0:Vector2
X = a
Y = b
[else]
return v1


sd Neuron(inputsPerNeuron)1
This is a sub diagram of SequenceDiagram Network
neurons[i]:Neuron
randomR:Random
numInputs = initNumInputs + 1
weights = new double[numInputs]
1: new double[numInputs]
weights:double
loop [int i=0; i<numInputs; i++]
weights[i] = ((float)(Cosmos.randomR.NextDouble()) * 2) - 1
2: Cosmos.randomR.NextDouble()

Pathfinder represented the movement environment with a surrounding mesh resembling the building. Gaps in the surrounding mesh displayed the walls and areas that could not be passed by the occupants. Doors were represented by unique edges in the navigation mesh. Stairways and elevators were represented in the pathfinder application.

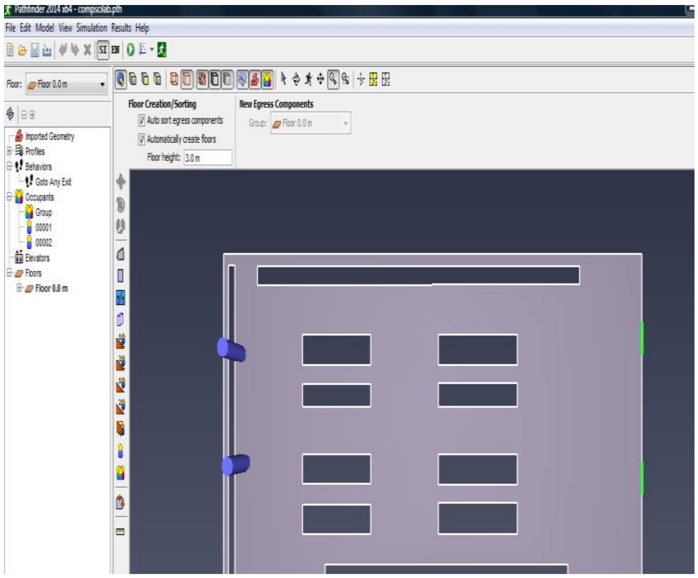


Figure 22: Pathfinder Graphical User Interface

Simulation and Results

Validation was done by comparing the evacuation time when the simulation was completed on the intelligent agent application and pathfinder application. Agents in the intelligent application were drawn as circles in the simulation while occupants in pathfinder were shown as upright cylinders on the movement mesh. The position and speed of both agents and occupants in intelligent and pathfinder were both specified in the simulation applications. Intelligent application had one simulation mode where agents steered themselves so they did not pass through other agents. Pathfinder had steer mode and also the SFPE mode where occupants were allowed to go through other occupants.

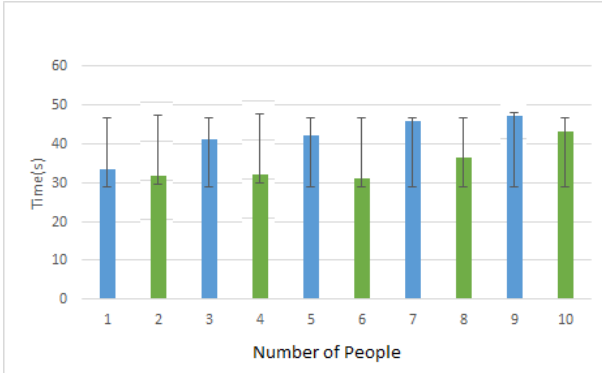


Figure 23: Evacuation Time Error Bar comparison of intelligent and pathfinder applications

Our implemented C# evacuation application and the pathfinder application both had similar functionality and performance that allowed them to be effective tools for data visualization in emergency evacuation. Figure 23 showed the error bar of the comparison seen when running the intelligent agent application (blue bar) and the pathfinder application (green bar). The simulation involved 25, 35, 45, 65, 75 occupants similarly placed in positions in the room and timed to see how fast they could evacuate the room. Bar graphs 1 and 2 represented the simulation running with 25 people in the evacuation and path finder applications respectively. The average time was 33 seconds for evacuation application and 25 seconds for path finder application. Bar graphs 3 and 4 represented the simulation running with 35 people in the evacuation and path finder applications respectively. The average time was 41 seconds for evacuation application and 27 seconds for path finder application. Bar graphs 5 and 6 represented the simulation running with 45 people in the evacuation and path finder applications respectively. The average time was 42 seconds for evacuation application and 31 seconds for path finder application. Bar graphs 7 and 8 represented the simulation running with 65 people in the evacuation and path finder applications respectively. The average time was 46 seconds for evacuation application and 36.5 seconds for path finder application. Bar graphs 9 and 10 represented the simulation running with 75 people in the evacuation and path finder applications respectively. The average time was 53 seconds for evacuation application and 43 seconds for path finder application.

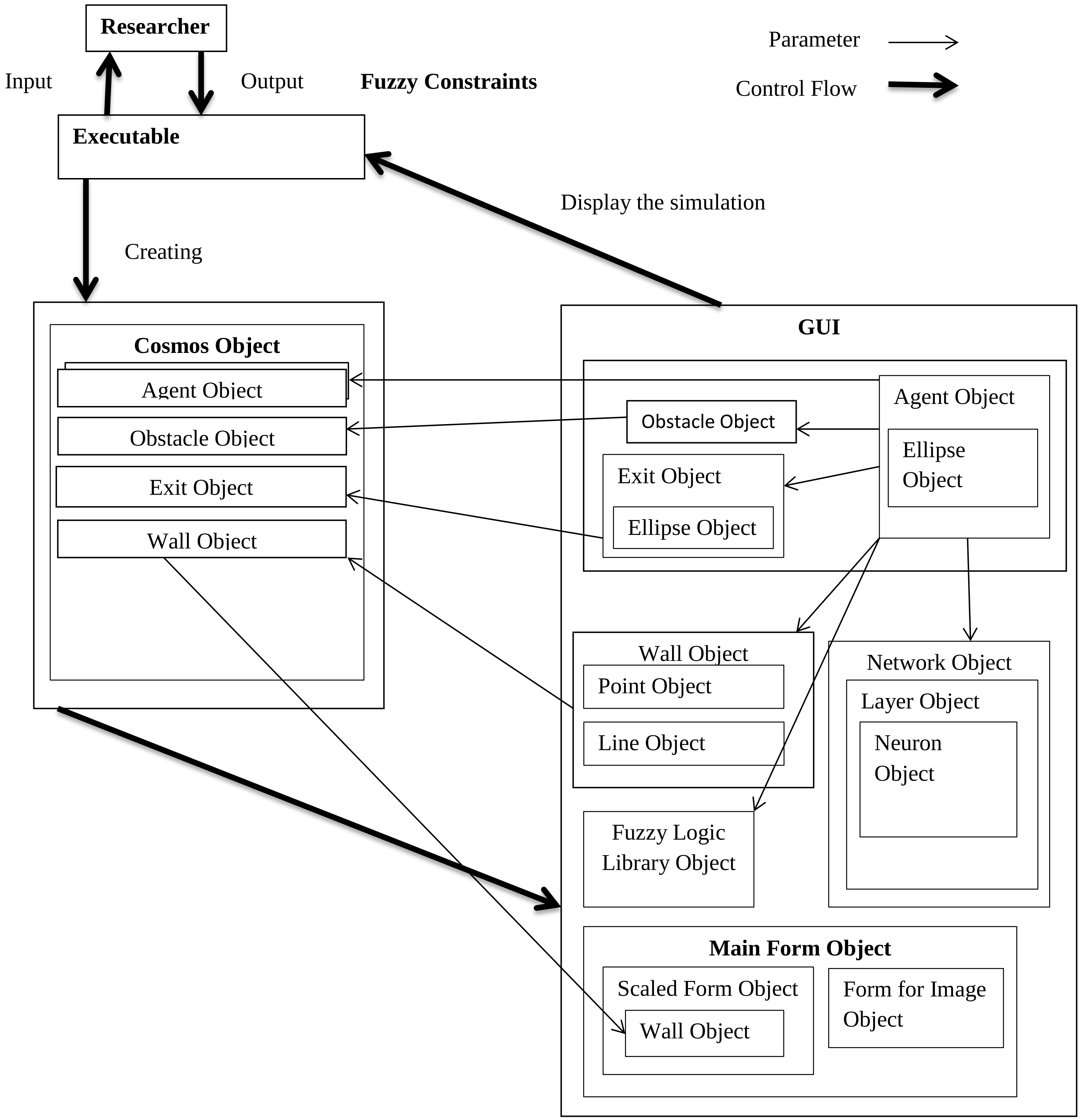
Our proposed C-Sharp application evacuation time ranged from 33 and 53 seconds and was pretty close to the measured evacuation time of 25 and 43 seconds in pathfinder application. As more people were added, the differences in evacuation time between the applications were magnified in the simulation. For modeling emergency evacuation there was data needed for how people behave in various events such as panic, stress, fire, smoke, chaos, explosions, and human uncertainty. The intelligent agents in our implemented C-Sharp application were able to include modeling human emotional behavior that was important for decision-making strategies. We expected this technique to be very useful for large-scale evacuations, such as a multi-story office building or a stadium. We were able find the evacuation time people left the exit, with a good reliability. We hope our implemented tool will aid in visualizing evacuation time and what-if scenarios by incorporating data on human emotions and movements. Thus, the implemented C-Sharp application could be used to model situations that are difficult to test in real-life due to safety considerations.

CHAPTER 4

EXPERIMENTS, RESULTS AND DISCUSSIONS

System Architecture

The architecture of the Intelligent Goal Finding C- Sharp Application is broken up into four main sections: *Initialization*, *Construction*, *Simulation*, and *Termination*. When the executable is launched in a desktop window, it spontaneously forms the entities of C-Sharp Main Form class. This is triggered by the Neural Networks => Agents => Initialize tab of the application. The input file comprises behavior features of each agent. The behavior features include number and position of agents, number and position of obstacles, number and position of exits, speed, and level of stress and panic of agents. The behavior features can be applied to a collection of individuals through the input file. The Cosmos object represents the simulation instrument and it contains all the entities (refer to Figure 24) for generating the simulation environment. In order to create the simulation environment, the researcher must use an input file to define the environment. The environment comprises the demarcation of walls, exits, and agents and other necessary considerations such as speed, levels of stress and panic (refer to Figure 25) for simulation. The construction section begins when the executable is launched. The cosmos object contains agent object, wall object, exit object, and obstacle object. The application displays the executable in the window's screen. The simulation section begins when the researcher presses the "initialize" button. The executable re-draws the environment after every one sec. Finally, the researcher can view the presented display and simulation constraints on the executable. This phase is called the Termination phase. The bold black arrows in Figure 24 display the switch movement between objects and the light black arrows display the connected relation between objects.



<u>Figure 24</u>: The System architecture diagram for C-Sharp Goal Finding Application.

The fuzzy constraints are demarcated in the input file which behaves as a form of passing messages between the researcher and the executable. The simulation tool can be displayed as a

C-Sharp executable- a windows desktop program that runs in a form that has a windows operating system.

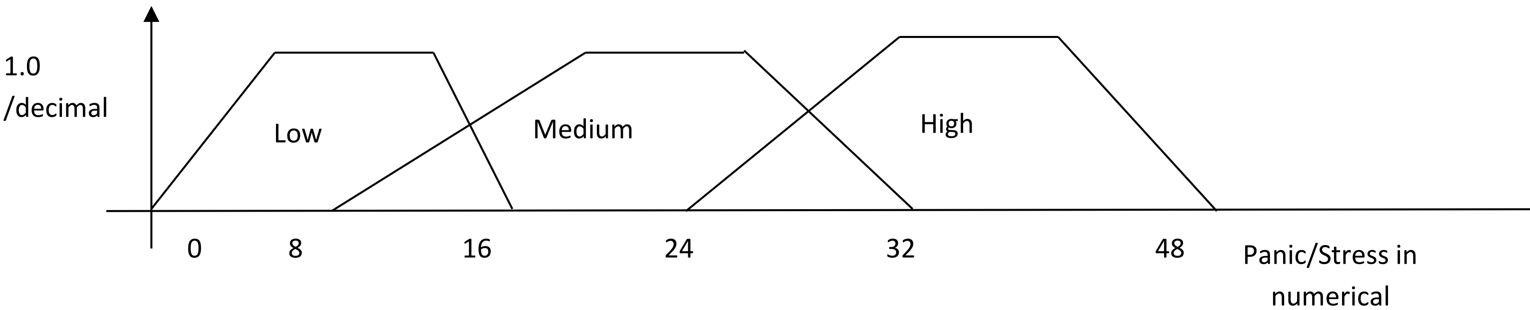


Figure 25: Membership Function of Level of Stress and Panic.

The application uses concurrent threads to track the tool. For example the simulation threads used are timer thread, statistics threads, exit thread, and simulation running thread. The researcher can use the start, stop and pause buttons to relate with the simulation while it is running. The simulation thread also informs several modules and constraints such as: when agent has exited, agent collision with static and dynamic obstacles, simulation time, etc. The form object displays the environment Cosmos object which implements the simulation thread in the executable.

The connection between the above classes in intelligent goal finding application has been detailed in Figure 24. The simulation starts when the initialize button in the intelligent goal finding application is clicked and can be paused and resumed by the click of Pause and Start buttons. If the simulation is restarted the following parameters are updated:

- Number of agents exited is initialized to zero.
- Speed of agents is initialized using parameters in input file.

- Stress and panic of agents is initialized using parameters in input file.
- Number of obstacles is initialized using parameters in input file.
- Number of exits is initialized using parameters in input file.
- The agents are placed inside the environment using coordinates in input file.
- Main form is refreshed.
- Simulation time is initialized to zero.

During the simulation, the redraw process restores the simulation every 1 second. After the simulation is done the researcher has an opportunity to begin another simulation. Figure 26 displays the movement illustration of how an agent relates with the environment, obstacles, and other agents. The Intelligent agent goal application is also based on a collective or group dynamics model working with a linear model. The collective or group dynamics model arises in the representation when the exit is small and agent queue at the exit. Contingent on the agent constraints (speed, wait time) the agents can drive other agents in their route to depart the environment. When the exit is narrower and agents queue at the exit, they start to repel each other. The input parameters of an agent can be changed in the intelligent agent goal application's input file to signify a heavy person (wait time, speed, and mass). The agent has the ability to drive other agents in an evacuation situation.

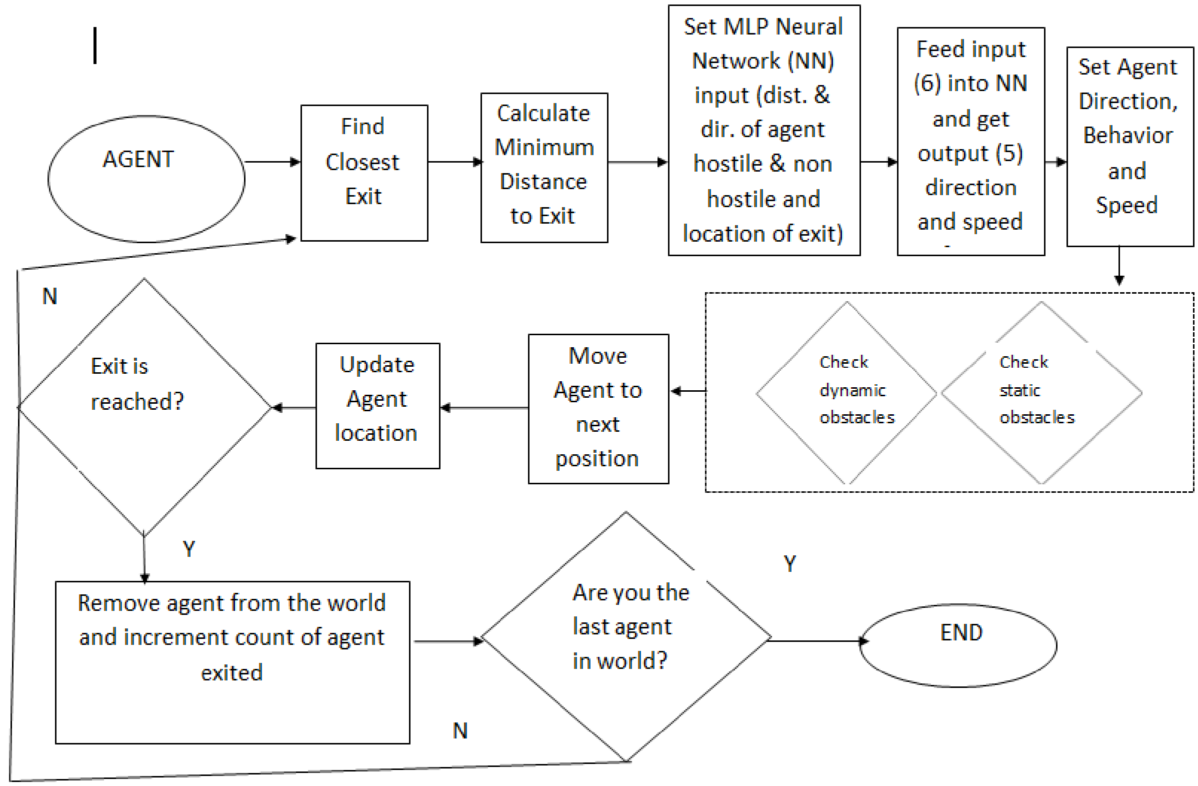


Figure 26: Navigation of agent with environment, obstacles, and other agents.

Flow Chart for Intelligent Goal Finding Application

The Microsoft Visual Studio 2010 Express IDE application calls the main form class to start the simulation. The researchers click the start button to begin the application. MainForm class is the main class where the initialize, and run methods are called to show the form. Figure 27 displays the flowchart of intelligent goal finding application. If the input file has an error, the error message is displayed in windows of the form. After the input file is read by the MainForm class, the subjects (walls, goals, agents) are shown in the executable. The simulation starts when the initialize button is clicked and can be paused and resumed by the click of Pause and Start buttons.

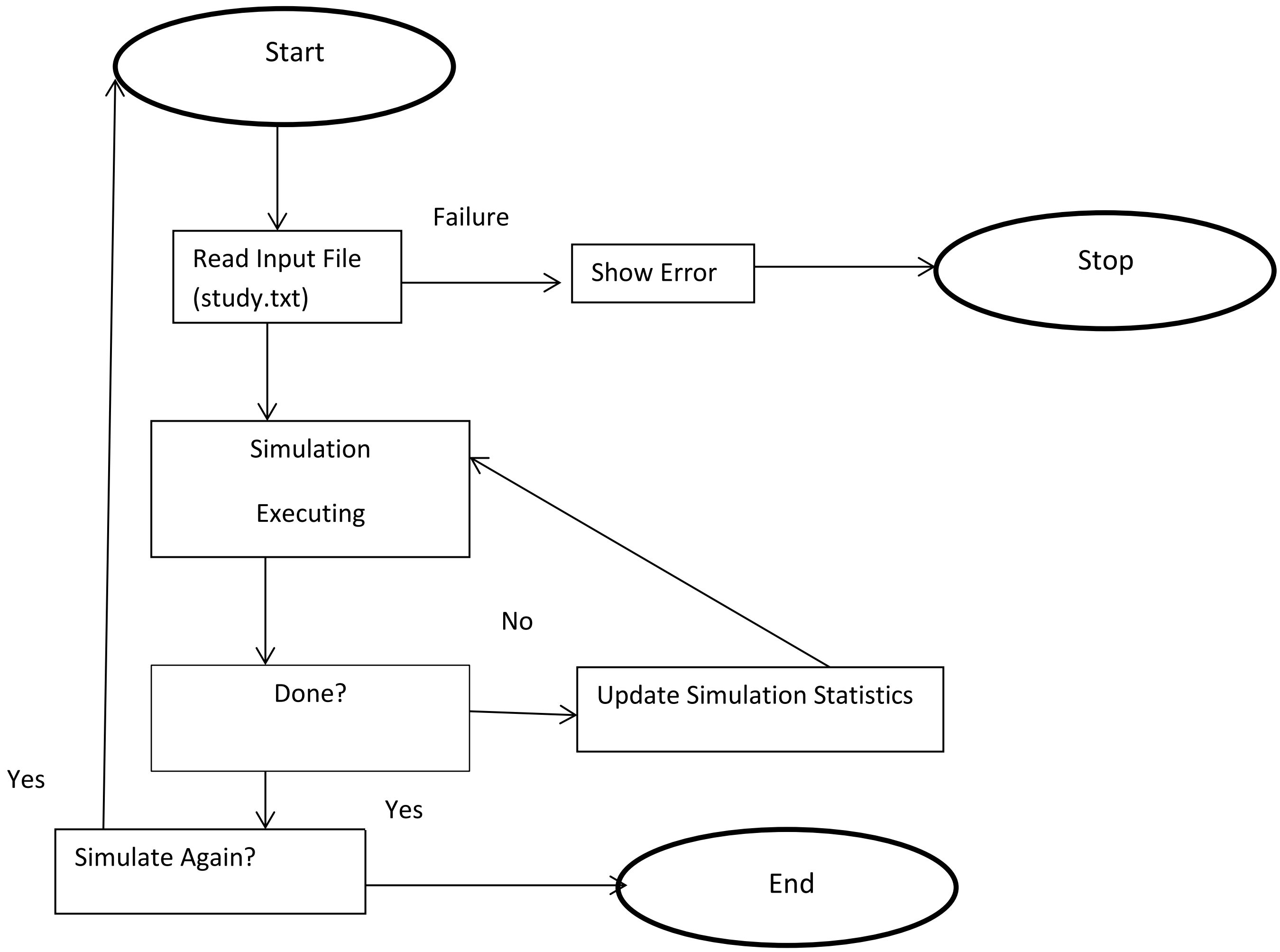


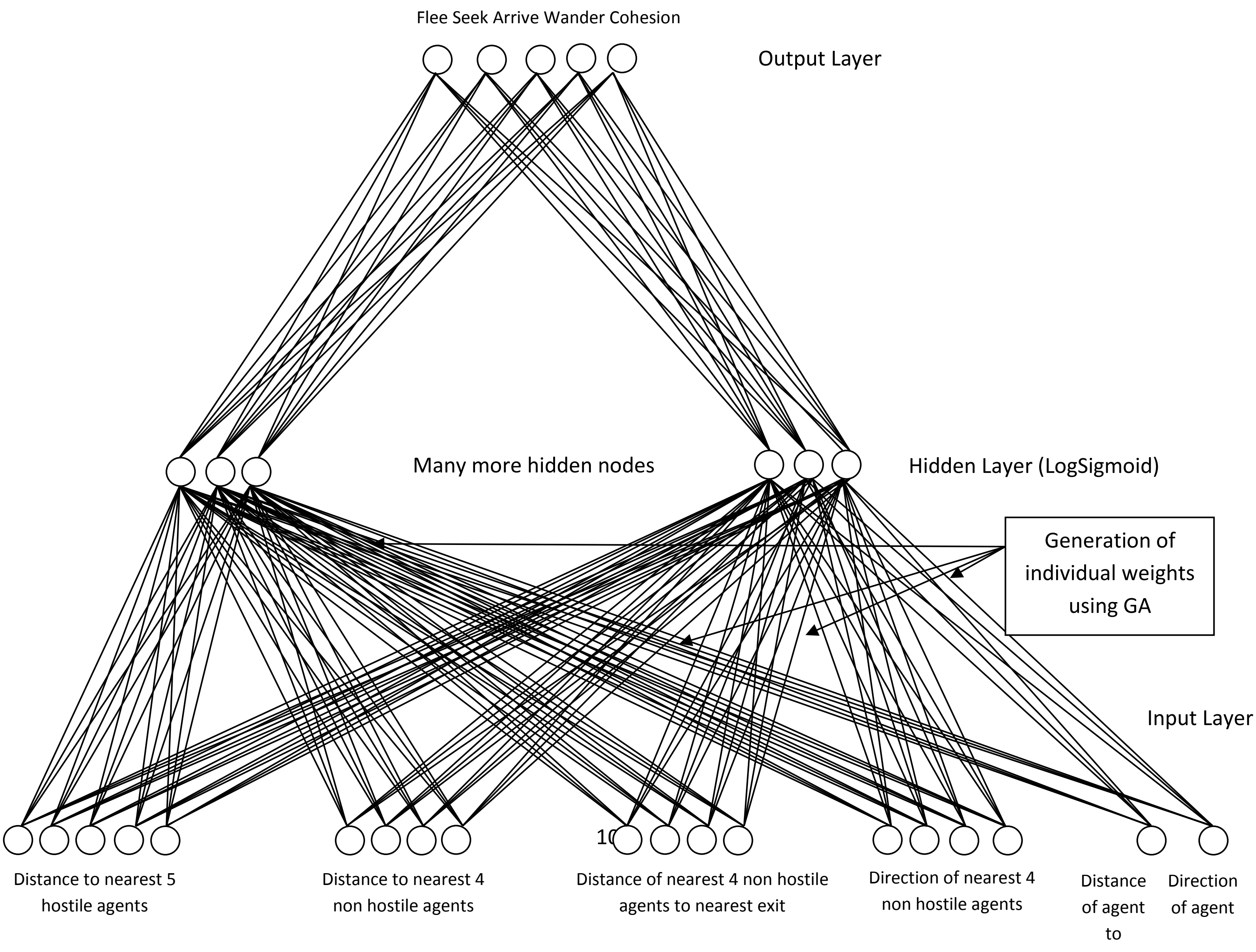
Figure 27: Flow Chart of Simulation.

Algorithm for Intelligent Agent Goal Application implementation

A. Description: An AI-controlled character makes use of 19 input values: the distance to the nearest 5 hostile agents, distance to the nearest 4 non hostile agents with their distance to nearest exit and direction values, and the distance to nearest exit and direction of the AI. We assumed 5 different output behaviors agents can exhibit on their way to an exit: flee, seek, arrive, wander and cohesion. We use a network with three layers: input layer and output layer, plus an intermediate (hidden) layer. The input layer has the same number of nodes 19. The output layer has the same number of nodes as there are possible outputs: 5. Hidden layers were 24.

<u>B. Steps :</u> to begin all the weights in the network are set to small random values and a set of iterations of the learning algorithm is done involving selecting an example scenario from the training set. We then take the inputs feed forward to guess the output and change the network (back propagation) by comparing the expected output and the guess. Every 10 to15 seconds, GA roulette wheel selection process is used to select the two parents that will be used to change the weight of children. After iterations are done, we can check to see if learning is done by running on the test group of examples. If the guess output meets our expected that we are sure NN has learned properly else we can run more training on the network (refer to Figure 28).

<u>Figure 28:</u> Multi-Layer perceptron architecture with GA.



C-Sharp class hierarchy

A C-Sharp executable is a window program shown inside a window form screen. The C-Sharp executable contains the Main form class, which is the parent class that contains classes for creating the user interfaces and for drawing graphics and images. Figure 29 shows the class hierarchy of intelligent goal finding application. The object contains a System component model, which in turn contains a data, which in turn contains a drawing, which in turn contains a text, which in turn contains Main window form class.

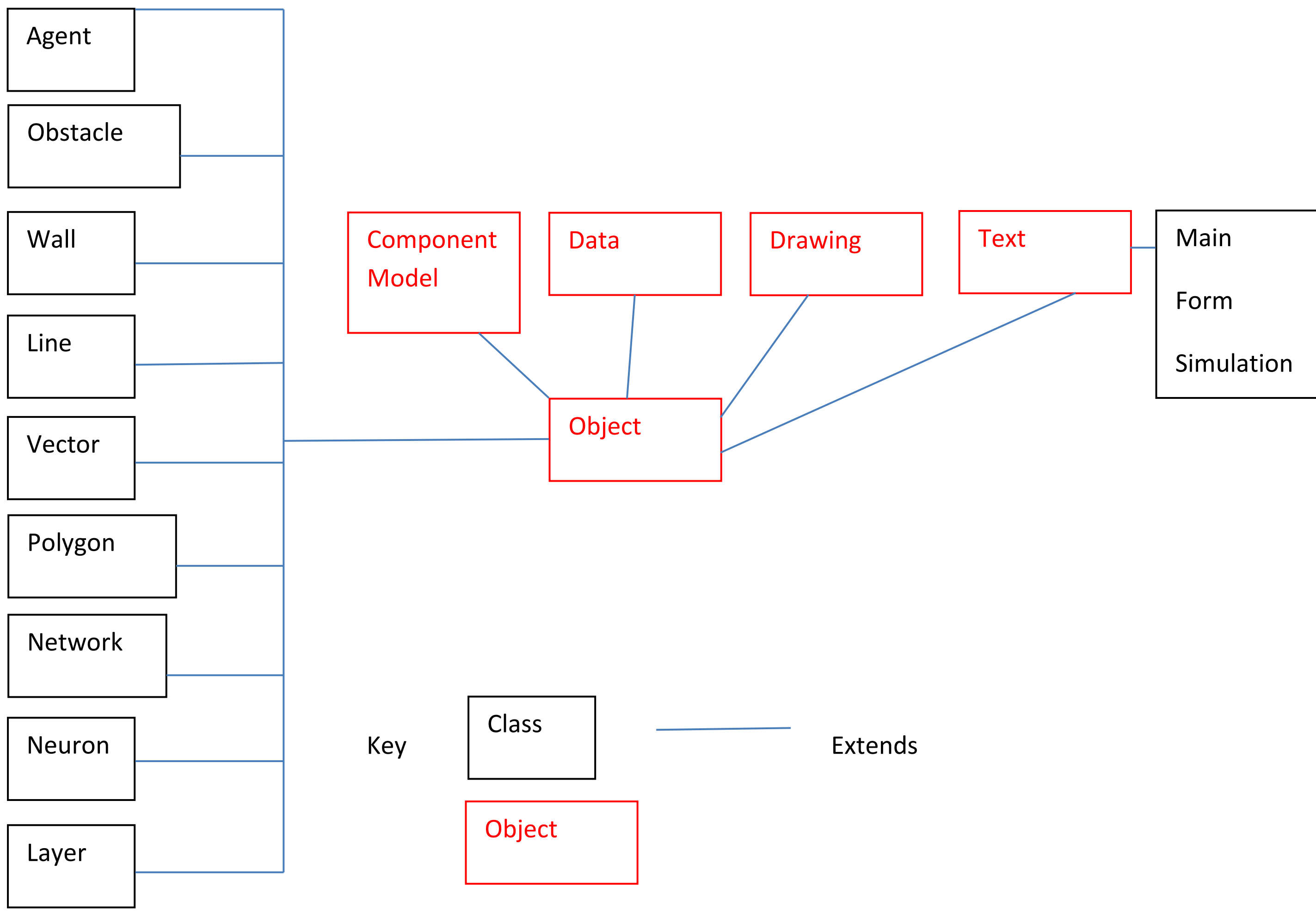


Figure 29: The C-Sharp class hierarchy diagram

Class responsibility collaborator model

A Class Responsibility Collaborator (CRC) card is a thinking method that was used to recognize the mechanisms and requests of strategy system for intelligent goal finding simulation. CRC Card Modeling is an object-focused investigation method. It is an active way for researchers and users to find out and comprehend the requests and strategy procedure. It is the goal of the CRC method to find out, check and inform the jobs or corresponding duties of classes and their relationships with additional classes on an abstract level. The jobs and duties of a class are the indication an object of the class has within, and the activities it can perform. CRC cards were organized before the application of the sample CRC card method was beneficial in accepting the jobs and duties of each class used. CRC method is a collection of regular file cards that have been separated into three divisions, as shown in Figure 30. In a CRC method, a card or portion of paper is prepared to signify an example of an object category. The objects duties is recognized and documented on the portion of paper. When one object calls another object, the second object is understood to be the first object's partner. The partnership of classes show which other classes a class has to partner in command to make accessible the mandatory utility. The designations, the duties, and the partnerships encapsulate the strategy at a junior level.

Class Name	
Duties	**Partners**

Figure 30: CRC card layout

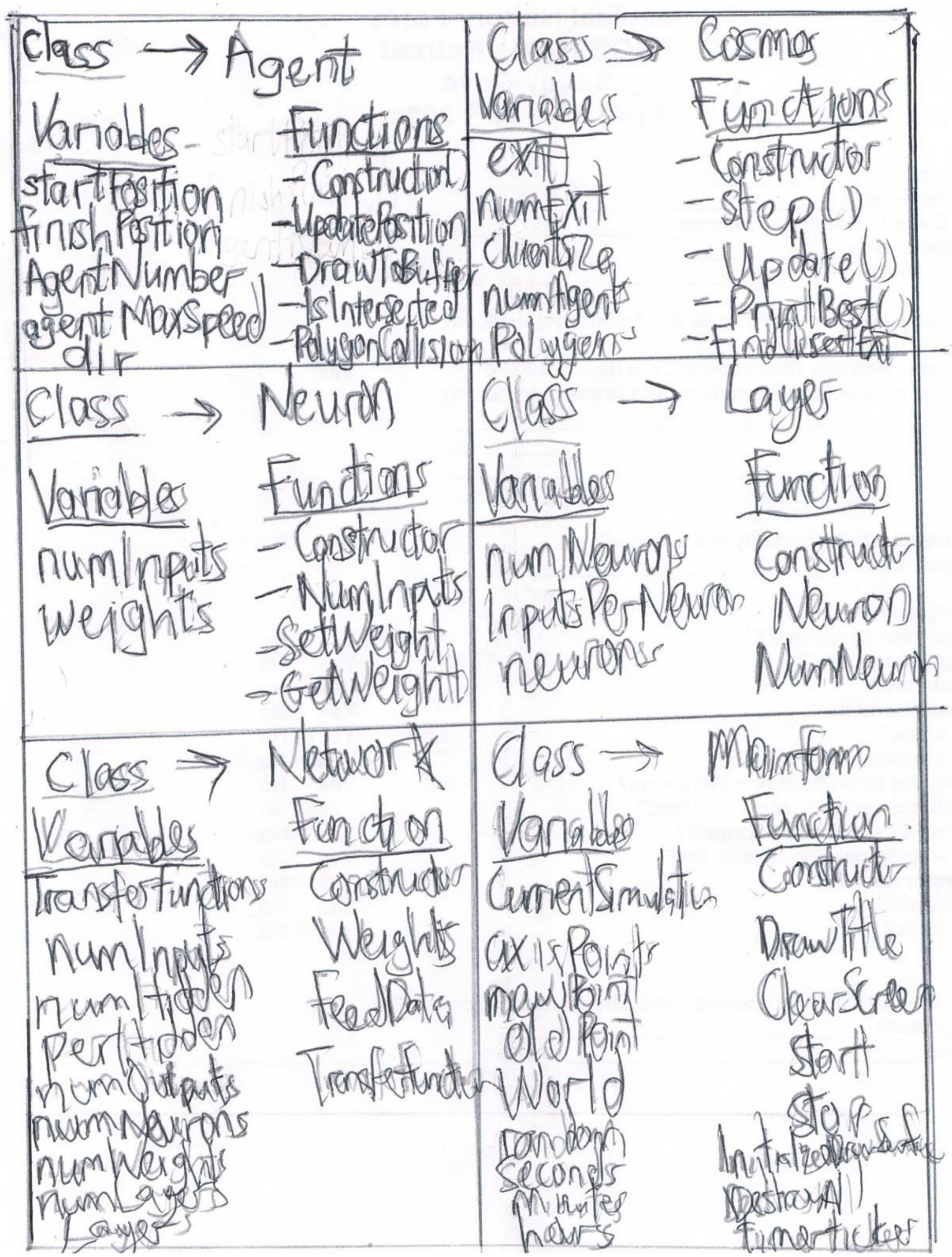


Figure 31: Six hand-drawn CRC cards

Class: A class signifies a group of comparable objects. An object is an individual, home, article, occasion, or thought that is pertinent to the structure considered. For instance, emergency classification can have classes like agent, exit, cosmos, network, layer, etc.

Duty: A duty is whatever that a class recognizes or organizes. For instance: Agent class can have functions: - constructor, update position, draw to buffer, is intersected, polygon collision, etc.

Partner: Sometimes a class has a work to fulfill, but does not have sufficient privileges to do it. For example, Agent class can have partner classes: cosmos, network, neuron, layer, cosmos, etc. The implementation of goal finding application in C-Sharp involved the use of fuzzy logic library.

The goal finding system was designed using CRC cards (refer to Figure 31) thinking method and contains numerous objects prearranged in a ranked style that can be understood. Figure 32 displays the UML diagram of intelligent goal finding application for the system order.

Limitation of Intelligent Goal Finding Application

Dynamic Obstacles: The intelligent goal finding application does not have the ability to add and remove obstacles once the simulation has started running. This would allow the researchers to study other interesting behaviors that can only be seen once a simulation has started running in the application.

Herding behavior: The intelligent goal finding application does not incorporate the herding behavior. Herding is a developing behavior that was projected by Craig Reynolds in 1986 for bird behavior. Herding behavior is used for mimicking agents with instructions to change their position. The instructions of herding behavior contain

1) Separation: change direction to evade massing confined herd companions.

2) Alignment: change direction to the typical direction of confined herd companions.

3) Cohesion: change direction toward the typical confined herd companions.

The instructions are for a herd of birds, pool of fish, or a swarm of insects. Herding appears as an asset of a collection of birds. Each bird performs as an autonomous agent and submits to the modest instructions by corresponding speed with neighboring herd companions. Individuals do not constantly obey the instructions of parting, alliance, and structure in alarm circumstances. However, the speed corresponding with close agents is a behavior detected in individuals and its absorption will provide stimulating outcomes.

Distinguishability: Intelligent goal finding application is incapable of distinguishing other agents in the location. For instance when one agent has noticed an incorrect goal then additional agents would not be capable to acquire this material from that agent. Also, the distinguishability of an exit also shows an essential part in disaster migrations. Intelligent goal finding application does not presently recognize the use of symbols, hues, and speech during the evacuation.

Transfer of preferred exit: Intelligent goal finding application is not capable of spontaneously allocate preferred exits to each agent. The user has to specify the exits as well as speed and level of stress and panic of an agent at the beginning of simulation using an input file. Agent is allowed to go through the process of finding and choosing the closest exit to them at start of the simulation. Currently there is no knowledge device for preferred exit. If the preferred exit is not achieved, the agent locates the subsequent closest exit in their view.

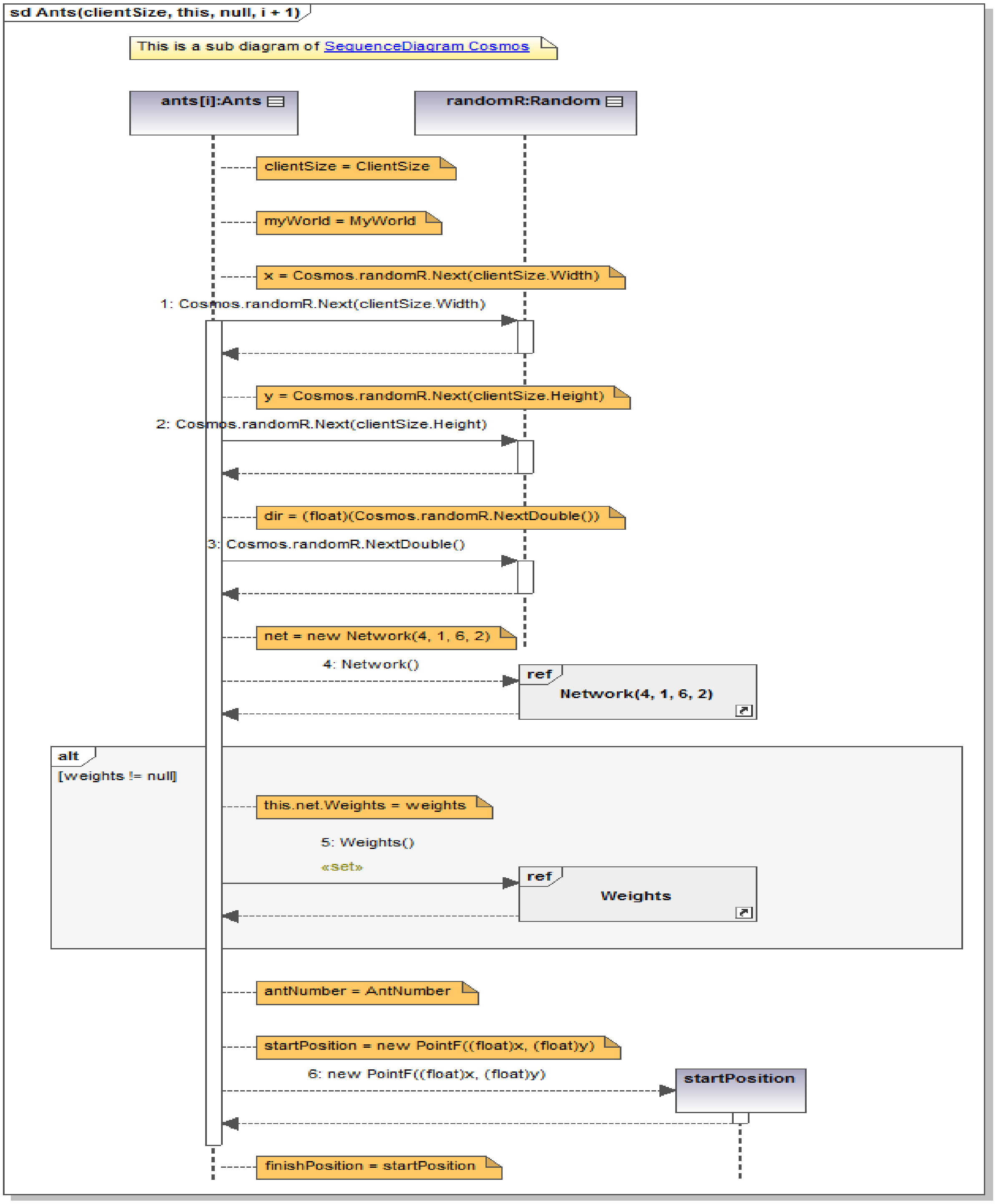


Figure 32: UML Diagram of intelligent goal finding application

Throng behavior: Intelligent goal finding does not include throng behavior. Throng behavior has been extensively investigated and is a technique of generating switch procedures for modest automatons by comprising knowledge and exploratory processes. It used for important conduct of ants, bees, termites, and additional communal insects. Throng brainpower is a shared behavior of self-governing agents like ants and bees deprived of management. Throng behavior is an accommodating and cooperative behavior of agents to attain some objective. The agents practice modest instructions to administer their joint arrangements through contact with situation. This behavior appears from the gathering of movements of the collection. However, individuals do not continuously perform together in terror circumstances. Their behavior becomes confused and self-centered in disaster situations. Nevertheless, the combination of this cooperative behavior will provide stimulating effects.

Sight impaired people: Intelligent goal finding application does not include the simulation of Sight impaired people. We have anticipated that individuals have a perception of preferred exit that is contingent upon their recollection and earlier communication with the setting. We also accept that individuals do not continually search for closest exit in disasters. We accept that each agent has a preferred exit at the start of simulation. Although for a sight impaired person, the user of intelligent goal finding application can allocate the exit from which the person is closest to as even a sight impaired individual may have recollection.

Graphical User Interface (GUI) of Intelligent Agent Goal Finding Application

The GUI of the intelligent agent goal application is broken down into three divisions. The top division consists of functions such as drop down menu items (start, pause, stop, initialize, settings, help, destroy all, and exit), and count down timer. The middle division displays the selected environment that contains the agents, walls, obstacles, and exits. The bottom division displays the statistics and current settings of the simulation such the transfer function and current generation. The functions of the graphical user interface are shown in Figure 33. The simulation presented on the middle division is displayed as the agents head to the exit.

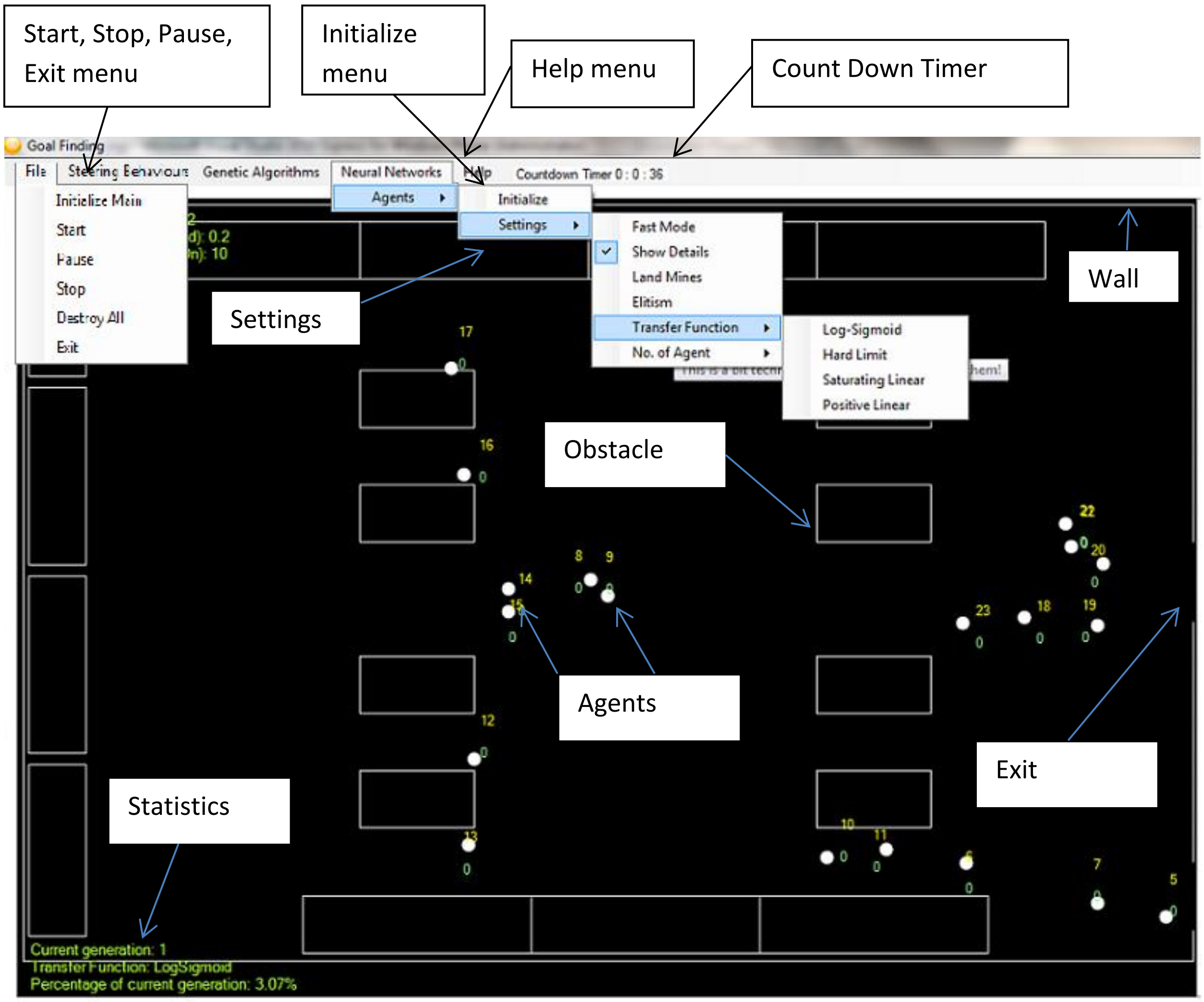


Figure 33: Functions of the intelligent agent goal finding application Graphic User Interface

Testing

Testing was done on the intelligent goal finding application to find out whether the simulation program performs according to expectation and to detect and fix bugs before it is released for general use. Testing the simulation done was done by running the program with generated mock-up data and verifying the outcome for faults, irregularities and non-functional requirements. The testing procedure was able to meet two major objectives:

1. **Testing every requirement**: each of the requirements shown in the CRC cards in Figure 31 was tested by using black box testing. This testing involved taking the input file shown in Table IV, running the simulation and then verifying to see if the correct number of agents, obstacles and exits was visible in the environment (refer to Figure 33). In addition the position of the agents, obstacles, walls and exits was also observed to see if it matched the input file coordinates. Furthermore the simulation was run with and without the genetic algorithm and neural network enabled to verify there was a difference in the speed, direction and evacuation time of the agents. The advantages of this kind of testing is that we do not need to know the inner working of C-Sharp programming language in which the simulation is based and we can focus on the GUI of the application.

2. **Testing irregularities**: white box testing was done to detect situations in the inner working of the application where the simulation could crash or go into any indefinite loop. Figure 32 shows the initialization of the neural network of the agent and a test was done to see what the behavior would be if a large number was entered in for the weights. The application was able to run successfully even though it was a little slow in its execution. A recommendation was then made to put some validation to make sure there is an acceptable range that the neural network can be set in the application. The advantages of this kind of testing is that it allows us to make sure all program paths are explored and it makes it easier to create test cases based on reviewing the code.

RESULTS AND DISCUSSIONS

The purpose of the research was to examine the behaviors that people acting as agents' exhibit during emergency evacuation situations. In those situations, the goal was to find the nearest exit. Furthermore, it sought to model learning and adaptive behavior, by focusing on individual agents changing their behavior as they receive external stimulus from the environment, and collective behavior such as crowd-modeling and emergency behavior. In addition new intelligent agent based characteristics such as autonomy, social ability, cooperativeness, learning ability and level of panic were examined as important factors to consider as the agents attempted to reach the exit goal. The following three research questions were based on the study's purpose and problem statement:

1. How can intelligent agents learn from their environment in a goal finding application for evacuation simulation?
2. What adaptive behavior and collective behavior are found in goal finding application for evacuation simulation?
3. Which agent-based characteristics affect the speed of finding exits in a goal finding application for evacuation simulation?

The previous chapters examined the purpose, problem statement, literature, and methodology employed in the current study. This chapter presents the data analysis and results.

The data analysis was straightforward and correlational in nature and the statistical analysis program used in the study was the Survey Monkey engine web application. The data displayed a normal distribution and positive correlation between a smaller size of occupants evacuating and a faster evacuation time. There was no noticeable correlation seen between the faster evacuation time and the number of runs of the evacuation application. Positive correlation was also identified between faster evacuation time and the type of behavior, such as stress and panic, exhibited by occupants. The next sections provide methods, design, measures, recruitment data analysis, and detail of the participant's demographic and descriptive results in the three real time drills that was done in our study.

Methods

The five-phase emergency evacuation study conducted over a three-month period, utilized a combination of both qualitative and quantitative methodologies. First phase of the study was to identify the sample of prospective participants and qualitative processes to be used in the analysis of the data. Second phase of the study was developing the questionnaire and the process of administering it to the prospective participants of the study. Data analysis after the study was the third phase of the study followed by the identification of strategies and recommendations phase. The last phase was the preparation of reports and feedback to stakeholders. These five phases made it possible to be compliant with published rules on the ethical conduct of emergency research, study participants and stakeholders were involved in creating the study questionnaire, data collection methods, and feedback and distribution plan.

At the end of the data analysis phase, over a long period lasting numerous months, we worked to categorize data-driven approaches that might speak to the greatest important hazard elements that potentially impacted the three major study outcomes. These outcomes include the start of the evacuation, how long it takes to evacuate, and reducing the chances of participants getting injured. All the study methods involving human participants had preceding evaluation and authorization of the Bowies State University Graduate School Institutional Review Board, and informed permission was acquired from each participant registered in every phase of human research. Confidentiality of the participants was also protected by not collecting identifiable information like their name as part of the study.

Study Design

In adjacent partnership with the chair of dissertation committee, a conceptual framework was developed consisting of evacuation behaviors, group behaviors, behavior intentions, environmental enabling factors and subjective norms. This model focused on the connection between individual and group behaviors, as well as the importance of the incorporation of fuzzy variables like stress and panic in emergency situations. The framework led to the development of key study concepts and specific survey questions.

The resultant two page, 21-item questionnaire included demographic information as well as the individual and group parameters that affect emergency evacuation. These parameters include the visibility of exits, queuing at exits, obstacles in the way of getting to an exit and working with others when evacuation a room. Duplicates of the questionnaire were organized in both printed- and internet-based presentations using survey monkey website. Wide-ranging authentication techniques were then implemented, including content, benchmark, and paradigm legitimacy processes. The concluding draft survey also experienced far-reaching intellectual and experimental testing, including trial tests of the internet type of the survey. Multiple choice questions were provided and different scales were utilized in the questionnaire. Copies of the survey are available by contacting the dissertation author.

Study Measures

Survey items examined the following five significant concepts:

1. *Distinct Safety Features*: Distinct safety features of participants include the social background and educational characteristics, height and weight ranges, knowledge of the building, pre and post event signs and behaviors, safety-related views and awareness of safety

and evacuation path. Items included in social background and education characteristics were age, gender, race/ethnicity and education. Knowledge of the building was addressed by asking if both exits in the study drill evacuation room were visible. Pre and post events signs and behaviors included measuring the stress and panic level of participant before, during and after the drill. Safety-related views were also measured by soliciting feedback on why an exit was chosen and whether queueing was experienced at an exit. Finally, the evacuation path was examined by checking on what obstacles got in the way of the participant on the way to the exit.

2. *Structural safety features:* Structural safety features included selecting someone to ensure everyone evacuated during the drill, providing written and verbal evacuation instructions, post-evacuation-selected meeting area and overall management of the drill.

3. *Building safety features:* Building safety features included the environment condition following the impact and during the evacuation such as lighting, queuing, accessibility of exits and visible signs in the room and hallway of the building.

4. *How long to start evacuation:* In order to find out the time it takes to start the evacuation, participants were asked to wait until they heard the evacuation drill warning sound. This sound was loud enough to simulate something serious had occurred and there was a need to leave or evacuate the room.

5. *How long to complete evacuation:* In order to find out the time it takes to complete the evacuation, a timer was present as recorded by the two video cameras located outside the doors of the exit in the room. This was also accompanied by noting the initial location of participants before the start of the drill to trace their path to the exit.

Participant enlistment and survey management

A comparable section of building emergency evacuees was developed from two large groups (1) a major random slice of Bowie State computer science undergraduate students in the computer lab selected at the time of each of the three studies and (2) Bowie State computer science graduate students available from a stakeholder's class willing to invest their time in the study. Of the 19 students in the first study and 25 students in the second and third drill, 16 students and 23 students respectively successfully evacuated. These people made up the final data cohort. This dataset was compared to the same number of agents in the goal finding application simulation of the same scenario records in the drills.

Thirty-nine people completed the paper pre-evacuation questionnaire and web based post evacuation questionnaire. An evaluation of the demographic features of the participant's response in the questionnaire between the first and second/or third drills did not show any significant statistical differences, except that for the larger number of evacuees in the second or third drill had a higher level of stress and panic before the drill. No difference by age, gender, or other demographic variables was seen in the participants between the three drills.

Data analysis

Experimental test data were reviewed to find out the value of data received for each feature. This included checking for missing behaviors in the drill, bimodal distribution, and consistency of answers (cross-validation). The weight of response choices were reviewed and changed to provide three new scales and to reduce the length of the final survey questions.

Once the full facts set was finished, and after data methods were edited and checking for inner reliable and validity of participants responses, a group of descriptive statistics was done (means, averages and standard deviations). Graphical methods were used to display the ranges of variables using bar graphs, pie charts and line graphs to fine tune the measurement. This approach provided awareness of the data trends, confirming whether the data met the hypotheses required for the purposed statistics testing methods. Fact assessments were used to measure the population items for the variables. Once each variable was certified, all the ranges of the questionnaire responses were analyzed for correctness.

Descriptive statistics for areas of importance, such as room and people-related variables, such as age, education, gender, and stress and panic were also elucidated. To find out the independent and dual connection of several features with evacuation consequences such as how long it takes to start and finish evacuating, chi-square statistics were done. A variable that was statistically important in the bivariate prototype were included in the analysis. The choice of exit, obstacles that were accounted, queueing and whether participants went back to get something in the room during the evacuation, were noted in the model

Limitation of the study data was the lack of making random the sample collected. We attempted to make random the sample but we did our recruitment from an easy to reach sample of computer science students available in the building at the time of the drill. However in further comparison of the demographic combination of participants, we were able to validate the makeup of the participants was sufficient for the study. Another potential limitation of the study was that participants who were students of one of the stakeholders might have provided responses that were expected of them in the study, although the anonymous way in which the questionnaire was structured may have reduced their fears in this regard. We also must remember the second and third drills, which were done back-to-back and whether participants may have used their memory to give similar responses. However, the use of video cameras to record both drills may have helped. We believe that the size of recruitment and quantitative validation of data and how similar to goal finding application simulation help provide support to validate our study.

Real-Time Drill Results

A. Study One

Sixteen people completed the first survey and provided informed consent to participate in the study. Therefore, the first study included a total of (N=16) participants. The demographic information collected before the first drill included the following: gender, age, ethnicity or race, and weight. The participant's demographic information is displayed in Table II.

Table II: Participant Demographic Study 1 (N=16)

Characteristics	Responses
Gender	
Male	10
Female	6
Age	
18-25	4
25-30	6
30-39	4
40+	2
Prefer not to answer	0
Characteristics	**Responses**
Ethnicity or race	
White	0
Hispanic or Latino	0

Black or African American	6
Native American or American Indian	0
Asian/Pacifica Islander	8
Other	1
Prefer not to answer	1
Weight	
Less than 100lbs	
> 100lbs and <= 150	8
> 150lbs and <= 200	4
> 200lbs and <= 250	2
> 250lbs and <= 400	1
Prefer not to answer	1
Stress	
Low	9
Medium	7
High	0
Panic	
Low	8
Medium	8
High	0

Descriptive information was collected from participants after the first drill, which focused on behaviors (learning and adaptive) exhibited by participants in the study. In Table III, the following descriptive information was collected: gender, overall quality of the evacuation drill, efficiency of the evacuation drill, clarity of the evacuation drill instructions, visibility of the exits to participants, reason for choosing an exit, stress and panic level of participants before, during and after the drill, count of people known previously during the drill and participants' behavior during the drill.

Table III: Descriptive Statistics on Participants Learning and Adaptive Behavior Study 1

Variable	Responses	Percentages
Gender		
Male	10	62.50%
Female	6	37.50%
Overall Quality of the drill?		
Outstanding	5	31.25%
Significant	7	43.75%
Average	3	18.75%
Marginal	1	6.25%
Almost none	0	0.00%
Efficiency of the drill?		
Outstanding	5	31.25%
Significant	8	50.00%
Average	2	12.50%
Marginal	0	0.00%
Almost none	1	6.25%
Clarity of drill instructions?		

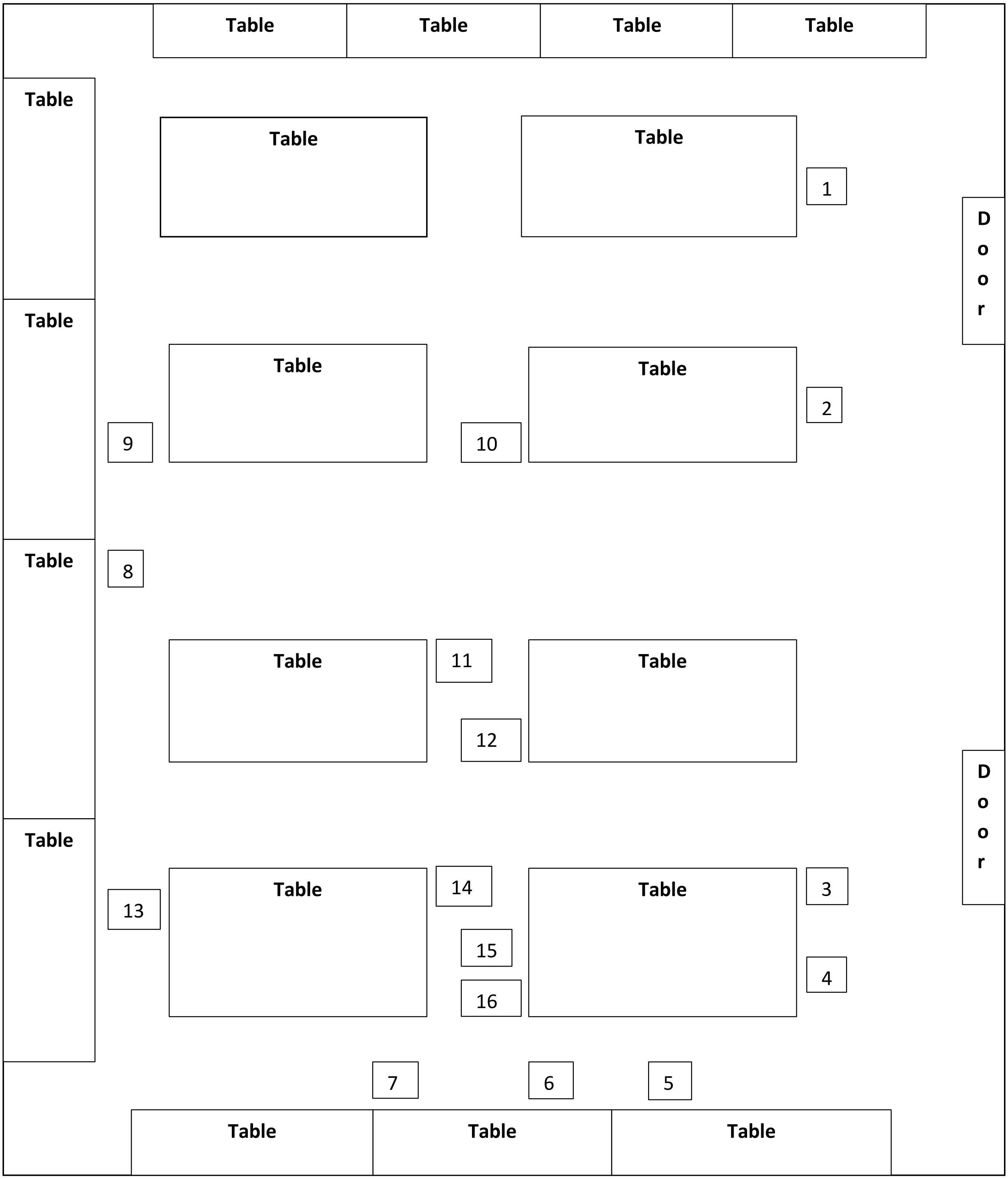
Yes	15	93.75%
No	1	6.25%
Followed drill instructions?		
Yes	16	100%
No	0	0.00%
Visibility of exits during drill?		
Outstanding	10	62.50%
Significant	5	31.25%
Average	1	6.25%
Marginal	0	0.00%
Almost none	0	0.00%
Reason for exit choice?		
Exit was near or close	10	62.50%
Followed other people	4	25.00%
Other exit was not visible	0	0.00%
Familiarity with exit	1	6.25%
Exit was not blocked	1	6.25%
Stress level before drill?		
Low	10	62.50%
Medium	6	37.50%
High	0	0.00%
Panic level before drill?		
Low	9	64.29%

Medium	5	35.71%
High	0	0.00%
Stress level during drill?		
Low	7	43.75%
Medium	9	56.25%
High	0	0.00%
Panic level during drill?		
Low	7	53.85%
Medium	6	46.15%
High	0	0.00%
Stress level after drill?		
Low	11	68.75%
Medium	4	25.00%
High	1	6.25%
Panic level after drill?		
Low	10	66.67%
Medium	5	33.33%
High	0	0.00%
People know previously?		
0-5	1	6.25%
6-10	3	18.75%
11-15	5	31.25%
16-20	7	43.75%

20-25	0	0.00%
Did you go back during drill?		
Yes	2	12.50%
No	14	87.50%
Slowed speed closer to exit?		
Yes	8	50.00%
No	8	50.00%
What obstacle got in the way?		
Other People	10	62.50%
Chair	2	12.50%
Table	4	25.00%
Experience queue at exit?		
Yes	8	50.00%
No	8	50.00%
Behavior during queuing?		
Waited patiently for exit	10	62.50%
Looked for another exit	2	12.50%
Went back to the room	0	0.00%
Talked to other people	4	25.00%
Used phone to surf web	0	0.00%

MAP OF ROOM 109 Computer Science Building, Bowie State University

Note: Square with numbers represents each participant's position in the room for first drill.



As discussed earlier in Chapter 3's Pilot Research and Preliminary Results section, real-time drills would be executed for the evacuation of computer lab 109, Computer Science Building at Bowie State University. The drills were done to validate the C-Sharp intelligent goal finding application. Figure 21 in Chapter 3 shows a similar situation simulated in the C-Sharp intelligent goal finding application for the equivalent evacuation. During the study, fuzzy parameters like stress and panic were chronicled by two cameras posted outside the two doors shown in the previous map of Room 109 for the 16 participants of the first study drill. Nevertheless, for displaying the behavior of ratio of participant panicked, we matched the same parameters in the study 1 to the parameters in intelligent agent application. The graph in Figure 34 represents the total evacuation time vs. number of people evacuating for drill 1 performed for computer lab Room 109, computer science lab at Bowie State University. It took 7 seconds for the first two participants to evacuate, 8 seconds for the next two participants to evacuate, 9 seconds for the next two participants to evacuate, 10 seconds for the next two participants, 13 seconds for the next two participants, 21 seconds for the next two participants, and 25 seconds for the next two.

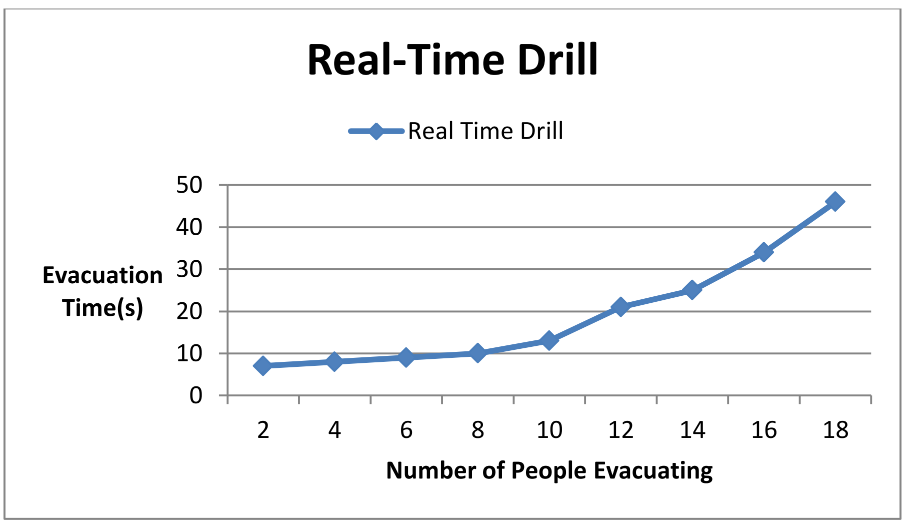


Figure 34: Graph shows number of people evacuating vs. total evacuation time for drill 1 at Room 109, Computer Science Building, Bowie State University

The parameters such as the location of obstacles, location of exits, location of participants, and stress and panic level were passed into the C-Sharp intelligent agent simulation model, using an input file as shown in Table IV.

Table IV: Input File Showing Parameters for Intelligent Goal Finding App Study 1

```
subGoalRad 1.5

Obstacles 19

obstacle 1
300 250
400 250
400 300
300 300
obstacle 2
800 500
700 500
700 550
800 550

………..

goal 0.79999 0.4600
goal 0.79999 0.47000
………

agentCircles 16
peopleCircle
type hostile
position 810 160
stress 10
panic 20
speed 1.19
peopleCircle
type hostile
position 870 260
stress 10
panic 10
speed 1.19
```

The graph in Figure 35a shows that 62.5% of participants could see all the exits during evacuation drill 1, and these correlated with Figure 35b, where the exit was near or close was chosen by 62.5% of the participants in the study as the reason for choosing an exit.

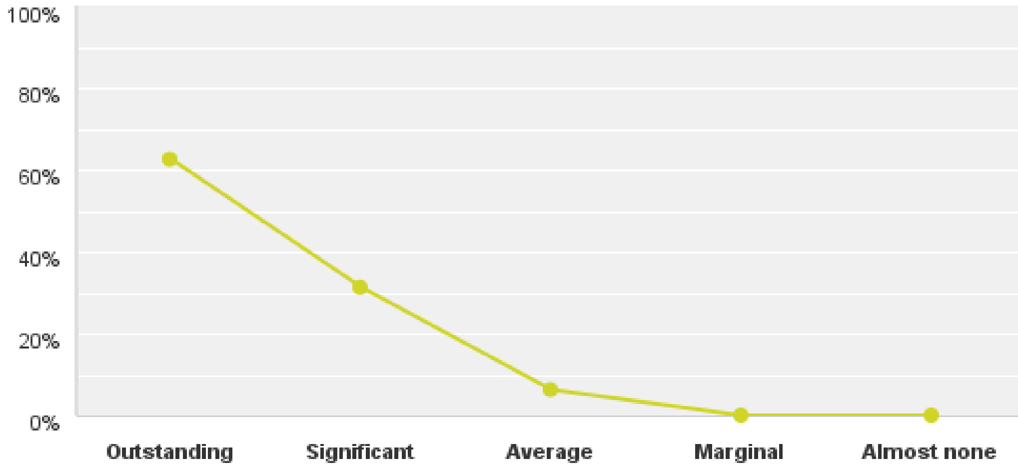


Figure 35a: Graph shows visibility of exit response, for drill 1 at Room 109, Computer Science Building, Bowie State University

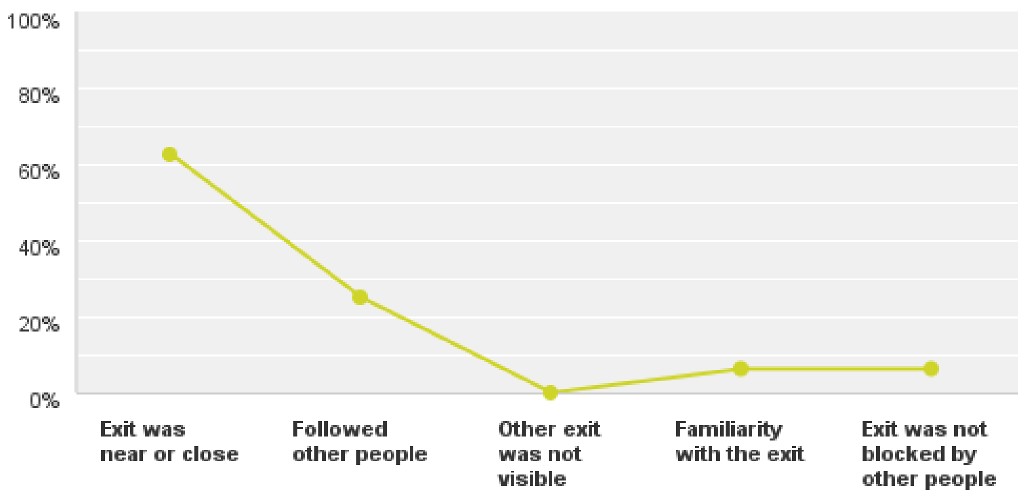


Figure 35b: Graph shows reason for choosing an exit for drill 1 at Room 109, Computer Science Building, Bowie State University

Visibility of exits also affected the level of stress and panic during emergency evacuation. In study 1, as shown in Figure 36, stress level and panic level was mainly low and medium before the drill, slightly increasing during the drill and decreasing after the drill. These parameters of stress and panic were represented and observed in the goal finding application.

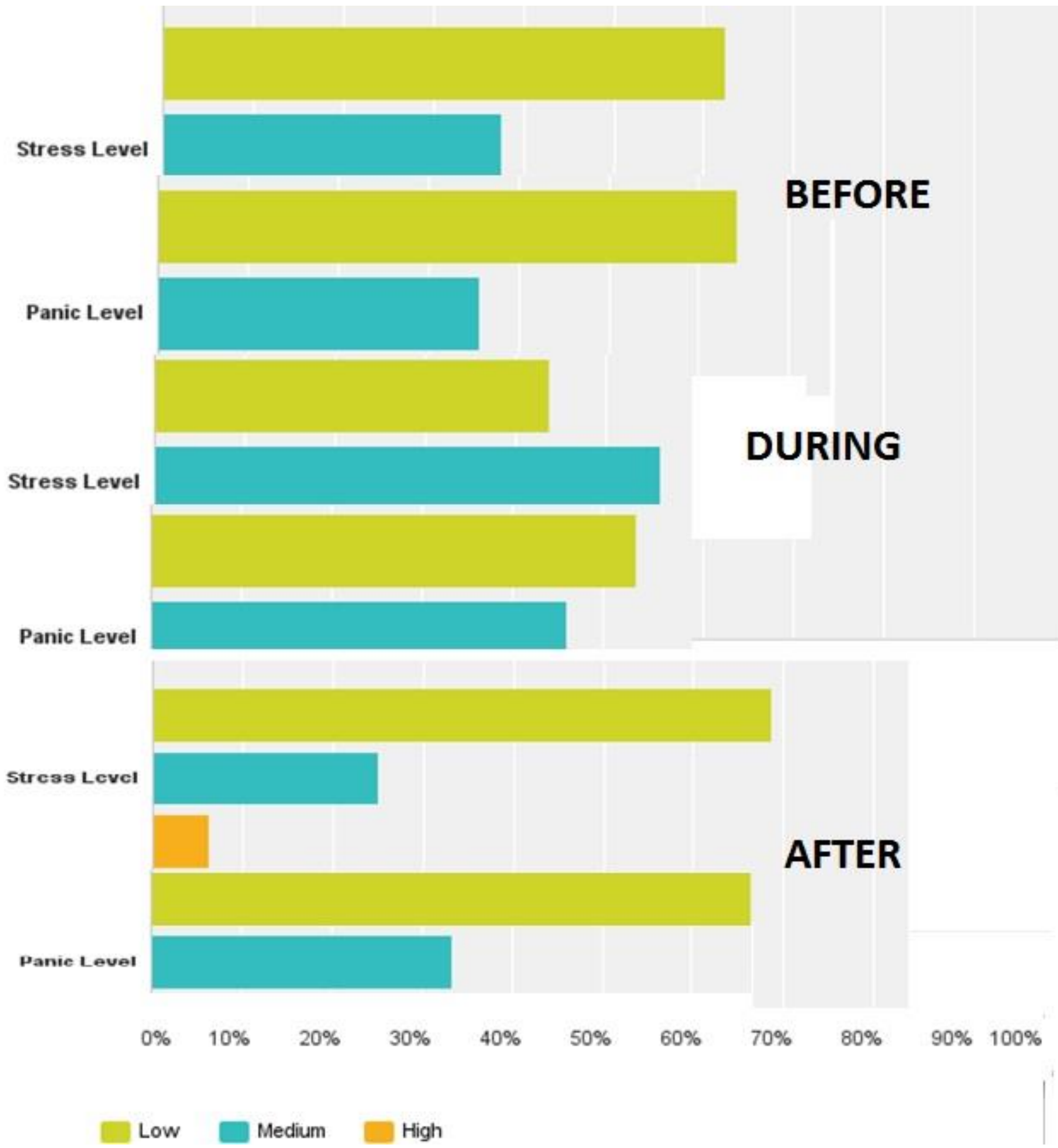


Figure 36: Graph shows stress and panic level before, during and after drill 1 at Room 109, Computer Science Building, Bowie State University

Stress and panic level was low before the drill and the data showed that it may be related to the results that majority of the participants knew each other previously before the drill, as shown in Figure 37. Fifty percent of the participants experienced queuing at the exit while trying to evacuate, and Figure 38 shows that they were able to patiently wait for their turn and talk to other participants before they evacuated the room.

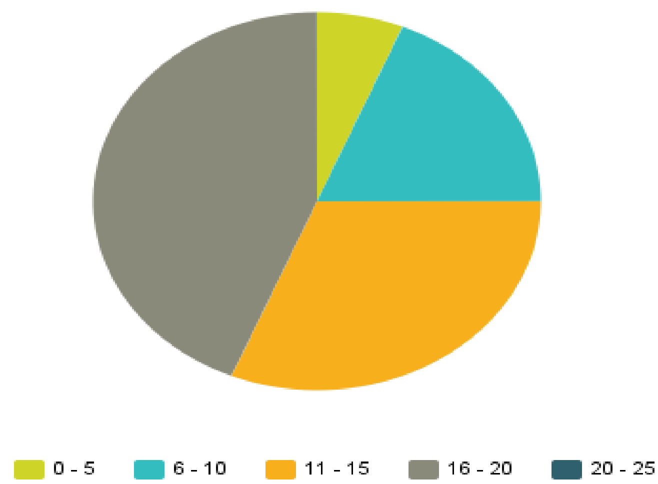


Figure 37: Graph shows number of people previously known before drill 1 at Room 109, Computer Science Building, Bowie State University

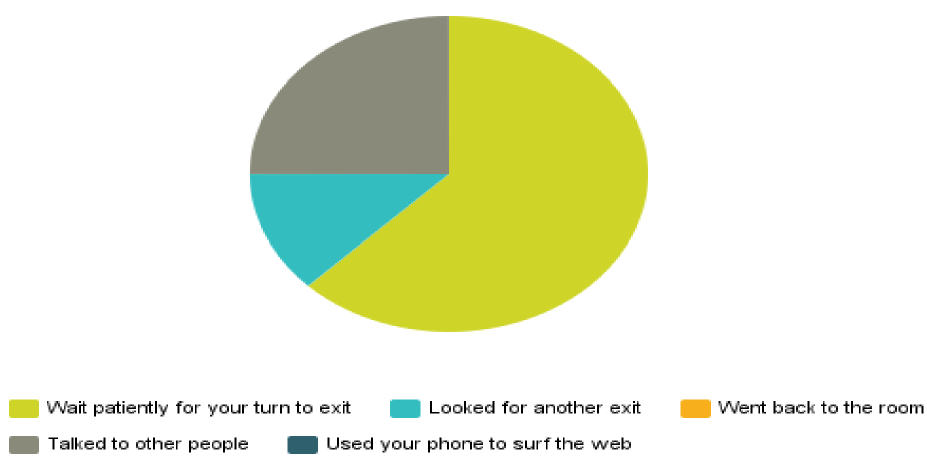
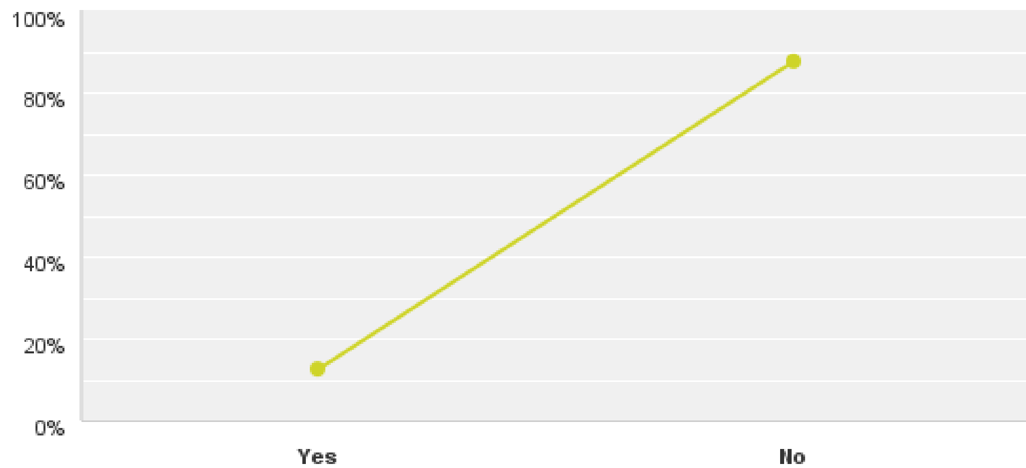
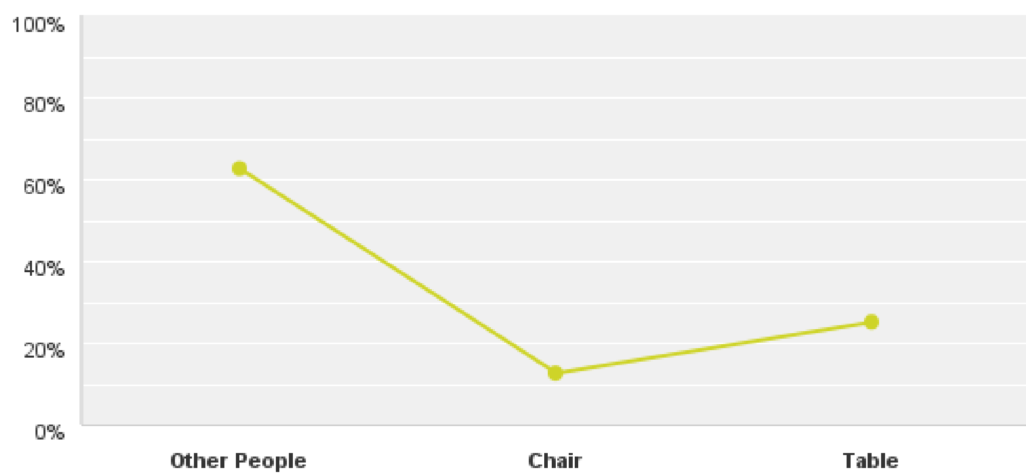


Figure 38: Graph shows what participants did when experiencing queuing at exit during drill 1 at Room 109, Computer Science Building, Bowie State University

During the evacuation, 50% of the participants slowed down their speed, as they got closer to the exit. In addition, a majority of them did not go back to get anything they left in the room, as shown in Figure 39. It was also seen that the major obstacle getting in the way of participants exiting the room was other participants, and other table and chairs in the room, as shown in Figure 40.



<u>Figure 39:</u> Graph shows percentage of participants who did not go back to get anything left in the room during drill 1 at Room 109, Computer Science Building, Bowie State University



<u>Figure 40:</u> Graph shows obstacles in participant's way during drill 1 at Room 109, Computer Science Building, Bowie State University

After the first drill was completed, we ran the intelligent agent goal application for three sets of 30 runs, matching the same parameters (refer to Table IV) seen in the drill. The first set of 30 runs was executed with the neural network and genetic algorithm enabled, second set with neural network and genetic algorithm disabled and last set with fuzzy panic and stress parameter enabled in the application. The evacuation time was recorded for all the runs, as shown in Figure 41.

NN and GA																															
No evac/	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**	**Avera**
2	20	19	18	17	21	20	20	21	19	20	21	19	20	21	21	20	21	21	20	20	21	21	20	20	19	20	21	21	20	21	20.1
4	21	22	23	21	21	21	22	23	21	22	23	23	21	22	22	23	23	22	23	21	22	23	23	21	23	23	21	22	23	23	22.1
6	26	25	24	26	24	24	24	22	23	23	23	24	24	25	24	25	26	24	26	25	24	26	26	25	24	24	24	24	25	24	24.4
8	27	28	27	27	26	27	27	26	27	27	27	26	26	27	26	26	27	26	27	27	26	27	27	26	27	27	26	26	27	27	26.7
10	29	30	29	29	30	29	28	29	29	30	29	29	29	30	29	29	30	30	30	30	29	30	30	30	30	29	30	30	30	30	29.5
12	40	41	41	40	41	41	40	41	40	41	41	40	40	41	41	40	39	40	41	41	40	41	40	41	41	40	41	41	40	41	40.5
14	44	45	44	44	45	44	44	45	45	44	45	45	45	44	45	45	45	45	44	45	45	44	44	44	44	45	45	45	44	42	44.5
16	52	53	52	53	53	53	52	52	53	53	52	53	53	53	52	53	53	53	52	53	53	53	52	53	53	53	53	53	53	53	52.7
18	59	64	59	59	64	60	59	64	65	60	59	60	63	64	59	60	61	63	59	59	59	59	59	59	59	64	64	64	64	64	61.2
Without NN and Ga																															
No evac/	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**	**Avera**
2	14	15	15	14	15	15	15	14	14	14	14	15	14	14	14	15	14	14	14	14	14	15	14	14	14	14	15	14	14	14	14.3
4	15	15	14	15	15	15	15	15	14	15	15	15	14	15	15	15	15	15	15	15	15	15	15	15	15	15	15	15	15	14	14.9
6	17	16	16	16	17	16	16	17	16	16	16	17	16	16	17	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16.2
8	20	21	20	21	22	20	20	21	21	20	20	21	22	22	21	20	22	22	20	21	20	20	20	21	22	20	20	21	20	21	20.7
10	30	30	30	30	30	30	30	30	30	30	30	39	30	30	30	30	30	30	30	30	29	30	30	30	30	29	30	30	30	30	30.2
12	42	43	44	42	43	43	43	42	42	43	44	42	42	43	43	44	42	42	42	43	42	42	43	43	42	42	43	42	43	43	42.6
14	45	48	47	48	46	47	48	47	48	48	47	48	48	47	48	48	47	48	48	48	47	47	48	48	47	48	48	47	48	47	47.5
16	56	55	56	56	57	56	56	56	56	56	56	56	57	56	56	56	57	56	56	56	57	57	56	56	56	56	56	56	57	56	56.2
18	58	60	59	60	60	59	64	64	65	64	64	64	66	66	66	65	66	66	66	66	65	66	66	66	65	66	62	65	66	66	64
Fuzzy																															
No evac/	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**	**Avera**
2	22	23	22	24	22	22	23	22	22	24	23	22	22	22	23	23	24	24	22	22	22	23	23	23	24	24	24	22	22	22	22.7
4	24	23	23	24	25	24	24	24	25	24	24	24	24	24	25	24	24	24	25	23	24	24	24	25	25	24	24	24	24	24	24.1
6	34	35	36	36	36	35	37	36	36	36	36	36	36	36	36	36	16	36	36	35	36	36	36	35	35	35	36	36	36	36	35.1
8	39	38	38	39	39	38	38	39	39	37	38	38	39	38	39	39	38	38	39	39	38	38	39	39	38	39	39	38	39	39	38.5
10	40	41	41	42	40	42	42	42	42	42	42	41	41	41	42	42	42	42	41	41	42	42	42	41	42	42	41	42	42	42	41.6
12	44	43	44	44	43	45	44	44	44	43	44	45	43	44	44	44	45	45	44	44	44	43	44	44	44	44	45	45	44	44	44
14	46	45	46	46	44	45	46	46	46	45	45	46	46	46	45	45	46	46	45	46	45	46	46	45	46	46	45	46	46	46	45.6
16	48	49	50	48	48	49	49	50	50	49	49	48	49	49	50	50	50	49	48	49	49	48	50	50	49	49	48	49	49	49	49
18	65	61	65	64	65	68	65	65	68	65	64	65	65	60	65	65	68	65	66	66	65	66	65	66	66	65	66	66	65	66	65.2

Figure 41: Table shows evacuation time generated from intelligent agent goal application using same parameters from drill 1 at Room 109, Computer Science Building, Bowie State University

We ran the simulation numerous times for the aggregate evacuation period, each time increasing the number of people evacuating up to the maximum number of participants seen in the real-time drill. We increased the number of participants evacuating in intervals of 2, 4, 6, 8, 10, 12, 14, 16, and 18. Thus, in total, we performed the simulation 90 times to get a more minutiae curve for stress and panic, which integrates average and standard deviation. The error bars on the chart in Figure 42 represents the standard deviation in time for percentage of participants evacuating. The points on the chart represent the average time for each interval. The graphs show the variation of total evacuation time when the percentage of participants evacuating was increased. Overall, the curve of Fuzzy NN and GA is going uphill the highest, which displays data that when more parameters of stress and panic are added, there is a slight increase in the evacuation time. According to the model, wait time of participants increases when panic and stress is high. This delay is seen in real life with participants becoming unpredictable when it comes to making decisions in emergencies. As a result, their behavior becomes erratic. At each interval, we took average time of evacuation. The error bar on bar graphs shows the difference in average time with 95% confidence level. For 16 people evacuated from Room 109, we can see that it took an average time of 65.2 for Fuzzy, 64 for No NN and GA and 61.2 for NN and GA with 30 simulation runs done. Table V shows screen shots of agents evacuating matching study 1.

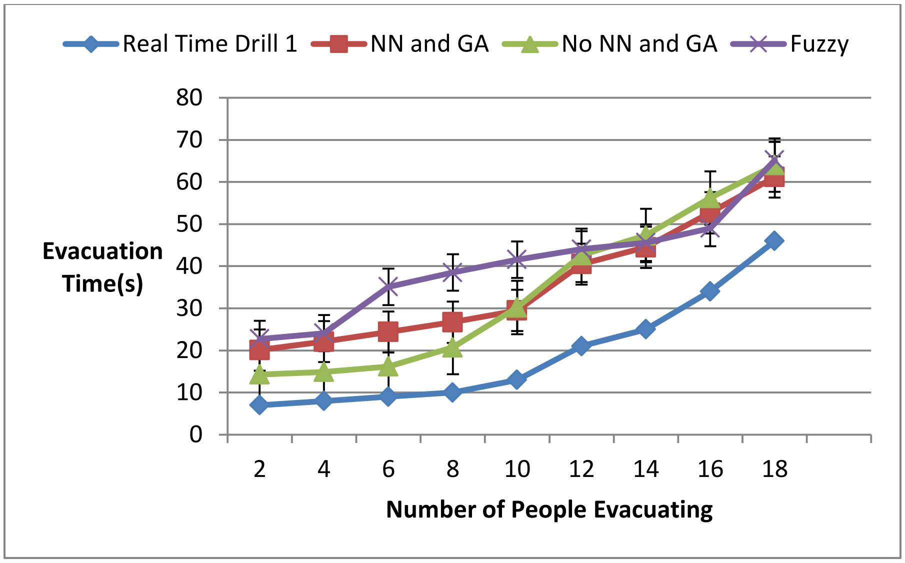
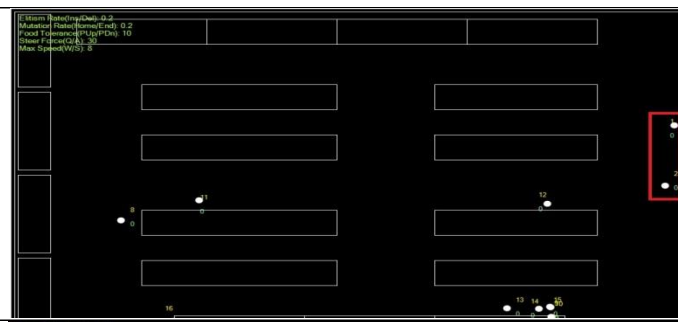
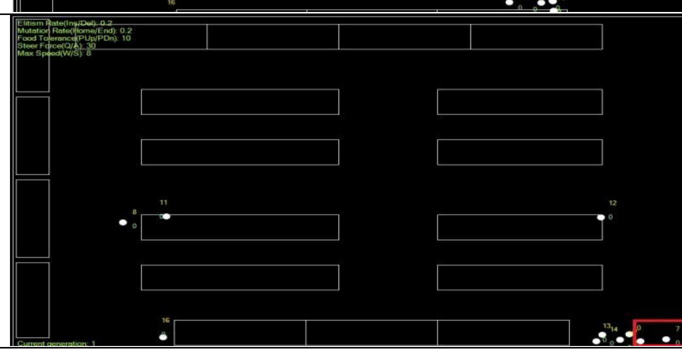
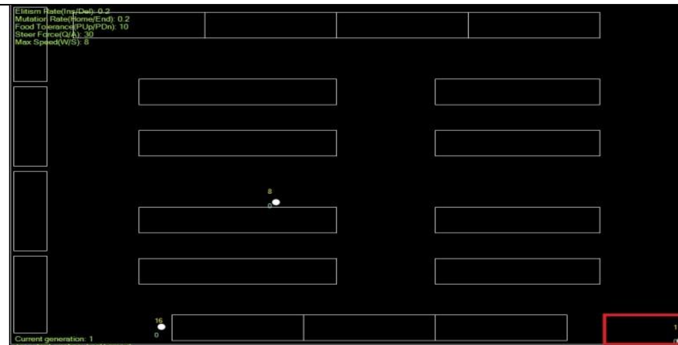
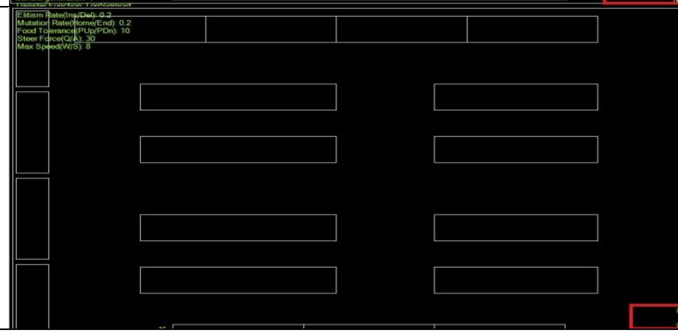


Figure 42: Graphs show evacuation time comparison Real-Time Drill 1 versus NN and GA versus no NN and GA versus Fuzzy NN and GA

Table V: Screen shots for agent evacuating intelligent goal finding application Study 1

Time Elapsed in seconds	Input Variables	Screen Shot
20	agentCircles=18 NumAgentsExited=2	
22	agentCircles=18 NumAgentsExited=4	

26	agentCircles=18 NumAgentsExited=6	
29	agentCircles=18 NumAgentsExited=8	

B. Study Two

A second study was done with a larger group of 23 people, who attempted the evacuation study, and 18 people who completed the second survey and provided informed consent to participate in the study. The second study was done twice back-to-back, with the same participants to ensure that we did not miss capturing any of the parameters or variables needed for our simulation. The second study included a total of (N=23) participants. The demographic information collected before the second drill included the following: gender, age, ethnicity or race, and weight. The participant's demographic information is displayed in Table VI.

Table VI: Participant Demographic Study 2 (N=23)

Characteristics	Responses
Gender	
Male	15
Female	7
Age	
18-25	7
25-30	7
30-39	2
40+	3
Prefer not to answer	4
Characteristics	**Responses**
Ethnicity or race	
White	0
Hispanic or Latino	0
Black or African American	9
Native American or American Indian	0
Asian/Pacifica Islander	10
Other	0
Prefer not to answer	3
Weight	
Less than 100lbs	

> 100lbs and <= 150	12
> 150lbs and <= 200	3
> 200lbs and <= 250	3
> 250lbs and <= 400	0
Prefer not to answer	5
Stress	
Low	11
Medium	8
High	2
Panic	
Low	15
Medium	5
High	0

Descriptive information was collected from participants after the second drill, which focused on behaviors (learning and adaptive) exhibited by participants in the study. In Table VII, the following descriptive information was collected: gender, overall quality of the evacuation drill, efficiency of the evacuation drill, clarity of the evacuation drill instructions, visibility of the exits to participants, reason for choosing an exit, stress and panic level of participants before, during and after the drill, count of people known previously during the drill and participants behavior during the drill.

Table VII: Descriptive Statistic on Participants Learning and Adaptive Behavior Study 2

Variable	**Responses**	**Percentages**
Gender		
Male	12	66.67%
Female	6	33.33%
Overall Quality of the drill?		
Outstanding	6	33.33%
Significant	7	38.89%
Average	3	16.67%
Marginal	1	5.56%
Almost none	1	5.56%
Efficiency of the drill?		
Outstanding	6	33.33%
Significant	8	44.44%
Average	2	11.11%
Marginal	0	0.00%
Almost none	2	11.11%
Clarity of drill instructions?		
Yes	16	88.89%
No	2	11.11%
Followed drill instructions?		
Yes	18	100%
No	0	0.00%

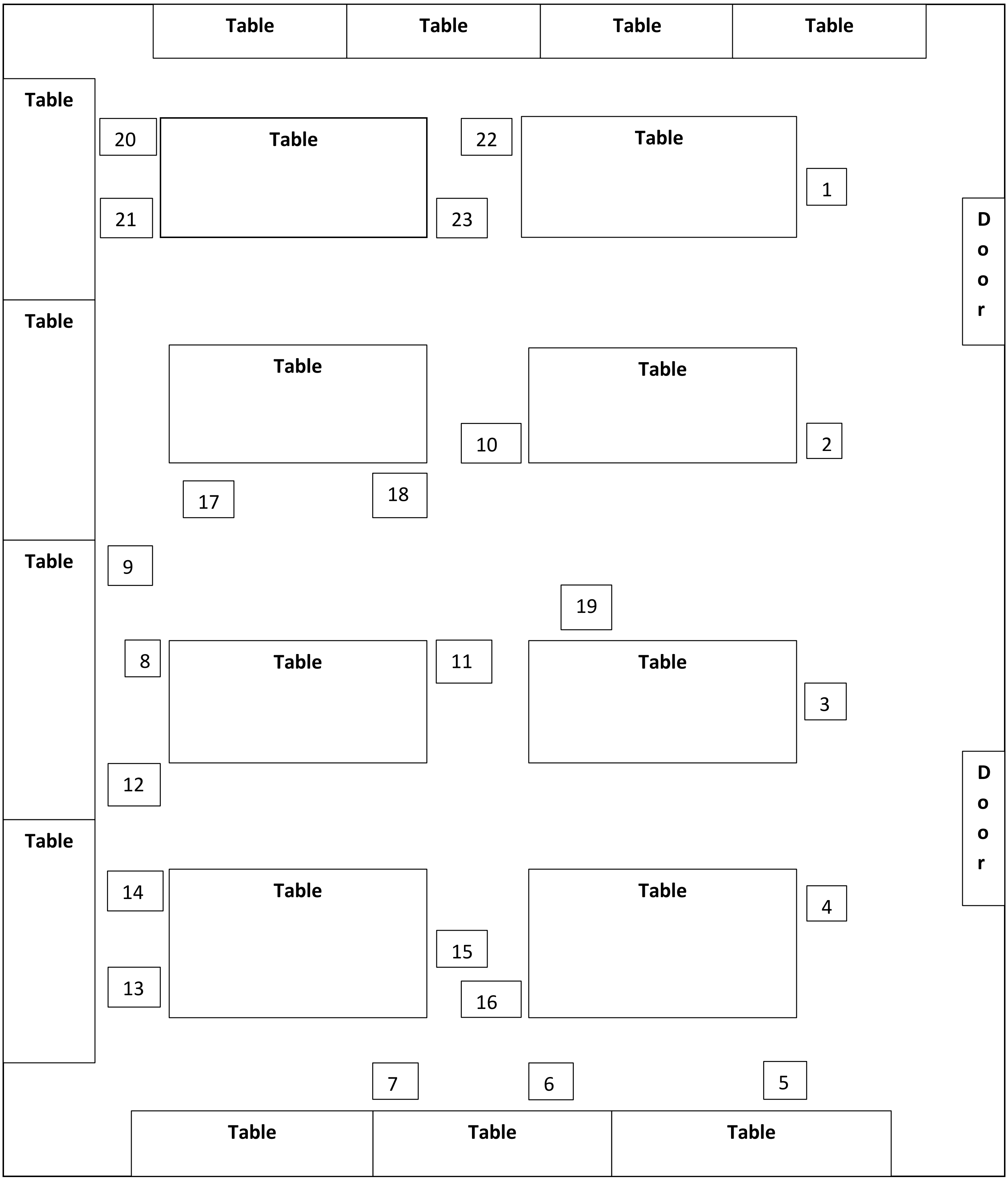
Visibility of exits during drill?		
Outstanding	12	66.67%
Significant	5	27.78%
Average	1	5.56%
Marginal	0	0.00%
Almost none	0	0.00%
Reason for exit choice?		
Exit was near or close	12	66.67%
Followed other people	4	22.22%
Other exit was not visible	0	0.00%
Familiarity with exit	1	5.56%
Exit was not blocked	1	5.56%
Stress level before drill?		
Low	11	61.11%
Medium	6	33.33%
High	1	5.56%
Panic level before drill?		
Low	11	68.75%
Medium	5	31.25%
High	0	0.00%
Stress level during drill?		
Low	7	38.89%
Medium	10	55.56%

High	1	5.56%
Panic level during drill?		
Low	9	60.00%
Medium	6	40.00%
High	0	0.00%
Stress level after drill?		
Low	11	61.11%
Medium	5	27.78%
High	2	11.11%
Panic level after drill?		
Low	11	64.71%
Medium	6	35.29%
High	0	0.00%
People know previously?		
0-5	1	5.56%
6-10	4	22.22%
11-15	5	27.78%
16-20	7	38.89%
20-25	1	5.56%
Did you go back during drill?		
Yes	2	11.11%
No	16	88.89%
Slowed speed closer to exit?		

Yes	8	44.44%
No	10	55.56%
What obstacle got in the way?		
Other People	11	64.71%
Chair	2	11.76%
Table	4	23.53%
Experience queue at exit?		
Yes	8	44.44%
No	10	55.56%
Behavior during queuing?		
Waited patiently for exit	12	66.67%
Looked for another exit	2	11.11%
Went back to the room	0	0.00%
Talked to other people	4	22.22%
Used phone to surf web	0	0.00%

MAP OF ROOM 109 Computer Science Building, Bowie State University

Note: Squares with numbers represent each participant's position in the room for second drill.



evacuation of computer lab 109, Computer Science Building, Bowie State University. The drills were done to validate the C-Sharp intelligent goal finding application. Figure 21 showed a similar situation simulated in the C-Sharp intelligent goal finding application for the equivalent evacuation. During the study, fuzzy parameters like stress and panic were chronicled by two cameras posted outside the two doors shown in the previous map of Room 109 for the 23 participants of the second and third study drills.

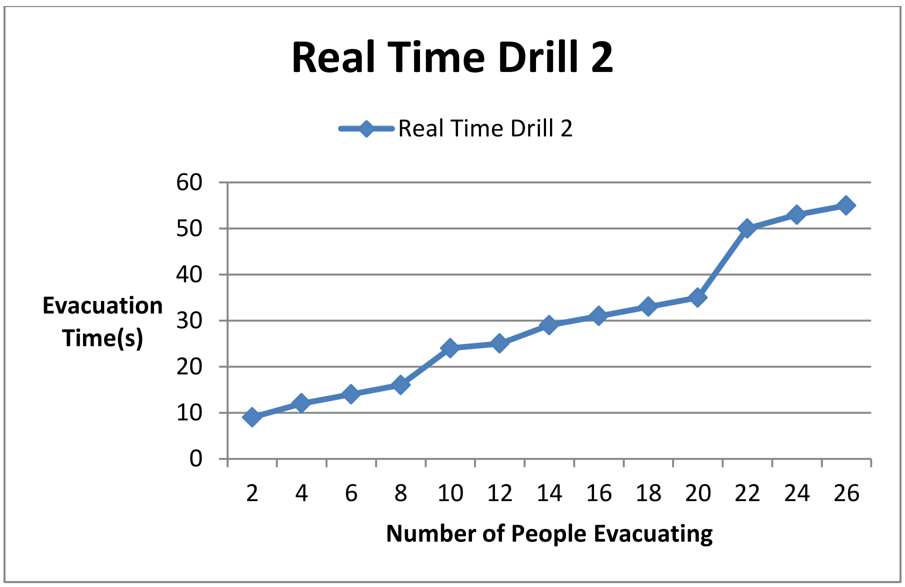


Figure 43: Graph shows number of people evacuating vs. total evacuation time for drill 2 at Room 109, Computer Science Building, Bowie State University

The graph in Figure 43 represents the total evacuation time vs. number of people evacuating for drill 2 performed for computer lab Room 109 computer science lab at Bowie State University. It took 9 seconds for the first two participants to evacuate, 12 seconds for the next two participants to evacuate, 14 seconds for the next two participants to evacuate, 16 seconds for the next two participants, 24 seconds for the next two participants, 25 seconds for the next two participants, 29 seconds for the next two participants, 31 seconds for the next two participants, 33 seconds for the next two participants, 35 seconds for the next two seconds, 50 seconds for the next two seconds, 53 seconds for the next two seconds and 55 seconds for the last two participants.

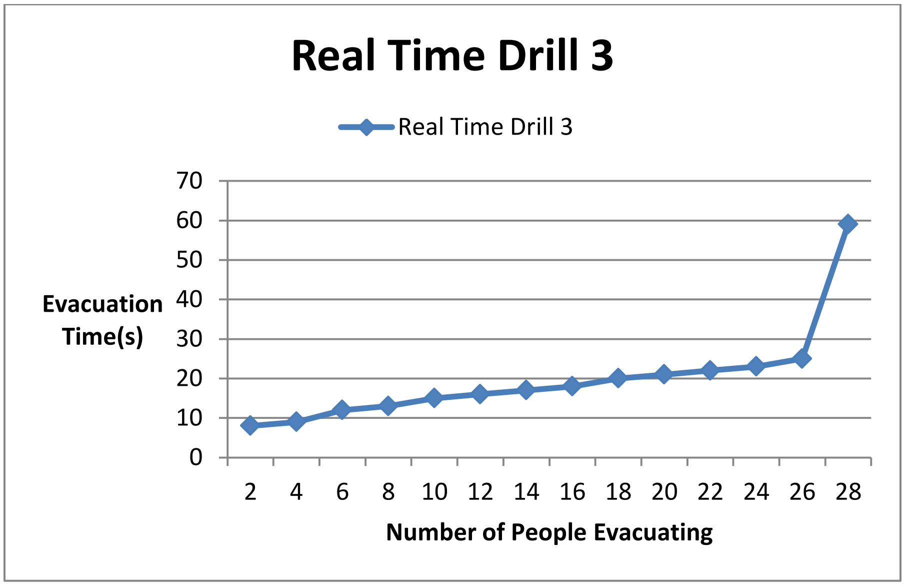


Figure 44: Graph shows number of people evacuating vs. total evacuation time for drill 3 at Room 109, Computer Science Building, Bowie State University

The graph in Figure 44 represents the total evacuation time vs. number of people evacuating for drill 3 performed for computer lab Room 109, Computer Science lab at Bowie State University. It took 8 seconds for the first two participants to evacuate, 9 seconds for the next two participants to evacuate, 12 seconds for the next two participants to evacuate, 13 seconds for the next two participants, 15 seconds for the next two participants, 16 seconds for the next two participants, 17 seconds for the next two participants, 18 seconds for the next two participants, 20 seconds for the next two participants, 21 seconds for the next two seconds, 22 seconds for the next two participants, 23 seconds for the next two participants, 25 seconds for the next two participants and 59 seconds for the last participant.

The graph in Figure 45a on the left shows that 66.67% of participants could see all the exits during evacuation drills 2 and 3, and these correlated with Figure 45b; the exit was near or close was the reason chosen for selecting an exit by 66.67% of the participants in the study. This matched the results we saw in the first drill.

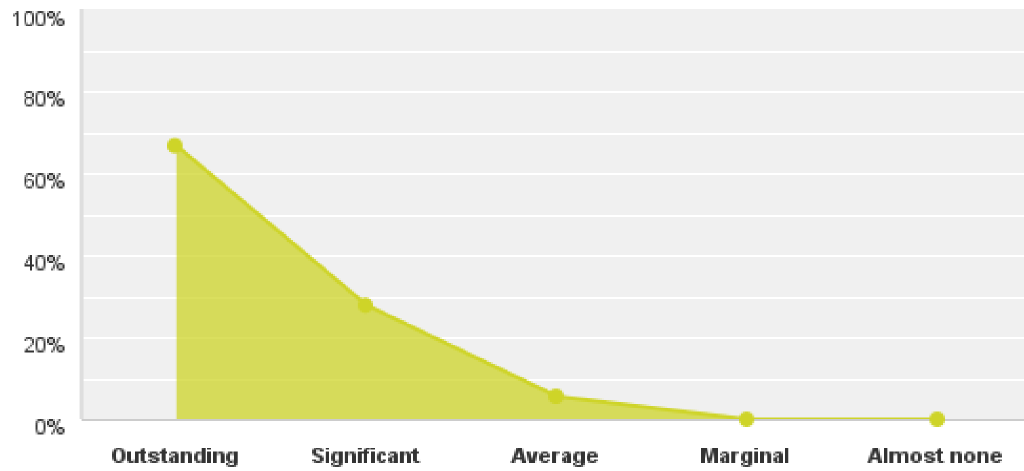


Figure 45a: Graph shows visibility of exit response, for drills 2 and 3 at Room 109, Computer Science Building, Bowie State University

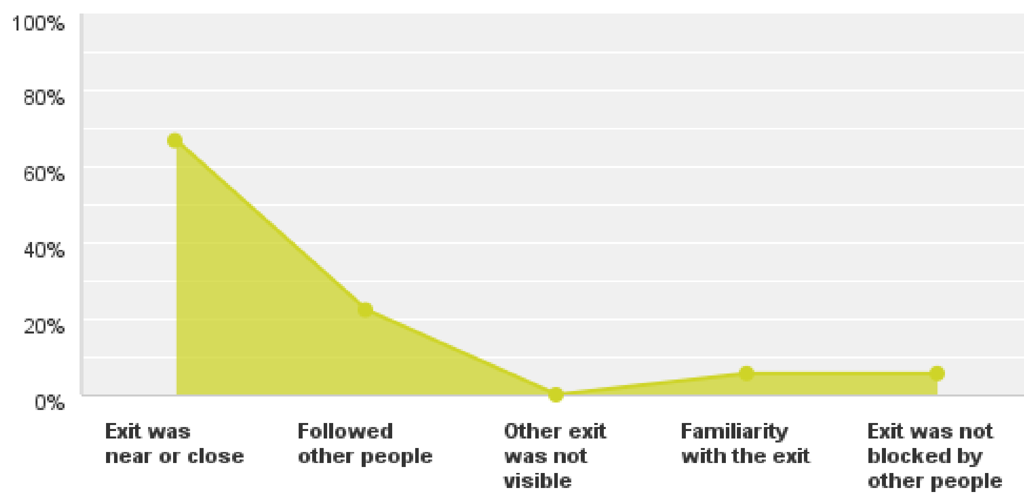


Figure 45b: Graph shows reason for choosing an exit for drills 2 and 3 at Room 109, Computer Science Building, Bowie State University

Visibility of exits also affects the level of stress and panic during emergency evacuation. In studies 2 and 3, as shown in Figure 46, stress level and panic level was mainly low and medium before the drill, then it slightly increased during the drill and decreased after the drill. These parameters of stress and panic were represented and observed in the intelligent agent goal finding application. A slightly high stress level before, during and after drills 2 and 3 was noticeable, which was different from drill 1 and could be attributed to the larger number of participants in the second and third drills.

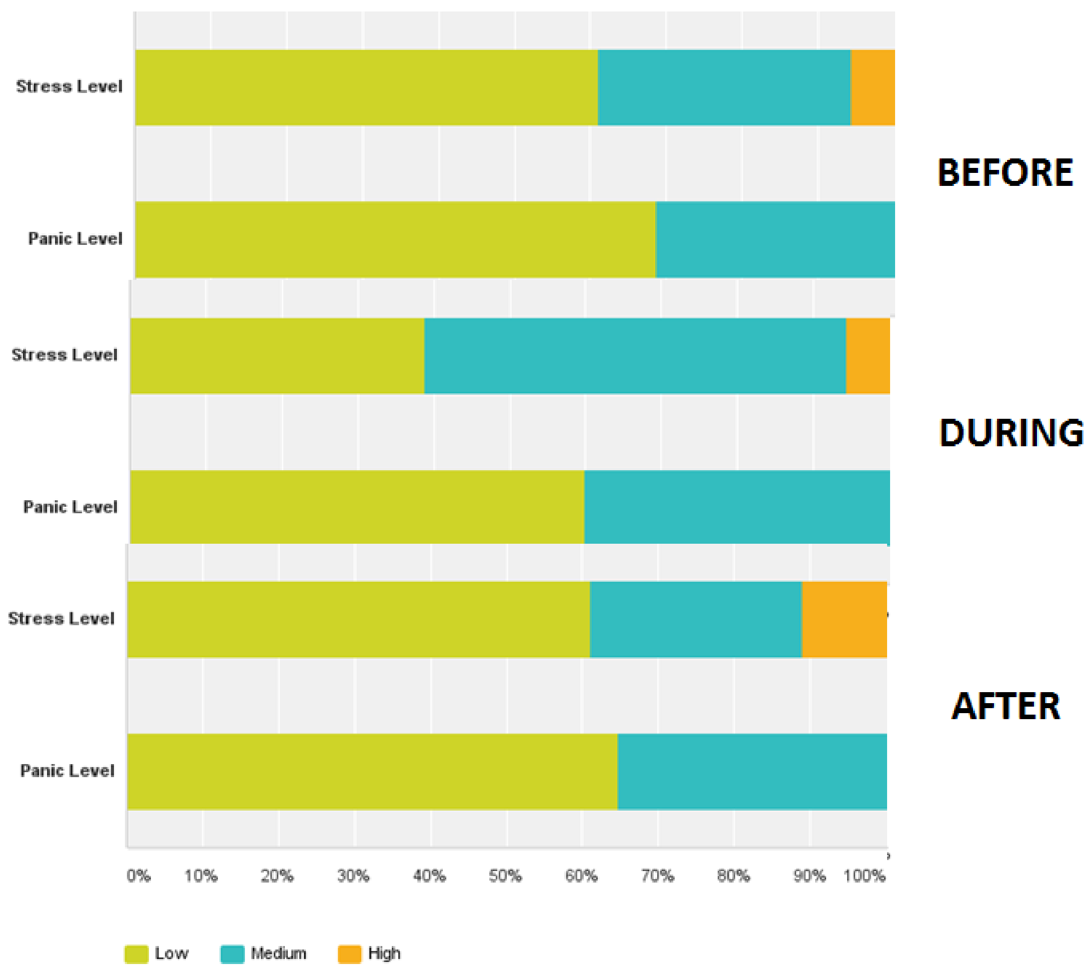


Figure 46: Graph shows stress and panic level before, during and after drills 2 and 3 at Room 109, Computer Science Building, Bowie State University

Stress and panic level was low before the drill and the data showed that it may be related to the results that majority of the participants knew each other previously before the drill as shown in Figure 47. 50% of the participants experienced queuing at the exit while trying to evacuation and Figure 48 showed that since participants knew each other, they were able to patiently wait for their turn and talk to other before finally evacuating the room. This result was the same as seen in the first drill.

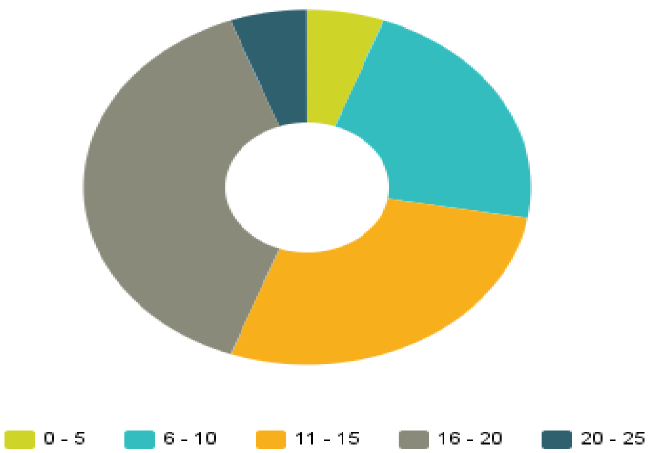


Figure 47: Graph shows number of people previously known before drills 2 and 3 in Room 109, Computer Science Building, Bowie State University

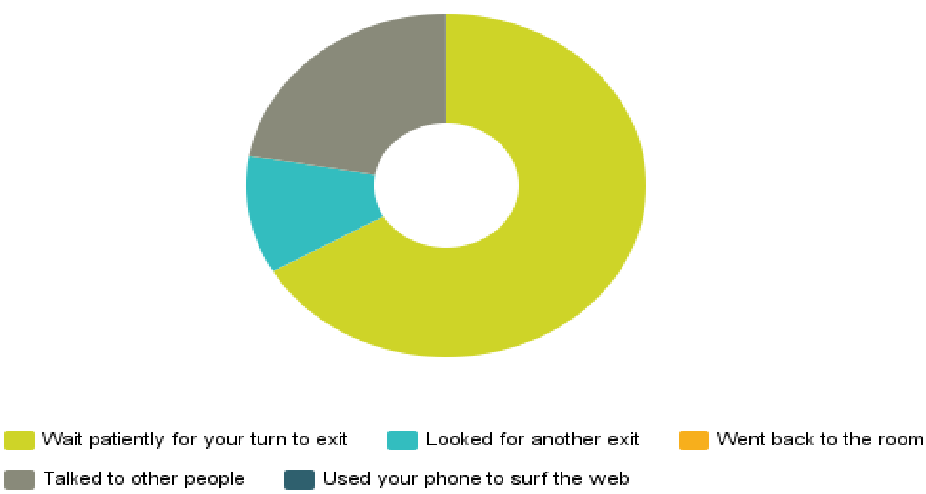


Figure 48: Graph shows what participants did when experiencing queuing at exit during drills 2 and 3 in Room 109, Computer Science Building, Bowie State University

During the evacuation 44.44% of the participants slowed down their speed as they got closer to the exit and majority of them did not go back to get anything they left in the room as shown in Figure 49. It was also seen that the major obstacle getting in the way of participants exiting in the room was other participants, and other table and chairs in the room as shown in Figure 50. Again, these results matched the data points noticed in the first drill.

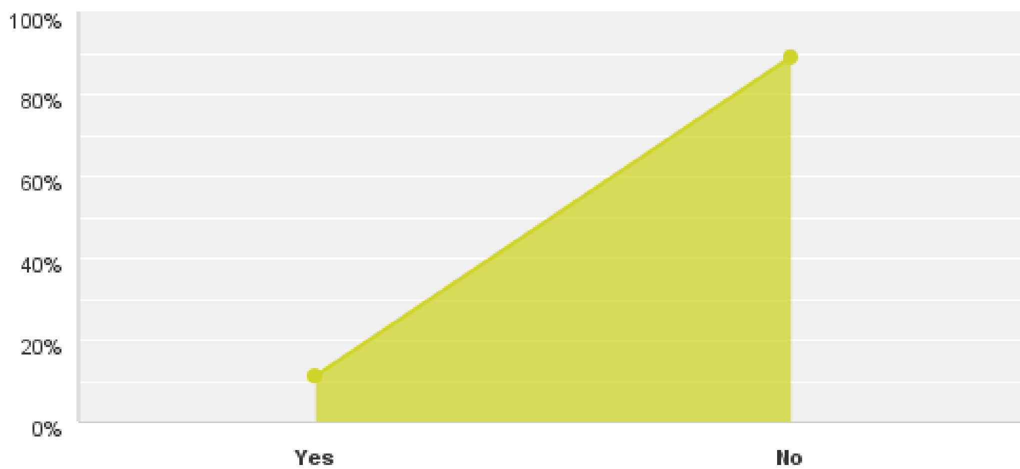


Figure 49: Graph shows percentage of participants who did not go back to get anything left in the room during drills 2 and 3 in Room 109, Computer Science Building, Bowie State University

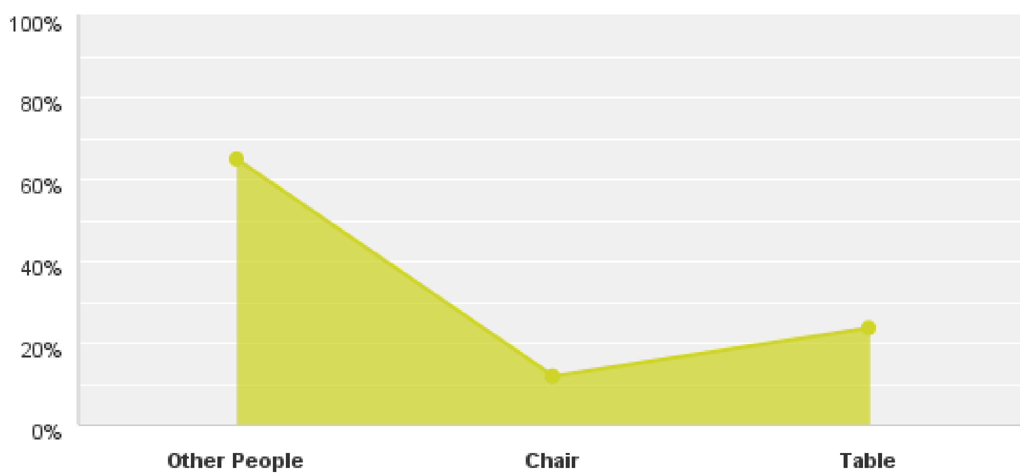


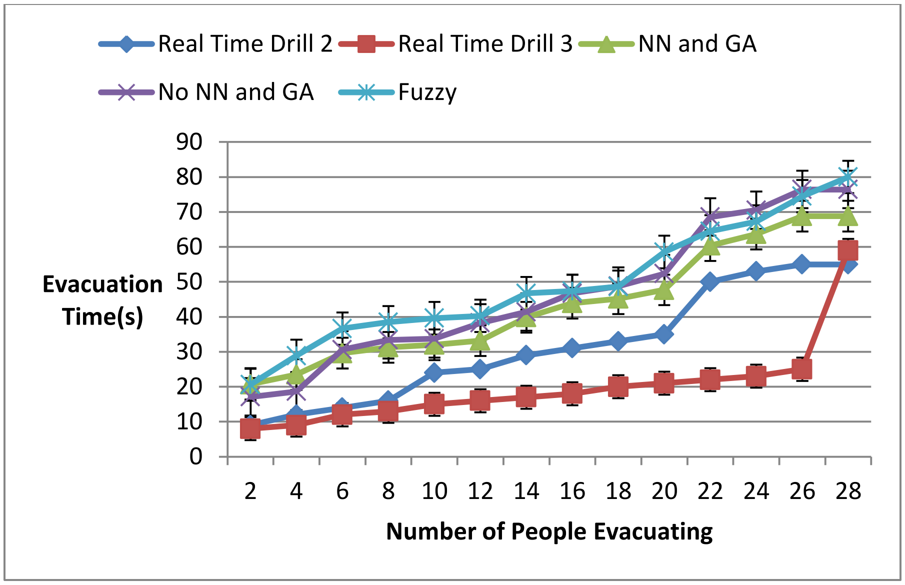
Figure 50: Graph shows obstacles in participant's way during drills 2 and 3 in Room 109, Computer Science Building, Bowie State University

After the second and third drill was completed, we ran the intelligent agent goal application in three sets of 30 runs matching the same parameters seen in the drills. The first set of 30 runs was executed with the neural network and genetic algorithm enabled, second set with neural network and genetic algorithm disabled and the last set with fuzzy panic and stress parameter enabled in the application. The evacuation time was recorded for all the runs as shown in Figure 51.

NN and GA																															
No evac/	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**	**Avera**
2	22	21	23	21	21	21	21	21	22	21	21	21	21	21	21	19	19	19	19	19	19	21	21	21	21	21	21	21	21	21	20.7
4	24	24	23	23	23	24	24	24	24	22	23	23	24	24	24	23	23	24	24	24	23	23	23	24	24	24	23	24	24	23	23.5
6	28	31	31	26	29	29	29	29	31	31	30	30	31	30	30	30	30	30	30	30	31	30	29	29	29	29	29	29	29	29	29.6
8	40	35	32	31	31	32	32	31	31	31	31	31	31	30	31	31	30	30	31	31	31	31	31	31	31	31	31	30	30	30	31.3
10	32	33	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32	31	32	32	32	32	32
12	33	34	33	33	33	34	33	33	33	34	33	33	33	33	34	33	33	33	34	33	33	33	33	34	33	33	34	33	33	33	33.2
14	40	40	40	40	40	40	40	40	39	40	40	40	40	40	40	39	40	40	40	40	40	40	40	40	40	40	40	40	40	40	39.9
16	43	44	44	43	44	45	43	44	44	44	44	44	45	44	44	44	44	44	44	44	44	44	44	44	44	44	44	43	44	44	43.9
18	45	46	45	45	45	45	46	46	45	45	45	46	46	45	45	45	45	45	46	46	45	45	45	45	45	45	45	45	45	45	45.2
20	47	48	48	47	47	48	48	48	48	47	47	48	48	48	48	47	47	48	48	48	48	47	48	48	48	48	48	48	48	48	47.7
22	60	61	60	60	61	60	60	61	60	61	60	60	61	60	60	61	61	60	60	61	60	60	61	61	60	61	60	60	60	60	60.4
24	63	63	64	62	63	63	64	65	64	65	63	63	64	64	65	65	64	64	63	63	64	63	63	63	64	64	64	65	64	64	63.7
26	68	69	68	69	68	69	69	69	69	69	69	68	69	69	69	69	68	69	69	69	69	69	68	69	69	69	69	69	69	69	68.8
Without NN and Ga																															
No evac/	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**	**Avera**
2	18	17	17	17	17	17	17	18	17	17	17	17	17	17	18	17	17	17	17	17	17	17	17	17	17	17	17	17	17	17	17.1
4	19	18	19	19	18	19	19	18	18	18	19	19	19	18	19	18	18	19	19	18	19	19	18	19	19	19	18	19	19	19	18.6
6	31	30	31	31	31	30	30	31	30	30	30	31	31	30	30	30	31	31	31	31	31	31	30	30	31	30	31	31	31	31	30.6
8	33	32	34	34	33	33	34	34	33	34	34	33	33	34	33	34	33	34	33	33	33	34	33	33	34	33	34	34	33	33	33.4
10	34	34	33	34	34	34	34	33	34	34	34	34	33	34	34	34	33	34	34	33	33	34	34	33	33	34	34	33	34	34	33.7
12	40	38	39	38	38	38	37	38	38	39	38	38	38	39	38	38	38	39	38	38	39	38	38	38	38	39	38	38	38	38	38.2
14	42	41	42	42	41	41	41	41	41	42	41	41	42	42	41	41	41	42	41	41	42	41	41	42	41	41	42	42	41	42	41.4
16	47	46	47	47	46	47	47	46	47	47	47	46	47	47	47	46	47	47	47	46	47	47	46	47	47	47	46	47	47	47	46.7
18	49	48	49	48	49	49	49	48	49	49	49	49	48	49	49	48	49	49	49	48	49	49	49	49	49	49	49	48	49	49	48.8
20	52	53	53	52	52	53	53	52	53	52	53	53	52	52	52	52	52	52	53	52	52	52	52	52	53	53	52	52	52	52	52.3
22	69	68	69	68	69	68	68	69	68	69	69	68	68	69	69	68	68	68	69	69	68	69	69	69	68	69	69	69	68	69	68.6
24	71	70	70	72	71	71	70	70	71	70	70	71	70	71	71	70	70	71	70	70	71	70	70	71	70	70	71	71	70	71	70.5
26	75	76	75	76	77	76	77	75	76	76	77	75	76	77	75	77	77	77	77	77	77	77	76	77	77	77	76	77	77	77	76.4
Fuzzy																															
No evac/	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**	**Avera**
2	20	21	22	20	21	20	21	20	21	20	21	20	22	23	20	20	20	20	20	21	20	21	22	20	20	20	21	21	21	20	20.6
4	28	29	30	28	28	28	29	29	30	30	29	28	28	29	30	30	28	28	29	30	28	28	29	30	28	28	29	30	30	28	28.9
6	36	37	38	36	36	36	37	36	38	36	37	37	37	38	36	36	36	36	37	37	36	36	36	37	37	38	38	36	36	36	36.6
8	38	39	39	38	39	38	39	38	39	38	38	39	38	38	39	38	39	38	39	38	39	38	38	39	38	39	38	39	39	38	38.5
10	39	39	38	39	39	38	39	38	39	49	40	39	39	39	40	39	39	40	39	40	40	41	40	39	40	40	39	39	40	39	39.6
12	40	41	40	40	40	41	42	40	40	40	41	41	40	40	40	41	40	40	40	40	40	40	40	40	40	40	40	40	40	41	40.3
14	45	46	45	45	45	46	47	45	45	46	47	46	45	48	48	48	48	48	46	47	47	48	48	47	47	48	48	47	48	48	46.7
16	47	46	47	47	47	48	48	47	47	48	48	48	47	47	47	47	48	48	48	47	47	48	48	48	47	47	47	47	47	48	47.4
18	49	48	48	49	49	48	49	49	48	49	49	48	49	49	48	48	49	49	48	49	49	48	49	49	48	49	49	48	49	49	48.6
20	58	59	60	58	58	59	60	58	58	58	59	59	58	58	59	57	58	58	59	59	58	59	58	59	58	58	59	58	59	60	58.5
22	65	64	64	65	64	64	65	65	64	64	64	65	65	64	64	64	65	65	64	64	65	65	64	64	65	65	65	65	64	64	64.5
24	66	67	66	67	66	68	68	67	68	66	66	68	67	68	66	68	67	67	68	67	68	67	68	68	67	68	68	67	68	68	67.3
26	70	73	75	70	76	75	77	75	75	76	75	76	75	75	75	74	75	75	76	74	75	75	74	74	74	74	74	74	74	75	74.5

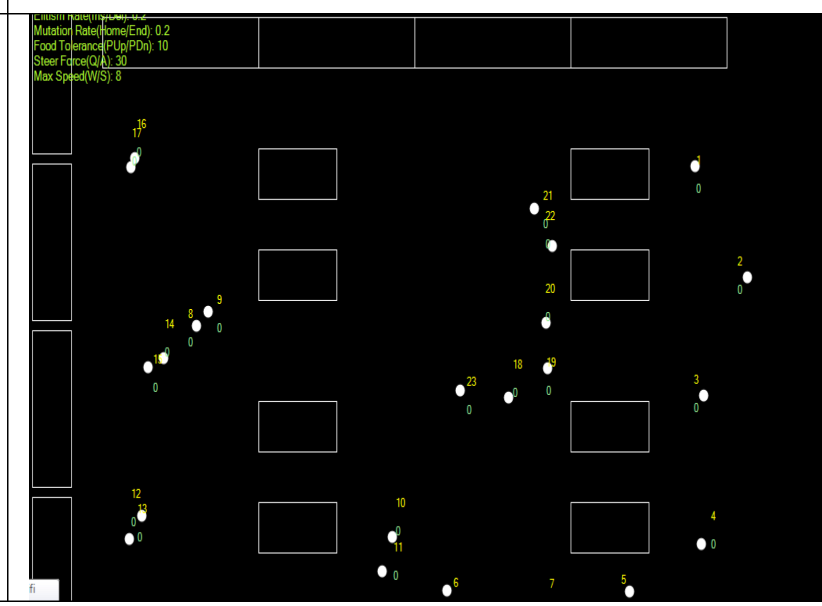
Figure 51: Table shows evacuation time generated from intelligent agent goal app using same parameters from drills 2 and 3 in Room 109, Computer Science Building, Bowie State University

We implemented the simulation numerous times for the aggregate evacuation time by increasing the number of people evacuating up to maximum of participants seen in the real time drill. We increased the number of participants evacuating in the interval of 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22 and 24. Thus, in total we performed the simulation 90 times to get a more minutiae curve for stress and panic, which integrates average and standard deviation. The error bars on the chart represent the standard deviation in time for percentage of participants evacuating. The points on the chart represent the average time for each interval as shown in Figure 52. The graphs show the variation of total evacuation time when the percentage of participants evacuating was increased. Overall, the curve of Fuzzy NN and GA is going uphill the highest, which displays that when more parameters of stress and panic are added, there is a slight increase in the evacuation time. According to the model, wait time of participants increases when panic and stress is high. Participants become unpredictable when it comes to making decisions in emergencies. As a result, their behavior becomes erratic. At each interval, we took average time of evacuation. The error bar on bar graphs shows the difference in average time with 95% confidence level. For 23 people evacuated from Room 109, we see that it took an average time of 75 for Fuzzy, 76.4 for No NN and GA and 68.8 for NN and GA with 30 simulation runs done. Table VIII shows screen shot of agents evacuating the intelligent goal finding application for study 2. The behavior of the agent is in line with the dynamics that plays out in the interaction of crowd agents when trying to avoid obstacles as mentioned earlier is that everyone tries their best to evacuate in an orderly manner to avoid panic (Tran, 2013).



<u>Figure 52:</u> Graphs shows evacuation time comparison Real-Time Drills 2 and 3 versus NN and GA versus no NN and GA versus Fuzzy

Table VIII: Screen shots for agent evacuating intelligent goal finding application Study 2

Time Elapsed in seconds	Input Variables	Screen Shot
20	agentCircles=28 NumAgentsExited=2	

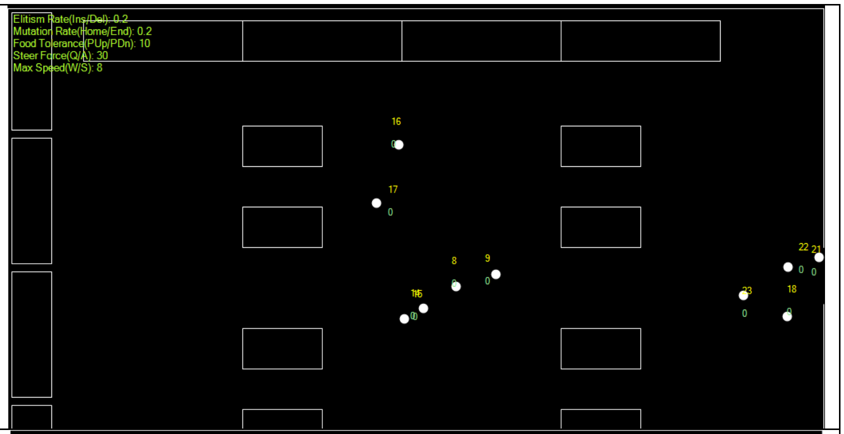
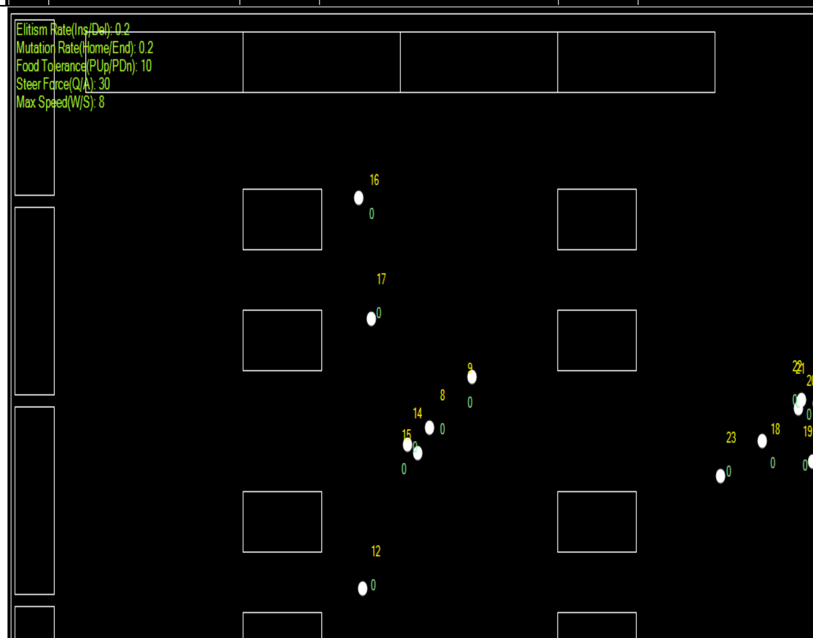
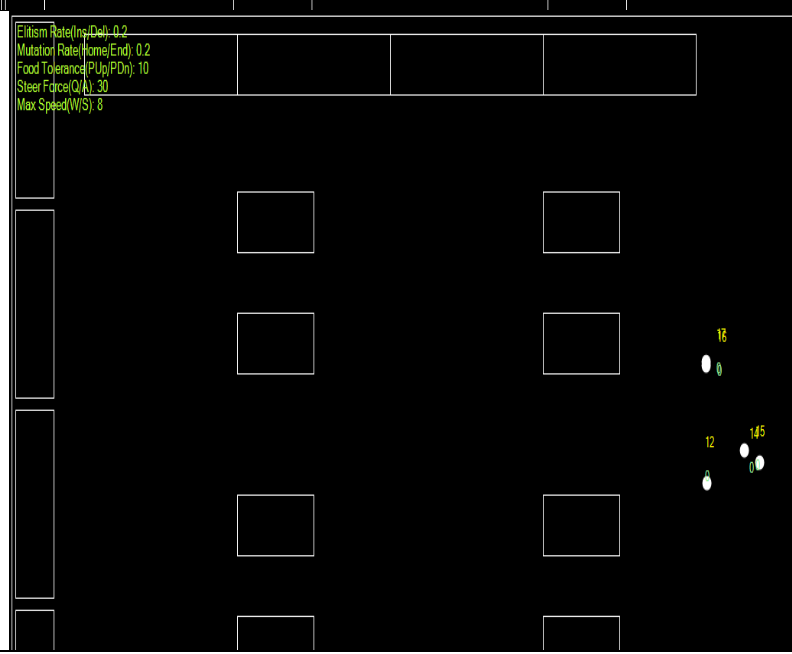
23	agentCircles=28 NumAgentsExited=4	
32	agentCircles=28 NumAgentsExited=12	
40	agentCircles=28 NumAgentsExited=14	

Figure 53 shows the wide angle image of the real-time computer lab where evacuation drills was performed in the computer lab and its matching model in intelligent goal finding evacuation application in Table VIII. Tables and chairs behave as static obstacles and other people in the room as dynamic obstacles. Figure 54 is the image of the computer lab from the right exit and Figure 55 is the image from the left exit. The input file of the intelligent goal finding application shown in Table IV is used to enter in the parameters such as the number and position of agents, number and position of exits, and number and position of obstacles. The intelligent goal finding application environment is scaled to match the actual dimensions of the computer lab. We also entered in the levels of stress and panic of the agents and also the speed of the agent as a constant value.

Figure 53: Real-time computer lab where the evacuation drill was performed.

Figure 54: Image from right exit of where the evacuation drill was performed.

Figure 55: Image from left exit of where the evacuation drill was performed.

Comparison of Implemented Model with Real-Time Evacuation and Existing System

For preliminary validation of our system, we did a set of small, proof-of concept style experiments modeling an actual room used as a computer lab in our department building. Our implemented application for emergency evacuation was written in Microsoft C-Sharp (C#). Learning and adaptive behavior of agents by using GA & NN to perform automated testing in evacuation simulation. The functionality would consist in the user designing of a room, and then having the program run automatically a set of tests, varying the placement and number of agents, obstacles and exits. After a defined set of tests have taken place, the program would use the collected statistics to automatically determine optimal placement for fire exits, doors, etc.

Our implemented C# application is shown in Figure 33 as generation and current best. The length of a generation is the time limit of 90 seconds after which the numbers of agents that have successfully evacuated are recorded in the current best. Walls and other obstacles in the application are shown as rectangles. Doors are represented as gaps in the wall. The parameters for drawing the walls are passed to the simulation through an input text file specified by the user of the application.

A. MLP Algorithm GA & NN

An AI-controlled character makes use of 19 input values: the distance to the nearest 5 hostile agents, distance to the nearest 4 non hostile agents with their distance to nearest exit and direction values, and the distance to nearest exit and direction of the AI. We assume 5 different output behaviors agents can exhibit on their way to an exit: flee, seek, arrive, and wander and cohesion as shown in Figure 33. We have used a network with three layers: input layer and output layer, plus an intermediate (hidden) layer. The input layer has the same number of nodes

19. The output layer has the same number of nodes as there are possible outputs: 5. Hidden layers will have at least as large as the input layer and will be 24. Figure 33 show the simulation screens during agent evacuation.

Steps: to begin, all the weights in the network (refer to Figure 28) were set to small random values and a set of iterations of the learning algorithm was done by selecting an example scenario from the training set. We then took the input's fed forward to guess the output and changed the network (back propagation) by comparing the expected output and the guess. Every 10 to15 seconds, a GA roulette wheel selection process was used to select the two parents that would be used to change the weight of children. After iterations were done, we checked to see if learning was done by running on the test group of examples. If the guess output met our expectation, we were sure NN had learned properly or else we ram more training on the network.

Figure 33 depicted a lecture hall where there were 23 agents. Each agent's behavioral characteristics as well as group characteristics were defined through the input file. Agents in the C# application were drawn as circles in the simulation. The application had one simulation mode where agents steered themselves so they did not pass through other agents.

B. Evaluation

The intelligent agent application for emergency evacuation was written in Microsoft C-Sharp. It served a similar purpose as Pathfinder 2014 developed by Thunderhead Engineering for agent based egress modeling. The intelligent and Pathfinder applications both had a graphical user interface for simulation and design but intelligent was capable of 2D visualization for result analysis while Pathfinder has 2D and 3D (Figures 33 and 56). Output of the intelligent application is shown to the right in Figure 2 below as generation and current best. The length of a generation is 90 seconds after which the agents are counted that have successfully evacuated and are recorded in the current best. The movement environment in the intelligent application is a 2D rectangle (Figure 33) representing the dimensions of the room. Dimensions of the room are represented on the screen with a rectangle width 1015 and height 695 pixels. The total number of people modeled in the test simulation is about 45.

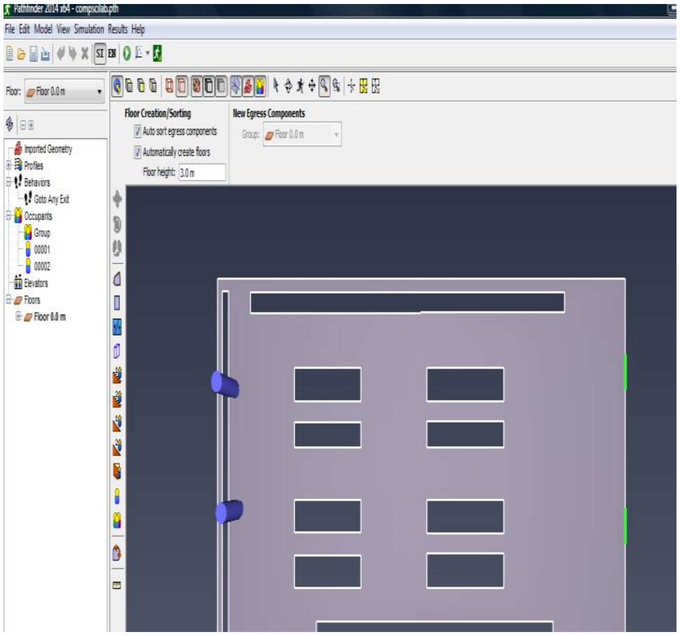
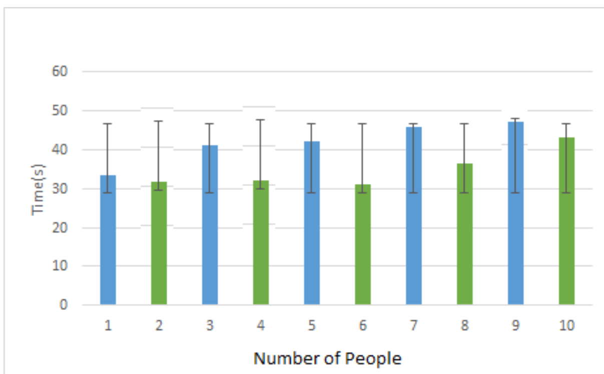


Figure 56: Pathfinder Graphical User Interface

Walls and other obstacles in the intelligent application are shown as rectangles that cannot be passed through by the agents. Doors are represented as gaps in the wall. The parameters for drawing the walls are passed to the intelligent simulation through an input text file specified by the user of the application. Pathfinder represents the movement environment with a surrounding mesh resembling the building. Gaps in the surrounding mesh displays the walls and areas that cannot be passed by the occupants. Doors are represented by unique edges in the navigation mesh. Stairways and elevators can also be represented in the pathfinder application. The flexibility in the movement simulation is combined to provide a dominant simulation device with malleable switch over inhabitants and performance to bring improved consequences. Pathfinder can import AutoCAD format DXF and DWG files to swiftly use the introduced geometry to describe the inhabitant walking cosmos for the evacuation model. PyroSim or Fire Dynamics Simulator (FDS) models can also be used to abstract the walking universe. If you have a blueprint, it can be introduced in GIF, JPG, or PNG format and then used as a contextual to help you speedily draw your model directly over the image. Triangulation also assists constant drive of folks throughout the typical environment, likened to other simulators that split the cosmos into cells that can exaggeratedly constrain the movement of occupants. Pathfinder cares for two replication approaches. In Piloting mode, agents proceed independently to their objective, while dodging other tenants and difficulties. Door flow rates are not specified but result from the interaction of occupants with each other and with boundaries. In SFPE mode, agents use behaviors that follow SFPE guidelines, with density-dependent walking speeds and flow limits to doors. SFPE results provide a useful standard for evaluation with other outcomes, but SFPE calculations do not prevent many individuals subjugating the similar space. Optionally, Pathfinder approves to decide door drive amounts in steering method to gain greater conception in a controlled ideal. You can effortlessly control among methods in the Pathfinder user interface. By default, each inhabitant (agent) produces a mixture of constraints to hand-pick their present pathway to an exit. The constraints include: file intervals for each entry of the present area, the period to travel to each gate of the current room, the projected interval from each gate to the exit, and the space already covered in the area. The agent replies vigorously to altering logjams, gate lead-ins/ends, and variations in area promptness restrictions. The user can modify the default parameter weights to change the behavior. For example, occupants can neglect queues and only look for the closest exit.

C. Result

Validation was done by comparing the evacuation time when the simulation was completed on the intelligent agent application and pathfinder application. Agents in the intelligent application are drawn as circles in the simulation while occupants in pathfinder are shown as upright cylinders on the movement mesh. The position and speed of both agents and occupants in intelligent and pathfinder are both specified in the simulation applications. Intelligent application has one simulation mode where agents steer themselves so they do not pass through other agents. Pathfinder has steer mode and also the SFPE mode where occupants are allowed to go through other occupants.



<u>Figure 57:</u> Evacuation Time Error Bar comparison of intelligent and pathfinder application

Our implemented C# evacuation application and the pathfinder application both had similar functionality and performance that allowed them to be effective tools for data visualization in emergency evacuation. Figure 57 shows the error bar of the comparison seen when running the intelligent agent application (blue bar) and the pathfinder application (green bar). The simulation involved 25, 35, 45, 65, 75 occupants similarly placed in positions in the room and timed to see how fast they could evacuate the room. Bar graph 1 and 2 represents the simulation running with 25 people in the evacuation and path finder applications respectively. The average time was 33 seconds for evacuation application and 25 seconds for path finder application. Bar graph 3 and 4 represents the simulation running with 35 people in the evacuation and path finder applications respectively. The average time was 41 seconds for evacuation application and 27 seconds for path finder application. Bar graph 5 and 6 represents the simulation running with 45 people in the evacuation and path finder applications respectively. The average time was 42 seconds for evacuation application and 31 seconds for path finder application. Bar graph 7 and 8 represents the simulation running with 65 people in the evacuation and path finder applications respectively. The average time was 46 seconds for evacuation application and 36.5 seconds for path finder application. Bar graph 9 and 10 represents the simulation running with 75 people in the evacuation and path finder applications respectively. The average time was 53 seconds for evacuation application and 43 seconds for path finder application.

Research Questions and Hypotheses

As noted earlier, there were three research questions and hypotheses pointed out during the current study. This section explains the results of the research questions and hypotheses.

Hypothesis 1

Hypothesis 1: There will be a positive correlation between the faster evacuation time and smaller size of occupants was confirmed in the study.

Inverse of Hypothesis 1 was rejected: There will not be a positive correlation between the faster evacuation time and smaller size of occupants.

Hypothesis 2

Hypothesis 2: There will be a positive correlation between the faster evacuation time and the number of runs was not confirmed. Generally, it was seen that the evacuation time remained between a small ranges across the multiple runs of the application.

Inverse of Hypothesis 2 was accepted: There will not be a positive correlation between the faster evacuation time and the number of runs.

Hypothesis 3

Hypothesis 3: There will be a positive correlation between faster evacuation time and the type of behavior exhibited by occupant was confirmed. This was confirmed that as we added more behaviors such as stress and panic, the delay time of participants evacuating increased in the application.

Inverse of Hypothesis 3 was rejected: There will not be a positive correlation between faster evacuation time and the type of behavior exhibited by occupant.

Summary

Chapter 4 provided a detailed analysis of the results found in the current study. The demographic, expressive, measurable, and qualitative questions and report were described and displayed throughout the chapter. Positive correlations were found between evacuation time and size of participant evacuating and type of behaviors exhibited by participants. Thus accepting the two original hypotheses and rejecting one. However, no correlation was found between evacuation time and the number of runs of the applications and the original hypothesis was rejected. The next chapter discusses the conclusion, recommendations and future work.

CHAPTER 5

SUMMARY AND CONCLUSIONS

Conclusions

The offered simulation model—Intelligent goal finding application studies how agents are able to move around static and dynamic obstacles and change their velocity on the way to reaching their goals. The prototype's steering procedure is centered on commuters nearest goal (exit selection) while avoiding obstacles in its way. An obstacle avoidance and nearest exit procedure is established to permit individuals to efficiently route around hindrances (refer to Chapter 3). In Chapter 4, we liken nearest goal algorithm with neural network and genetic algorithm and three real-time evacuation drills. The outcomes demonstrate that neural network and genetic algorithm completes well than nearest goal algorithm when mortal steering is involved. The outcomes shown in the chapter are constant by way of our hypothesis that the type of behaviors exhibited by people will affect their evacuation time during emergencies. Different from machines, individuals do not continuously practice the nearest exit. Consequently, we can determine that neural network and genetic algorithm accomplishes faster evacuation time than nearest goal algorithm during people evacuation.

This research defines in what way the simulation prototype—Intelligent goal finding application can incorporate together neural networks, artificial intelligence and fuzzy reasoning constraints. We used fuzzy reasoning for filtering the agent interactive prototype by combination of fuzzy constraints in a multiple agent setting. Contingent on the responsive state of each agent, the velocity and delay time exhibited by the agent will cause the rate at which they move towards the closest exit to be different for each agent. These constraints are interconnected individually and the fuzzy result is contingent on the value placed respective distinct constraint (refer to Figure 28). Chapter 4 associates the ability of intelligent goal finding application with the non-genetic algorithm, neural network and fuzzy constraints (i.e., when the genetic algorithm, neural network and fuzzy constraints were not added). In addition, we matched the outcomes with the three real-time evacuation drills examples and found that the incorporation of genetic algorithm and neural network does make a difference in the behavior of the prototype (refer to Figure 42). The outcomes indicated that the behavior of genetic algorithm and neural network is nearer to the three real-time evacuation drills when compared to non-genetic algorithm and neural network intelligent goal finding application.

Additionally, we believe that the application of intelligent goal finding application environment can be changed to simulate other types of layout, such as combat zone situation and aircraft flight situation. In the combat zone situation, the agents in the intelligent agent goal application will be replaced by subdivisions of fighters and the exit can be the adversary encampment. The input file of the application can be automated with parameters from simulations that can be extracted by an examination of withdrawal techniques and by interpretations of social conduct during a real-time emergency drill. The likelihood of unsafe occasions can be verified by orientation to former calamity accounts and real-time evacuation drills can be acquired through using simulations generated by a computer.

Recommendations

Multiple agent situation application creation has appealed to many investigation technologists located around the world. Complexity and generalization are continually preferred and supportive in disaster situations. After there are fuzzy constraints like stress and panic added, the fuzzy and genetic algorithm and neural network approach for estimating of likelihood is the most favorable method. The suggested fuzzy adept application mimics an individual performance method to forecast the likelihood of agents when fuzzy conduct features are seen (refer to Figure 42). Owing to the landscape of genetic algorithm, neural network and fuzzy parameters in the resolution offered in this thesis is just one out of the several total of resolutions. The strategy and example offered is applied in C-Sharp programming language and displayed as an executable form in a windows program that runs on a computer that has the .net framework installed on it.

Based on our experimentations, we determine that video camera recording is an operational device for getting footage of single area exits in an evacuation. We had the capability to identify which exit each individual left through; including the time they left, with decent consistency (refer to Chapter 3). The outcomes were collected mechanically by the application. Due to the proficiency, and since accumulating more users involves little supplementary price or interval, we assume this method to be very advantageous for extensive evacuations, such as an office building with many floors or an arena. Our video camera recoding method also documented exit times, which are advantageous for rapidly and precisely computing the movement of individuals through the exits, and reckoning out the time it took individuals to find their way out of the building. We also had the capability to reliably and competently find out which participant exited through which door. We discovered that by integrating investigational information into the intelligent goal finding application, the evacuation time was reasonably similar for the simulation rounds.

The results from the three evacuation drills, using video cameras experiment at the

Computer Science Building at Bowie State University, showed a slight noticeable change in the time for the path to exits between the first, second and third drills, which had a large number of participants. The change between simulation and the real-time evacuation drills was established to be very minor and movement speed was persistent in both cases (refer to Chapter 3). The social features of an agent, such as quickness, delay time, stress, panic, choice of exit, can be attuned to mimic a real-time situation. We trust that after the social parameters of an agent are attuned for one situation to portray an individual, the same conduct can be projected into a different situation.

Our proposed C# application can be used to model situations that are difficult to test in real life due to safety considerations. It is able to include agent characteristics and behaviors. The findings of this modeling are very encouraging as the agents are able to assume various roles to combine GA and NN on the way to reaching their goals. We hope our proposed tool will help in visualizing evacuation time and what-if scenarios for environments that are difficult to model in real life. It will also act as training and educational tools for depicting different evacuation strategies and damage control decisions during evacuation. The proposed application will aid in running multiple evacuation drills for what-if scenarios by incorporating agent characteristics. The future work will involve the revisions of exercises of an agent to signify an individual's social features, executing many drills in many situations and implementation of altruistic behavior and selfish behaviors.

Future Work

Multilayer Perceptrons (MLP) explored in this research contain hidden layers that increase the area of theories that the network can symbolize, which means the number of problems it can solve is very large. (Russell & Norvig, 2003). Individual hidden neurons signify a perceptron that points to a lenient threshold function within the input space. The output neuron derives its value from leniently adding together the various threshold functions together from the input space. When we adjust the weights in MLPs, we change the function in the network and learning, such as classification or regression is achieved. Its structure and training is well developed and has a good generalization capability. A lot more remains to be discovered to incorporate some of the advantages from the other types of neural networks shown in the Table IX. For example, sufficient training data needs to be generated to be able to use MLP effectively in a simulation. The use of ambiguous reasoning in discovering multiple agent situations can add to the insertion of multiple agent enhancements, when things are not so clear, and the practice of vague data recovery methods for adaptive education. Considerably additional work needs to be done in finding out problems encountered during ambiguity of autonomous agents. Upcoming effort shall comprise improvement of additional ambiguous actions and producing more challenging outcomes and execution of genetic algorithm, neural network and fuzzy behavior.

Forthcoming work will be dedicated to building up the application's ability to adapt to new actions, while the simulation is running. The ability to acquire knowledge by itself is an ability that would be encouraged and fine-tuning of the procedures and connection utilities for the adaptive actions might contribute stimulating outcomes.

Table IX: Comparisons of Various Types of Neural Networks

Neural Network Models	MLP	Radial Basis Function (RBF)	Wavelet	Arbitrary Structures
Advantages	1. It is a fast and accurate model of original problem (continuous & integral functions) it learned after training. 2. It distributes the work of learning with the combination of many neurons responding to many external inputs, which turns it off, on, or in transition state.	1. It provides better initial values for hidden neuron centers using unsupervised training sample distribution compared to random weight. 2. Learns at faster rates and shows reduced sensitivity to the order of presentation of data. It makes them good for problems with small number of inputs. It works better when training data is large and sufficient.	1. It has a wavelet reduction algorithm that can selectively choose wavelets that best fit the training data. 2. Network weight parameters can be refined using supervised training combined with wavelets functions that match data output	1. Flexible structure not layered so all neurons can be connected to all other neurons. 2. External output and input can be applied to and from any predefined set of neurons
Disadvantages	1. Sufficient training data is required which could	1. When the amount of training data becomes	1. Unnecessary large number of initial wavelets can be created in	1. Complexity increases between the links

	be large for high dimensional problems 2. Too many or too little hidden neurons, which may lead to overlearning or under learning respectively.	minimized, it degrades faster compared to MLP. 2. It does not have a better generalization capability compared to MLP.	a lattice approach. This leads to redundancy. 2. Performance degradation may occur because of a large number of hidden neurons affecting training.	among the various neurons. 2. Have to manage optimizatio n of add/delete neurons and connections between training.

Table IX: Comparisons of Various Types of Neural Networks (continued)

Neural Network Models	**Self-Organizing Maps (SOM)**	**Recurrent**
Advantages	1. Facilitates automatic decomposition through processing and learning of training data. 2. It is useful in the case where problem is complicated and precise shape of subspace boundaries are not easy to determine	1. It allows time-domain behaviors of a dynamic system to be modeled.
Disadvantages	1. Sufficient training data is required similar to a clustering algorithm. 2. It can be dependent on MLP to provide input.	1. It depends on the history of system states and inputs, and thereby a mechanism must be available to capture and save the history.

Future work will comprise of the usage of 3D applications such as Microsoft XNA using triangles or pixels finding their way to a goal in a 3D space. Examination of 3D environment will make the prototype more in line with the real world and to forecast link of vision to stop agents from running into each other in the environment. Feasibly estimating the route using likelihoods for mimicking agent centered actions and the situation where agents jump find their way in the environment by using 3D XNA and its physics libraries. Figure 58 displays the promising addition of the intelligent goal finding application to XN

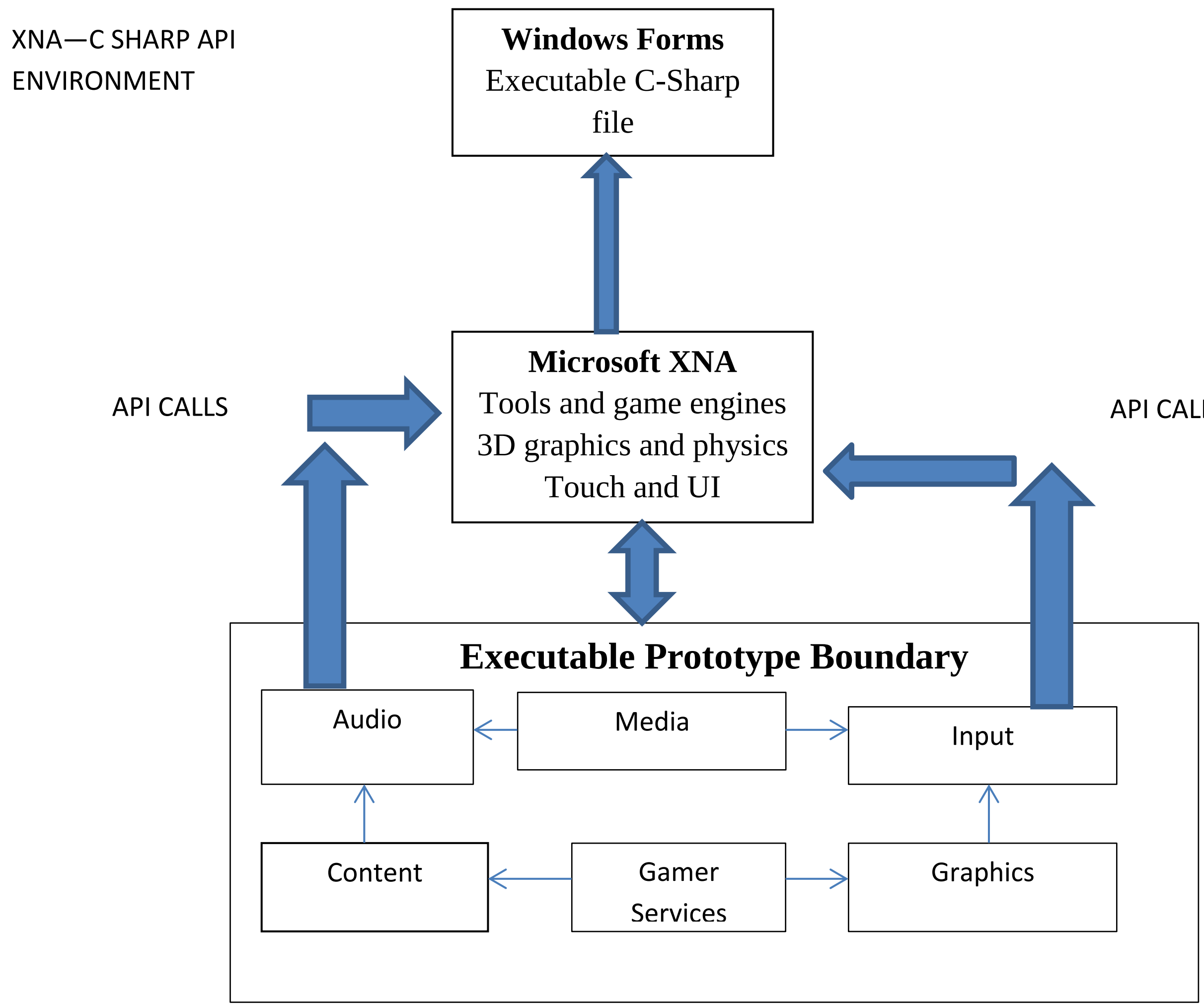


Figure 57: Extension of Intelligent Goal Finding Application to Microsoft XNA

The use of 3D graphics and physics in a simulation involves communication between Microsoft XNA tools and game engines and the C-Sharp .net executable. This communication can be performed by making calls to the Application Program Interface (API). The API is a group of procedures, conventions, and apparatuses for construction of software applications. The API stipulates by what method software modules ought to work together, and APIs are used when developing software for graphical user interface (GUI) modules. The contents that can be manipulated by the API calls to XNA include block images as obstacles, door images as exits and person images as agents in the simulation. Each of these objects can be rendered as a 3D texture and placed in a vector position in the environment. A sprite batch can be used to draw and visualize the texture, using the game engine graphics device. Obstacle avoidance can be done in the visual interface by calculating the vector to steer the agent away from the obstacle using certain thresholds, differentials and normalizing adjustment values. It is also possible to use keyboard state to manually control the up, down, left and right movement of an agent in the simulation. This can be a way for future work by a researcher interested in extending the simulator.

Actual records deliver valuable understanding and numerical data that will provide us the prospect to figure out numerous fascinating methodical matters, regarding the construction and application of people performance simulations. Numerous diverse expertise have been investigated to spontaneously find persons and items for the Computer Science building at Bowie State University, Maryland. Video camera recording is one of the tools investigated with, and it was discovered that it is effective in locating specific persons as they pass edges, such as the doors and exits to places or houses. Gathering actual records will be of enormous assistance to somebody who is fresh to the space and is forecasting to do additional examination.

Future work shall comprise expansion of additional activation functions, such as those in Figure 59 in the neural network simulation. This may produce more analysis outcomes and application of genetic algorithm and neural networks actions. The additional examination with genetic algorithm can also provide favorable consequences. Future work will be concentrated on creating the application's actions to be dynamic in the model of social agents. Autonomous agents and modification in the procedures and association utilities for the dynamic behavior might influence stimulating outcomes.

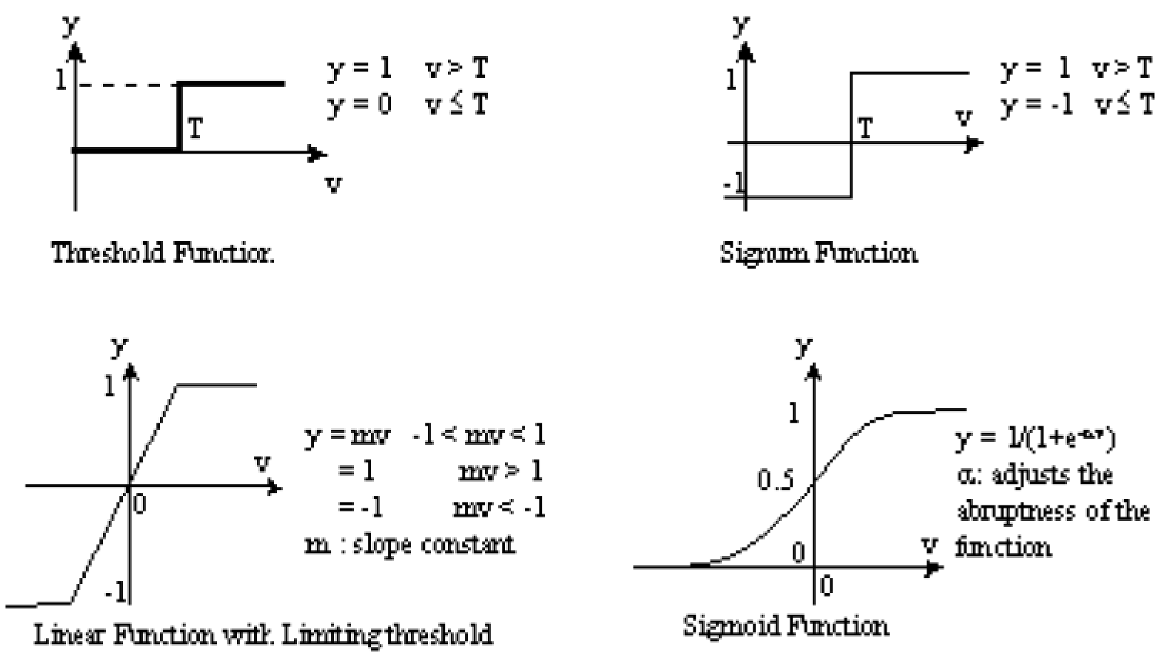


Figure 58: Activation Functions for Neural Networks

APPENDICES

Appendix A - Definition of Terms

1. **Genetic Algorithm (GA):** is a type of random variable or probability based search in which descendant states are created by bringing two parent states together (Russell & Norvig, 2003).
2. **Neural Network (NN):** is a reproduction of the brain's ability to attain, document, categorize and retrieve, display and broadcast information (Russell & Norvig, 2003).
3. **Multi-agent systems**: decipher composite dilemmas in a dispersed method without the need for more than one agent to know about the total problem being solved (Russell & Norvig, 2003).
4. **System:** is an arrangement, organization or a way of classifying a scheme.
5. **Evacuation:** mass departure, migration or flight from a situation.
6. **Simulation:** recreation, replication or mock-up model.
7. **Behavior:** performance, action, deeds, conduct or activities.
8. **Animated:** active, dynamic, vigorous or energetic.
9. **Emotion:** feeling, sentiment or sensation.
10. **Speed:** pace, rate, velocity or momentum.
11. **Direction:** route, path or track.
12. **Goal:** objective, aim, purpose or target.
13. **Location:** place, position or setting.
14. **Anger:** opposite of calm, annoyance or irritation.
15. **Stress:** anxiety, nervous tension, strain or pressure.
16. **Panic:** loss of self-control or being in a calm state.
17. **Weight:** influence, power or credence.
18. **Rule:** statue, law or regulation.
19. **Fuzzy Set:** is a way of to define how fit an object fulfills a clear explanation (Russell & Norvig, 2003).
20. **Model:** representation, reproduction or replica.
21. **Membership:** association, relationship or connection.
22. **Set:** a group
23. **Network:** complex arrangement of association or group.
24. **Algorithm:** step-by-step formula for computation, data processing.
25. **Cognition:** thinking or reasoning done usually by humans.
26. **Agent-Based Modeling and Simulation (ABMS)**
27. **Cognitive Architecture for Perception-Reaction Intelligent Computer Agents (CAPRICA).**
28. **Virtual Cognitive System (VIRCOG)**
29. **Laboratory of Intelligent Interfaces of Communication and Information Systems**

(LIRKIS)

30. **Virtual Environment (VE)**
31. **Virtual Reality (VR)**
32. **Focus of Expansion (FOE):** increase beginning from conversion motions with a point in or outside the retina in an up to date course path (Bruder, Steinicke, & Wieland, Self-motion illusions in immersive virtual reality environments, 2011).
33. **Head Mounted Display(HMD)**
34. **Role-Based Collaboration (RBC):** is an up-and-coming style to aid a managerial organization, provide logical classification activities, and combine structure protection for both individual and non-human entities that work together and organize their actions with or within systems (Ferrari & Zhu, White Cat: Making Agent Roles Perceivable, 2010).
35. **Multi-Agent Systems (MASs)**
36. **Vehicle Routing Problem (VRP)**
37. **Evolutionary Algorithms (EAs):** random seeking methods that imitate evolutionary processes seen in nature (Thangaraj, Pant, Chelliah, & Abraham, 2012).
38. **Opposition Based Chaotic Differential Evolution (OBCDE)**
39. **Opposition Based Learning (OBL) Rules**
40. **Belief Desire Intention (BDI):** is a pattern appropriate for representation of the cognitive method of agents on behalf of humans in an agent based replica reproduction (Scerri, Hickmott, & Padgham, 2012).
41. **Reverse Turing Test (RTT), Human Interaction Proof (HIP), Automated Turing Test (ATT), Completely Automated Public Turing test to tell Computers and Humans Apart (CAPTCHA):** computerized analysis or process of finding out difference between people and computers (McIntire, Havig, & McIntire, 2009).
42. **Social Learning Theory (SLT):** learning that happens within a group environment (Penaloza, Mae, Ohara, & Arai, 2012).
43. **Macro-Agent Evolutionary Model (MacroAEM)** (Jing, Zhong, & Jiao, 2009).
44. **Cosmos:** is a world or environment in which an agent has a perception through it sensors and acting upon it using actuators (Russell & Norvig, 2003).

Appendix B - Fuzzy Logic Model—Panic Behavior

Agent Controller

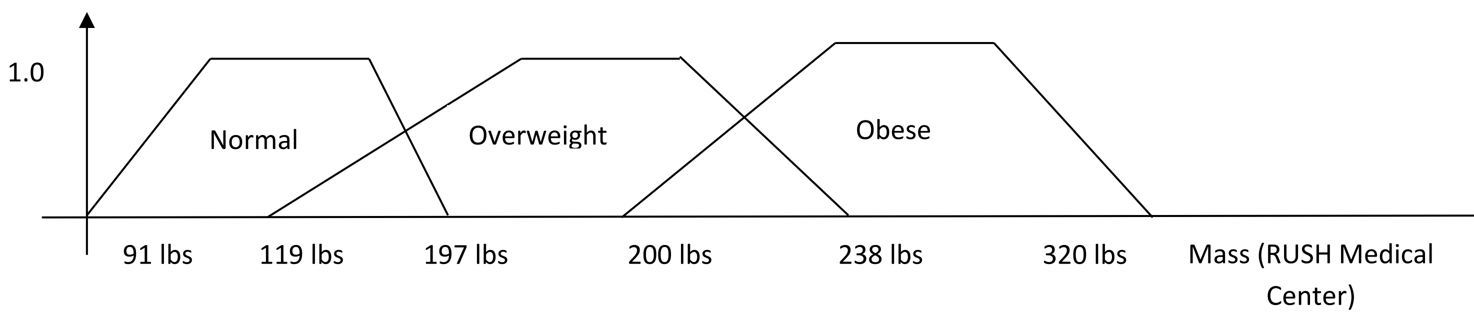
4 inputs:

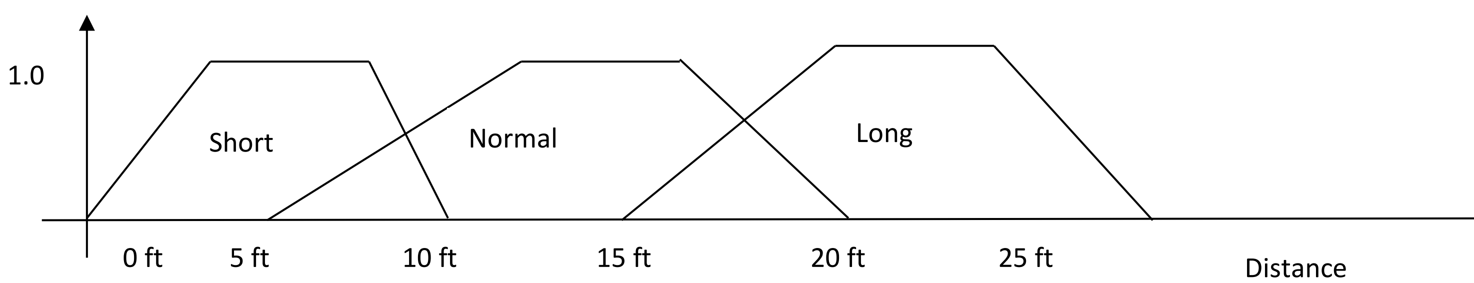
Mass of agent (3 levels)
Distance from current location to closest exit (3 levels)
Stress (3 levels)
Panic (3 levels)

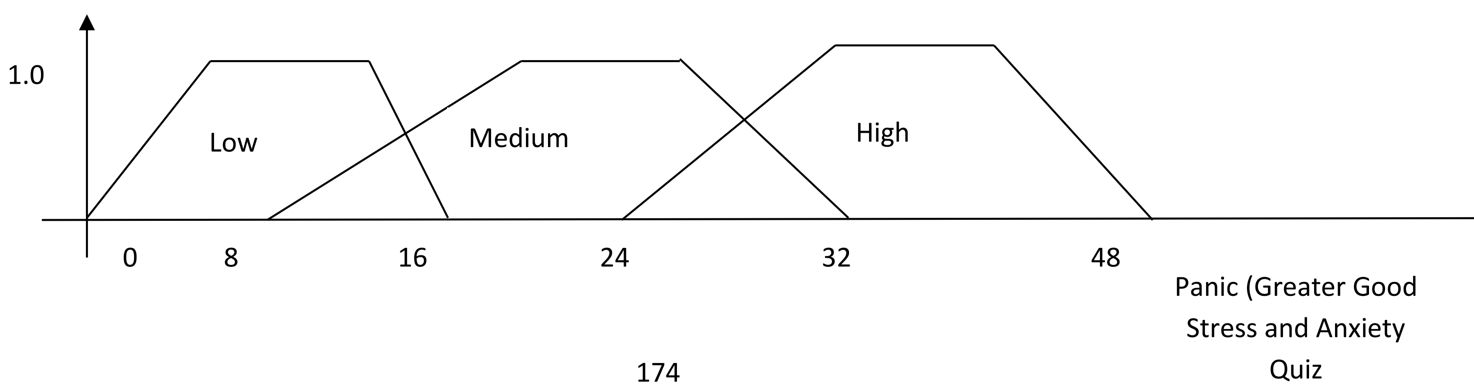
1 output:

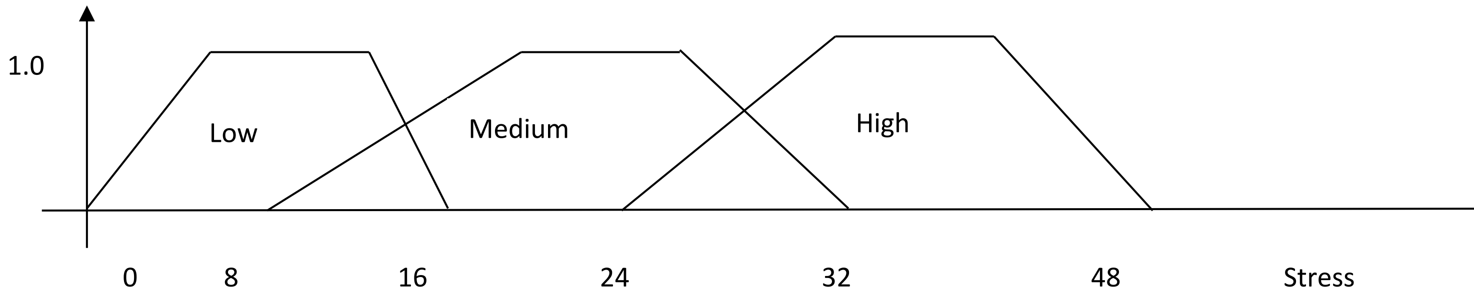
Speed (of agent from fuzzy logic model) (3 levels)

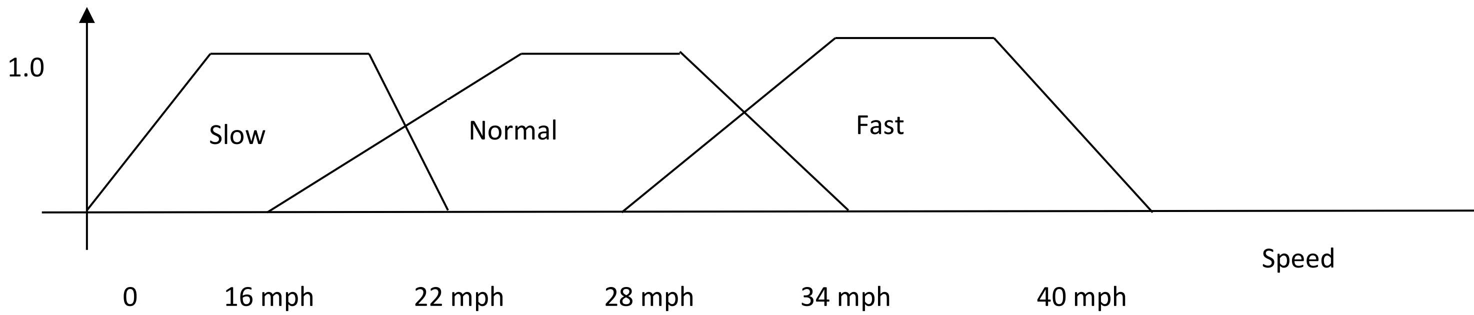
Set of rules to determine output based on input values



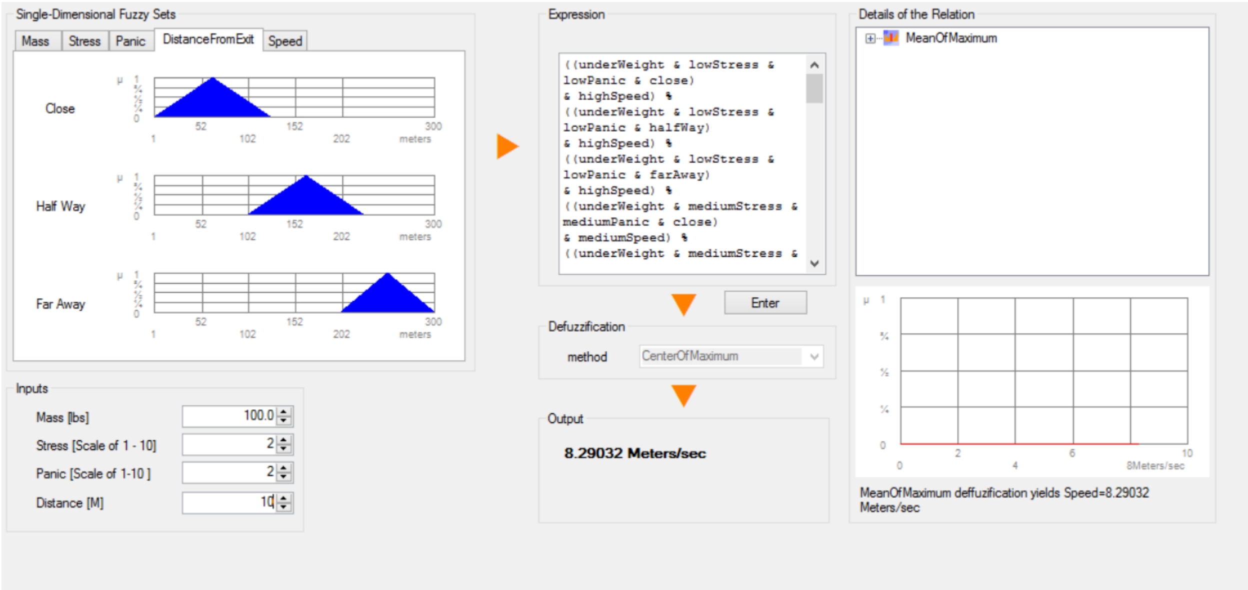


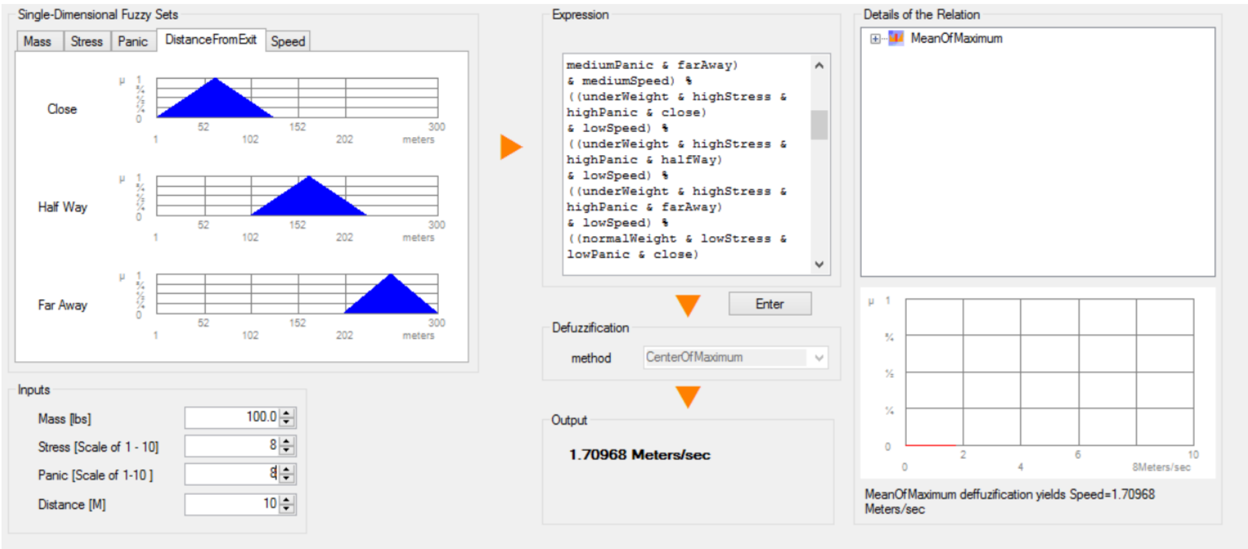


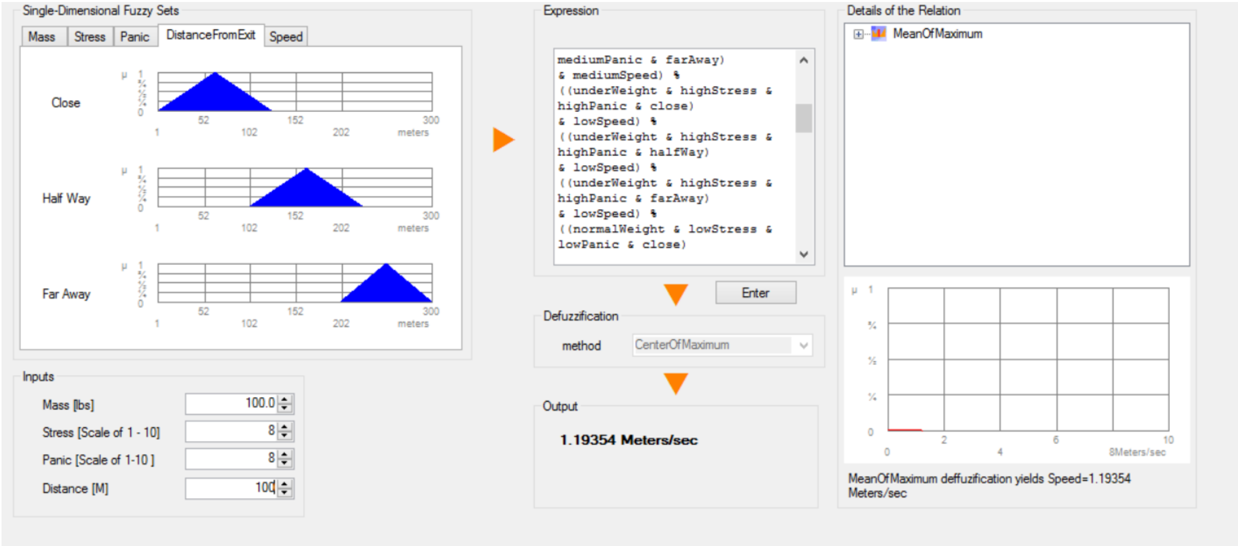

1.0
Low
Medium
High
0
8
16
24
32
48
Stress


1.0
Slow
Normal
Fast
Speed
0
16 mph
22 mph
28 mph
34 mph
40 mph

Appendix C—Rule Editor for Psychological Factors






Single-Dimensional Fuzzy Sets
Mass
Stress
Panic
DistanceFromExit
Speed
Close
Half Way
Far Away
meters
Inputs
Mass [lbs]
100.0
Stress [Scale of 1 - 10]
8
Panic [Scale of 1-10]
8
Distance [M]
100
Expression
mediumPanic & farAway)
& mediumSpeed) %
((underWeight & highStress &
highPanic & close)
& lowSpeed) %
((underWeight & highStress &
highPanic & halfWay)
& lowSpeed) %
((underWeight & highStress &
highPanic & farAway)
& lowSpeed) %
((normalWeight & lowStress &
lowPanic & close)
Enter
Defuzzification
method
CenterOfMaximum
Output
1.19354 Meters/sec
Details of the Relation
MeanOfMaximum
MeanOfMaximum deffuzification yields Speed=1.19354 Meters/sec

Appendix D—Input File for Defining the Behavior of an Agent

subGoalRad 1.5

Obstacles 19

obstacle 1

300 250

400 250

400 300

300 300

obstacle 2

800 500

700 500

700 550

800 550

obstacle 3

800 250

700 250

700 300

800 300

obstacle 4

300 400

400 400

400 450

300 450

obstacle 5

300 500

400 500

400 550

300 550

obstacle 6

800 400

700 400

700 450

800 450

obstacle 7

300 150

400 150

400 200

300 200

obstacle 8

800 150

700 150

700 200

800 200

obstacle 9

100 20

300 20

300 70

100 70

obstacle 10

300 20

500 20

500 70

300 70

obstacle 11

500 20

700 20

700 70

500 70

obstacle 12

700 20

900 20

900 70

700 70

obstacle 13

250 610

450 610

450 660

250 660

obstacle 14

450 610

650 610

650 660

450 660

obstacle 15

650 610

850 610

850 660

650 660

obstacle 16

10 155

60 155

60 10

10 10

obstacle 17

10 320

60 320

60 165

10 165

obstacle 18

10 485

60 485

60 330

10 330

obstacle 19

10 645

60 645

60 495

10 495

Goals 13

goal 0.79999 0.4600

goal 0.79999 0.47000

goal 0.79999 0.48000

goal 0.79999 0.49000

goal 0.79999 0.50000

goal 0.79999 0.51000

goal 0.79999 0.52000

goal 0.79999 0.8600

goal 0.79999 0.8700

goal 0.79999 0.8800

goal 0.79999 0.8900

goal 0.79999 0.9000

goal 0.79999 0.9100

agentCircles 23

peopleCircle

type hostile

position 810 160

stress 10

panic 20

speed 1.19

peopleCircle

type hostile

position 870 260

stress 10

panic 10

speed 1.19

peopleCircle

type hostile

position 810 400

stress 10

panic 10

speed 1.19

peopleCircle

type hostile

position 810 530

stress 20

panic 20

speed 1.19

peopleCircle

type hostile

position 710 600

stress 20

panic 20

speed 1.19

peopleCircle

type hostile

position 500 600

stress 20

panic 20

speed 1.19

peopleCircle

type hostile

position 620 600

stress 20

panic 10

speed 1.19

peopleCircle

type hostile

position 90 400

stress 10

panic 10

speed 1.19

peopleCircle

type hostile

position 90 425

stress 10

panic 10

speed 1.19

peopleCircle

type hostile

position 410 520

stress 10

panic 10

speed 1.19

peopleCircle

type hostile

position 410 540

stress 10

panic 10

speed 1.19

peopleCircle

type hostile

position 90 500

stress 10

panic 10

speed 1.19

peopleCircle

type hostile

position 90 525

stress 10

panic 10

speed 1.19

peopleCircle

type hostile

position 90 380

stress 10

panic 10

speed 1.19

peopleCircle

type hostile

position 110 380

stress 10

panic 10

speed 1.19

peopleCircle

type hostile

position 90 125

stress 10

panic 10

speed 1.19

peopleCircle

type hostile

position 90 140

stress 10

panic 10

speed 1.19

peopleCircle

type hostile

position 580 380

stress 10

panic 10

speed 1.19

peopleCircle

type hostile

position 620 380

stress 10

panic 10

speed 1.19

peopleCircle

type hostile

position 620 300

stress 10

panic 10

speed 1.19

peopleCircle

type hostile

position 620 200

stress 10

panic 10

speed 1.19

peopleCircle

type hostile

position 620 220

stress 10

panic 10

speed 1.19

peopleCircle

type hostile

position 520 400

stress 10

panic 10

speed 1.19

Bibliography

A., K. (n.d.). n Intelligent Predictive Control Approach to the High Speed Cross Country Autonomous Navigation Problem. *Carnegie Mellon Robotics Institute Technical Report, CMU-RI-TR-95-33*, i.

Abu-Taieh, E. M., & Sheikh, A. A. (2010). Handbook of Research on Discrete Event Simulation Environments-Technologies and Applications. *Information Science Reference (IGI Global) Hershey, New York, chapter 3, 4, 15,* .

Aleven, V., McLaren, B., & Sewall, J. (1). Scaling Up Programming by Demonstration for Intelligent Tutoring Systems Development: An Open-Access Web Site for Middle School Mathematics Learning. *IEEE Transactions, 2*, 2009.

Anderson, C. W. (n.d.). Strategy Learning with Multilayer Connectionist Representations. *Department of Computer Science Colorado State University Fort Collins, CO 80523.*

Arkin, R. C. (n.d.). Integrating Behavioral, Perceptual, and World Knowledge in Reactive Navigation. *School of Information and Computer Science, Georgia Institute of Technology, Atlanta, GA 30332-0280, USA*.

Arkin, R. C. (n.d.). Reactive Robotic Systems. *College of Computing, Georgia Tech, Atlanta, GA 30332-0280*.

Baert, S. (2000). “Motion Planning Using Potential Fields.

Barjis, J., Rychkova, I., & Yilmaz, L. (2011). Modeling and Simulation Driven Software Development. *Boston, Massachusetts, Society for Computer Simulation International*, 5 - 8.

Barto, A. G., Sutton, R., & Watkins, C. (n.d.). Learning and sequential decision making. *Technical report, COINS Technical Report 89-95, Dept. of Computer and Information Science, University of Massachusetts*.

Benthorn, L., & Frantzich, H. (1996). Fire alarm in a public building: How do people evaluate information and choose evacuation exit? *Dept of Fire Safety Engineering, Lund University*.

Blythe, P. W., Miller, G. F., & Todd, P. M. (n.d.). Human Simulation of Adaptive Behavior: Interactive studies of pursuit, evasion, courtship, fighting, and play. *Center for Adaptive Behavior and Cognition Max Planck Institute for Psychological Research Leopoldstrasse 24, 80802 Munich, Germany*.

Board, N. T. (2000). Emergency Evacuation of Commercial Airplanes. Safety Study . *NTSB/SS-00/01. Washington, DC: NTSB*.

BRIEFINGS, B. (2006). SmartCockpit.com. *Version 01 (Feb.07)*, 1.

Bruder, G., Steinicke, F., & Wieland, P. (2011). Self-motion illusions in immersive virtual reality environments. *Virtual Reality Conference (VR) IEEE*, 39.

Bruder, G., Steinicke, F., & Wieland, P. (2011). Self-motion illusions in immersive virtual reality environments. *Virtual Reality Conference (VR), 2011 IEEE*, 39.

Brumitt, B. L., & Stentz, A. (n.d.). Dynamic Mission Planning for Multiple Mobile Robots. *Robotics Institute Carnegie Mellon University Pittsburgh, PA 15213*, 1.

Cazenille, L., Bredeche, N., Hamann, H., & Stradner, J. (n.d.). Impact of Neuron Models and Network Structure on Evolving Modular Robot Neural Network Controllers. *GECCO'12, July 7-11, 2012, Philadelphia, Pennsylvania, USA. Copyright 2012 ACM 978-1-4503-1177-9/12/07*.

Chakareski, J. (2011). Multi-Graph Regularization for efficient delivery of user generated content in online social networks . *Acoustic, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference*, 2363.

Chao, E. L., & John L. Henshaw. (2001). How to Plan for Workplace Emergencies and Evacuations. *OSHA 3088*, 2.

Chatila, R., Devy, M., Lacroix, S., & Herrb, M. (Mar 1994). Perception System and Functions for Autonomous Navigation in a Natural Environment. *Centre National de la Recherche Scientifique, Toulouse (France). Inst. de Recherche en Informatique*, 11.

Chertoff, D., Vanderbleek, S., Flore, S., Gallagher, S., & M, S. (2009). Cognitive architecture for Perception-Reaction Intelligent Computer Agents (CAPRICA) . *2009 4th ACM/IEEE Conference*, 1.

Choi, S. H., & Zhu, W. K. (2012). Performance optimization of mobile robots in dynamic environments. *Virtual Environments Human-Computer Interfaces and Measurements Systems (VECIMS) 2012 IEEE International Conference*, 1-3.

Chooramun, N., Lawrence, P., & Gale, E. (2010). Implementing a Hybrid Space Discretisation Within An Agent Based Evacuation Model. *PED 2010, NIST, Maryland USA*.

Cristina, P., & Santos, J. M. (2010). Discovering Sensing Capability in Multi-Agent Systems. *IEEE Computer Society*, 198.

Crombie, D. (29 October 1997). The Examination and Exploration of Algorithms and Complex Behaviour to Realistically Control Multiple Mobile Robots. *The Australian National University Faculty of Engineering and Information Technology*, 25.

D. Helbing, I. F. (2000). Simulating Dynamical Features of Escape Panic. *Macmillan Magazines Ltd*.

Dorigo, M., & Colombetti, M. (n.d.). Robot Shaping: Developing Autonomous Agents through Learning. *INTERNATIONAL COMPUTER SCIENCE INSTITUTE TR-92-040 Revised April 1993.*

Dzima, J. A. (2001). AI Madness: Using AI to Bring Open-City Racing to Life.

Evarts, B. (June 2011). Human Factors Contributing to Fatal Injury. *NFPA Fire Analysis and Research, Quincy MA,*, 1.

Express, V. R. (2014). Passenger Safety. *A Commuter Guide to Safe Travel*, 5.

Feng, D., Singh, S., & Krogh, B. (August, 1990). Implementation of Dynamic Obstacle Avoidance on the CMU Navlab. *Proceedings of IEEE Conference on Systems Engineering*, 208.

Ferrari, L., & Zhu, H. (2010). White Cat: Making Agent Roles Perceivable. *Collaborative Technologies and Systems (CTS) International Symposium*, 637.

Ferrari, L., & Zhu, H. (2011). Enabling Dynamic Roles For Agents. *Collaboration Technologies and Systems (CTS), 2011 International Conference*, 500.

Feurte, F. (2000). Simulating the Collision Avoidance Behavior of Pedestrians. *The University of Tokyo, School of Engineering Department of Electronic Engineering*.

Filho, J. L., & Treleaven, P. C. (June 1994). Genetic-algorithm program-ming environments. *College of London*, 43.

Galea, E. (November 2013). buildingEXODUS V6.0 User Guide and Technical Manual. *Doc Rev 5.0*.

Gat, E., Slack, M. G., Miller, D. P., & Firby, R. J. (1990). Path Planning and Execution Monitoring for a Planetary Rover. *Proceedings of the IEEE International Conference on Robotics and Automation*.

Hayes-Roth, B. (n.d.). An Architecture for Adaptive Intelligent Systems. *Knowledge Systems Laboratory Stanford University 701 Welch Road, Building C Palo Alto, Ca. 94304*.

Hebert, M. (1994). Pixel-Based Range Processing for Autonomous Driving. *Proceedings of the IEEE International Conference onRobotics and Automation*.

Helbing, D., & Molnár, P. (1995). Social Force Model for Pedestrian Dynamics. *Physical Review, E 51*, 4282-4286.

Helbing, D., Farkas, I., Molnar, P., & Vicsek, T. (2002). Simulation of pedestrian crowds in normal and evacuation simulations. *Pedestrian and Evacuation Dynamics. Springer*, 21-58.

Henein, C. M. (2008). Crowds are made of people: Human factors in microscopic crowd models. *Institute of Cognitive Science Carleton University Ottawa, Ontario, Canada*.

Hostikka, S., Korhonen, T., & Keski-Rahkonen, O. (2005). Two-model Monte Carlo Simulation of Fire Scenarios. *Fire Safety Science - The Eight International Symposium on Fire Safety Science, September 18-23, Beijing,China*, 12.

Ian Millington, J. F. (2009). *Artificial Intelligence For Games Second Edition.* Burlington: Morgan Kaufmann.

Jing, L., Zhong, W., & Jiao, L. (2009). Macro-Agent Evolutionary Model for Decomposable Function Optimization. *Trondheim, Norway, IEEE Press*, 62.

Johnson, A., Thompson, E. M., & Coventry, K. R. (2010). Human Perception, Virtual Reality and the Built Environment. *Information Visualization (IV), 2010 14th International Conference*, 604 & 608.

Kobes, M., Oberije, N., Post, J., & Weges, J. (5 - 7TH jULY 2007). Fire response performance model for a systematic analysis of evacuation safety in buildings- a case study of a fire in a football stadium. *Proceedings of the 12th International Fire Science & Engineering Conference, Interflam 2010*.

Korhonen, T., Hostikka, S., & Keski-Rahkonen, O. (2005). A Proposal for the Goals and New Techniques of Modelling Pedestrian Evacuation in Fires. *8th International Symposium on Fire Safety Science, Beijing, China*, 18-23.

Kuligowski, E. D. (2009). The Process of Human Behavior in Fires. *National Institute of Standards and Technology Technical Note 1632 Natl. Inst. Stand. Technol. Tech. Note 1632*, 15.

Kuligowski, E. (n.d.). Review of 28 Egress Models. *National Institute of Standards and Technology*.

Langer, D., Rosenblatt, J., & Hebert, M. (1994). An Integrated System for Autonomous Off-Road Navigation. *Proceedings of the IEEE International Conference on Robotics and Automation*, 2.

Lin, L. (1992). Self-Improving Reactive Agents Based On Reinforcement Learning, Planning and Teaching. *Kluwer Academic Publishers, 8*, 293-321.

Long-Ji, L. (1990). Self-improving reactive agents : case studies of reinforcement learning frameworks. *Computer Science Department. Paper 1967*.

Loscos, C., Marchal, D., & Meyer, A. (2001). Intuitive Crowd Behaviour in Dense Urban Environments using Local Laws. *University College London PSRC project, and with an IST EU project, CREATE (IST-2001-34231)*.

Lozano, J. -A., Gil-Gomez, J. -A., Alcaniz, M., Chirivella, J., & Ferri, J. (2009). Activities of daily living in a virtual reality system for cognitive rehabilitation. *Virtual Rehabilitation International Conference*, 205.

Mahadevan, S., & Connell, J. (1991). Automatic programming of behavior-based robots using reinforcement learning. *IBM T.J. Watson Research Center, Box 704, Yorktown Heights, NY 10598, USA* .

Mahadevan, S., & Connell, J. (1992). Automatic programming of behavior-based robots using reinforcement learning. *Artificial Intelligence, 55*, 311-365.

Mamdouh, A. M., Kaboudan, A., & F.Imam, I. (n.d.). Realistic Behavioral Model for Hierarchical Coordinated Movement and Formation Conservation for Real-Time Strategy and War Games. *IAENG International Journal of Computer Science, 39:3,IJCS_39_3_03*.

Map, E. E. (2014). Grosvenor-Strathmore Station Rockville Pike Exit. *62.014 EVAC Map.indd*, 1.

Matthies, L. (1992). Stereo Vision for Planetary Rovers: Stochastic Modeling to Near Real-Time Implementation. *International Journal of Computer Vision, 8*(1), 1.

McIntire, J. P., Havig, P. R., & McIntire, L. K. (2009). Ideas on authenticating humanness in collaborative systems using AI-hard problems in perception and cognition. *Aerospace & Electronics Conference (NAECON), Proceedings of the IEEE 2009 National*, 54.

Minh, L. V., Adam, C., Canal, R., Gaudou, B., Vinh, H. T., & Taillandier, P. (2010). Simulation of the Emotion Dynamics in a Group of Agents in an Evacuation Situation. *Pacific Rim workshop on Agent-based modeling and simulation of Complex Systems, Kolkata : India*.

Moreira, D. d. (1995). Agents: A Distributed Client/Server System for Leaf Cell Generation. *The University of Kent, Canterbury, UK* , 5.

Musse, S. R., & Thalmann, D. (n.d.). A Model of Human Crowd Behavior: Group Inter-Relationship and Collision Detection Analysis. *Computer Graphics Lab (LIG), Swiss Federal Institute of Technology (EPFL) CH-1015 Lausanne, Switzerland*.

Okazaki, S., & Matsushita, S. (n.d.). A study of simulation model for pedestrian movement with evacuation and queuing. *Department of Architecture and Environmental Design, Architecture and Environmental Engineering, Kyoto University, Yoshidahonmachi, Kyoto, 606-01 Japan*.

Pan, X., Han, C., Law, .. H., & Latombe, J. (n.d.). A COMPUTATIONAL FRAMEWORK TO SIMULATE HUMAN AND SOCIAL BEHAVIORS FOR EGRESS ANALYSIS. *The Center for Integrated Facility Engineering at Stanford University*.

Pankaj, M. (2012). Context-Aware Computing: Beyond Search and Location-Based Services. *IEEE Journals & Magazines*, 13-14.

Park, J. H., Kim, H., Whang, H., Parl, J., & Lee, D. (n.d.). Development of an Agent-based Behavior Module for Evacuation Models - Focused on the Behaviors in the Dark. *Korea Ocean Research and Developmen Institute (KORDI)*.

Peacock, R. D., Averill, J. D., & Kuligowski, E. D. (2009). Stairwell Evacuation from Buildings: What We Know We Don't Know. *National Institute of Standards and Technology, Gaithersburg, MD*, 16.

Penaloza, C. I., Mae, Y., Ohara, K., & Arai, T. (2012). Social human behavior modeling for robot imitation learning. *Mechatronics and Automation (ICMA), 2012 International Conference*, 457.

Pires, T. T. (2005). An approach for modeling human cognitive behavior in evacuation models. *Fire Safety Journal, 40*(0379-7112), 177-189.

Polani, D. (2011). An informational perspective on how the embodiment can relieve cognitive burden . *Artificial Life (ALIFE) 2011 IEEE Symposium*, 1.

Pomerleau, D. (1991). Efficient Training of Artificial Neural Networks for Autonomous Navigation. *Neural Computation, 3*(1), 1.

Poon, L. S. (1994). A Simulation Model of Occupants with Behavioural Attributes in Emergency Evacuation of High-Rise Building Fires. *FIRE SAFETY SCIENCE-PROCEEDINGS OF THE FOURTH INTERNATIONAL SYMPOSIUM*, 681-692.

Prithviraj, D., Cheng, K., & Li, F. (2009). Flocking-based distributed terrain coverage with dynamically-formed teams of mobile mini-robots. *Swarm Intelligence Symposium, 2009, SIS*, 1.

Pyeatt, L. D., & Howe, A. E. (1998). Learning to Race: Experiments with a Simulated Race Car. *American Association for Artificial Intelligence*.

Quinn, M. J., Metoyer, R. A., & Hunter-Zaworski, K. (n.d.). PARALLEL IMPLEMENTATION OF THE SOCIAL FORCES MODEL. *School of Electrical Engineering and Computer Science Department of Civil, Construction, and Environmental Engineering Oregon State University Corvallis, OR 97331 USA*.

Research, F. A. (November 2013). An Overview of the U.S. Fire Problem. *NFPA's Fire Analysis and Research Division*, 1.

Reynolds, C. W. (2000). Interaction with Groups of Autonomous Characters. *Proceedings of Game Developers Conference, CMP Game Media Group (formerly: Miller Freeman Game Group)*(California), 449-460.

Russell, S., & Norvig, P. (2003). Artificial Intelligence a Modern Approach. *Prentice Hall Pearson Education, Inc, Second Edition*, 116,526,632,736.

S. Gwynne, E. R., Lawrence, P. J., & Filippidis, L. (n.d.). A review of the methodologies used in the computer simulation of evacuation from the built environment. *Fire Safety Engineering Group, Center for Numerical Modeling and Process Analysis, University of Greenwich, London, UK*.

Scerri, D., Hickmott, S., & Padgham, L. (2012). User understanding of cognitive processes in simulation: A tool for exploring and modifying. *Simulation Conference (WSC), Proceedings of the 2012 Winter Conference*, 1-5.

Services, A. C. (FLT_OPS—CAB_OPS—SEQ 12—REV 01—NOV. 2006). Flight Operations Briefing Notes Cabin Operations Unplanned Ground Evacuation. , 1.

Sharma, S. (2009). Avatarsim: A multi-agent system for emergency evacuation simulation. *Journal of Computational Methods in Science and Engineering, 9*, S13-S22.

Sharma, S. (2009). Simulation and modeling of group behavior during evacuation. *IEEE Symposium Series on Computational Intelligence, Intelligent Agents*.

Sharma, S. (2010). Favorite goal in agent based modeling and simulation. *Proceedings of the 19th International Conference on Software Engineering and Data Engineering, SEDE, San Francisco,* *California, USA*.

Sharma, S. (2010). Fuzzy approach for predicting probability of reaching a target in a battlefield environment. *International Journal of Computers and their Applications, IJCA, 17*.

Sharma, S. (2010). Military route planning in battle field simulations for a multi-agent system. *Journal of Computational Methods in Science and Engineering, 10* , S97 S105.

Sharma, S. (2012). Use of Favorite Goal in Agent Based Modeling and Simulation. *IJCA, 19*, 1.

Sharma, S., & Gifford, S. (2005). Using RFID to evacuate evacuation behavior models. *Proceedings of North American Fuzzy Information Processing Society Conference, IEEE*.

Sharma, S., & Otunba, S. (2012). Collaborative virtual environment to study aircraft evacuation for training and education. *Collaboration Technologies and Systems (CTS), 2012 International Conference*, 569.

Sharma, S., & Singh, H. (2006). Multi-agent system for simulating human behavior in egress simulations. *Proceedings of NAACSOS, Annual Conference of the North American Association for computational social and organizational sciences, Notre Dame, Indiana*.

Sharma, S., Otunba, S., Ogunlana, K., & Tripathy, T. (2012). Intelligent Agents in a Goal Finding Application for Homeland Security. *Proceedings of the IEEE Southeastcon*, 1.

Sharma, S., Singh, H., & Gerhart, G. R. (2007). Simulation of convoy of unmanned vehicles using agent based modeling. *SPIE conference on security and defense, Florence, Italy*.

Sharma, S., Singh, H., & Prakash, A. (2008). Multi-agent modeling and simulation of human behavior in aircraft evacuations. *IEEE Transactions on Aerospace and Electronic Systems, 44*.

Shen, T., & Chien, S. (2005). AN EVACUATION SIMULATION MODEL (ESM) FOR BUILDING EVALUATION. *Graduate School of Fire Science and Administration, Central Police University, Taiwan International Journal on Architectural Science, 6, Number 1*, 15-30.

Shi, L., Xie, Q., Cheng, X., Chen, L., Zhou, Y., & Zhang, R. (n.d.). Developing a database for emergency evacuation model. *State Key Laboratory of Fire Science, University of Science and Technology of China, West Campus, Anhui, Hefei 230027, PR China*.

Shichel, Y., & Sipper, M. (2011). GP-RARS: Evolving Controllers for the Robot Auto Racing Simulator. *Department of Computer Science, Ben-Gurion University, Beer-Sheva, Israel* .

Signoretti, A., Feitosa, A., Campos, A., Canuto, A., Xavier, A., Fialho, J., & S.V. (2011). Using an Affective Attention Focus for Improving the Reasoning Process and Behavior of Intelligent Agents. *Web Intelligence and Intelligent Agent Technology (WI-IAT)*.

Simulex User Guide 6.0. (2012). *Integrated Environmental Solutions Limited*.

Singh, S., Feng, D., Keller, P., Shaffer, G., Shi, W., Shin, D., . . . Wu, B. (1991). A System for Fast Navigation of Auton-omous Vehicles. *Robotics Institute Technical Report CMU-RI-TR-91-20, Carnegie Mellon University*.

Sobota, B., Hrozek, F., Korecko, S., & Szabo, C. (2011). Virtual reality technologies as an interface of cognitive communication and information systems. *Cognitive Infocommunications (CogInfoCom), 2011 2nd International Conference*, 1.

Stentz, A., & Martial, H. (August 1995). A Complete Navigation System for Goal Acquisition in Unknown Environments. *In Autonomous Robots, 2*(2).

Still, G. K. (n.d.). Review of Pedestrian and Evacuation Simulations. *Crowd Dynamics Ltd. The Mill House, Staveley Mill Yard, Staveley, LA8 9LS. UK*, 1-9.

Still, K. G. (2005). Designing collective behavior in a group of humans using a real-time polling system and interactive evolution. *Swarm Intelligence Symposium, Proceedings SIS 2005 IEEE, 8-10*(8-10), 15—2.

Sun, R. (2009). Theoretical status of computational cognitive modeling. *Cognitive Systems Research, 10, No.2*, 2.

Sutton, R. S. (1988). Learning to Predict by the Methods of Temporal Differences. *Kluwer Academic Publishers Machine Learning 3: 9-44,* .

Tecchia, F., Loscos, C., Conroy, R., & Chrysanthou, Y. (n.d.). Agent Behaviour Simulator (ABS): A Platform for Urban Behaviour Development. *University College London Department of Computer Science Gower Street London WC1E 6BT UK*.

Thangaraj, R., Pant, M., Chelliah, T., & Abraham, A. (2012). Opposition based Chaotic Differential Evolution algorithm for solving global optimization problems. *Nature and Biologically Inspired Computing (NaBIC), 2912 Fourth World Congress*, 1.

Tran, V. T. (2013). More than just another crowd, we need a waiting line instead. *Distinktion: Scandinavian Journal of Social Theory, 14*(2, 2013 Special Issue: Postmodern Crowds: Re-Inventing Crowd Thinking).

transitchicago. (n.d.). *CTA Trains Emergency and Evacuation Instructions.* Chicago.

Verth, J. V., Brueggemann, V., Owen, J., & McMurry, P. (2000). Formation-Based Pathfinding With Real-World Vehicles. *Red Storm Entertainment*.

Wilkins, D. E., Myers, K. L., Lowrance, J. D., & Wesley, L. P. (1994). Planning and Reacting in Uncertain and Dynamic Environments. *Artificial Intelligence Center SRI International* .

Winter, H. (2012). Modeling Crowd Dynamics During Evacuation Situations Using Simulation. *Lancaster University*.

Wooldridge, M. (1999). Intelligent agents, in Weiss, G. (ed.), Multiagent Systems, A Modern Approach to Distributed Artificial Intelligence. *MIT Press, MA*, 27-78.

Wooldridge, M., & Jennings, N. R. (1995). Intelligent agents: Theory and practice. *Knowledge Engineering Review 10 (2)*, 115–152.

WU, L., & Lin, H. (2009). A Case Study of Developing Personalized Spatial Cognitive Road Network and Raster Capable Route Finding Algorithm for Pedestrian Evacuation Behavior Simulation. *Geoinformatics, 2009 17th International Conference*, 1.

Yan-hua, R., & Xue-ren, Y. (2011). Thinking of China emergency rescue system of education and training. *Information Systems for Crisis Response and Management (ISCRAM) 2011 International Conference*, 342.

Yi, Z. (2010). Solving Vehicle Routing Problem Based on Improved Genetic Algorithm. *Information Science and Engineering (ISISE), 2010 International Symposium*, 594.

BOWIE STATE UNIVERSITY
Institutional Review Board
Proposal Submission Form (Approved 1/98)
Name _____Kola Ogunlana_______ E-mail _ogunlanak0415@students.bowiestate.edu____
Phone_3016338205_____
Home Address ______N/A____
School _College of Arts and Science_____ Department _Computer Science____________
Thesis/Dissertation Chair _Dr. Sharad Sharma_ *(If this is a student candidate's proposal)*
Start Date _May 2014________________ End Date _____September 2015________________ _
Sponsor's Name: N/A____________________________ Project title: (If this protocol applies to several sponsored projects, provide all different titles)
1. **Evacuation Drill for validating modeling and simulation of goal-finding agent behavior**
2.
3.
After completing the above section, please respond to questions 1 through 15 on this form. If the proposed research is **EXEMPT** from IRB review, please indicate the appropriate category number (1-6) from the Exemption Reasons attached.
Please allow 2-4 weeks for the IRB review process to be completed prior to the submission of the proposal to the sponsor; or if it is not a sponsored project, before the start date of the research.

1. Please provide a precise description of how human subjects will be involved in the research, including a clear description of all activities and responsibilities of the subjects

The purpose of the study is to collect data for students of different ages (above 18 years) who participate in the evacuation drill. The evacuation drill will occur in room 109 in Computer Science building at BSU.

The drill will be used to study building evacuations in multi user environment and can also be used as an education and training for emergency responders. Evacuation drills will be used to study human behavior that can be evaluated in the real world. The data collected will be used to validate goal finding application and safety recommendations to make emergency evacuation safer. We will use 8 cameras to record and collect data on the route, time, path and which exit the students take during the evacuation.

The students would be asked to participate in the evacuation drill scheduled for a 60-min session at their convenience. Upon arrival at the study the participants will undergo informed consent procedures in which drill personnel will explain to them the procedures, rules, and read consent form with them.50-70 Participants will be asked to participate in the multi-user environment, During the session the participants will be placed at various locations in the room with computers to perform their normal daily tasks like checking their email and working on their class projects. When the timer starts with the cameras rolling, they will be given instructions to evacuate the room by avoiding obstacles on their way to reach the goal. The goal for building evacuation will be to reach one of two exit doors. After the task has been completed, the participants will be given a survey questionnaire including questions on their experience (see appendix). They will be debriefed and given opportunity to ask questions or express any feedback they have.

2. What is the pool of subjects? Will there be any minors (under the age of eighteen)?

The participants will not be paid and their participation will be voluntary. Adult participants (above 18 years of age) will be selected from BSU campus. The participants will be recruited through the use of posted flyers around BSU campus. Participants below 18 years of age will be excluded from the study.

3. How many subjects will be recruited?

We aim to recruit 50 – 70 participants because the maximum room capacity where the evacuation drill will take place is 70 students. One session will have 10-20 students participating in a multi-user environment. We hope to have at least 5 sessions.

4. Describe the risk of the subjects? Could the research be done without using humans?

Risks to the participants are minimal. There is a risk of fatigue from rushing to evacuate the room. To minimize the risk, the participants will be given specific instructions before the evacuation drill and also informed that they can leave at any time during the drill.

5. How will the subjects be informed that they do not have to participate in the study, and may withdraw at any time with no penalty?

The consent procedures will be conducted in the same room where the study will take place so that the research staff can instruct them on how to stop participating in the drill.

6. In what way has the confidentiality and privacy of the subjects' responses been ensured?

Research data will be retained for three years after the completion of study. This will allow sufficient time for analysis and publication of research data. After three years the research data will be destroyed.

7. Is there deception to the human subjects? If yes, what debriefing procedures have been arranged?

No

8. If the procedures are physically invasive or potentially harmful, describe arrangements made for medical referral.

No

9. If the procedures could be emotionally upsetting, describe arrangements made for psychological counseling.

No

10. What provisions have been made for cultural and language problems, it they arise?

No anticipated cultural difference or language problems

11. Has consent been obtained from the authorities where the research is to be conducted?

N/A

12. Include a copy of the written informed consent form with the proposal. If it is not possible to obtain a written consent form, describe how an understandable explanation will be given to the subjects.

Attached

13. Attach a copy of a positive parental consent if the subjects are minors.

N/A

14. lf a surveyor questionnaire is used, please include copies and describe the exact nature of the questions to be asked.

Attached

15. lf a student candidate is to conduct the research, submit a statement from the faculty advisor, indicating:

- The faculty member's approval of the project: attached
- The faculty member's willingness to supervise the research
- An indication that the student candidate is competent to conduct the research

Submit 3 hard copies each of the Proposal Submission Form, the questionnaire (instrument) or survey (if used), consent forms and statement of support from the faculty advisor (when it's a student candidate's proposal) to:

Dr. Cosmas U. Nwokeafor, Chair IRB
Center for Business & Graduation Studies
Suite 1312
301-860-3406 (off)
301-860-3414 (fax)
cnwokeafor@bowiestate.edu

Exemption Reasons

1. Research that does not involve direct contact with human subjects such as interviews, surveys, etc.
2. Research involving the use of educational tests (cognitive, diagnostic, aptitude, achievement) survey procedures, interview procedures or observation of public behavior, unless: (i) information obtained is recorded in such a manner that human subjects can be identified, directly or through identifiers linked to the subjects; and (ii) any disclosure of the human subjects' responses outside the research could reasonably place the subjects at the risk of criminal or civil liability or be damaging to the subjects' financial standing, employability, or reputation.
3. Research involving the use of educational tests (cognitive, diagnostic, aptitude, achievement), survey procedures, or observation of public behavior that is not exempt under paragraph (2) if (i) the human subjects are elected or appointed public officials or candidates for public office; or (ii) the research is conducted for the Department of Justice under the Federal stature 42 U.S.C. 3789g, or for the National Center for Education Statistics under Federal statute 20 U.S.C. 1221 e-1, which provide certain legal protections and requirements for confidentiality.
4. Research involving the collection of existing data, documents, records, pathological specimens, or diagnostic specimens, if these sources are publicly available or if the information is recorded by the investigator in such a manner that subjects cannot be identified directly or through identifiers linked to the subjects.
5. Research and demonstration projects which are conducted by or subject to the approval of department or agency heads and which are designed to study, evaluate or otherwise examine (i) public benefit or service programs; (ii) procedures for obtaining benefits or services under those programs; (iii) possible changes in or alternatives to those programs or procedures; or (iv) possible changes in methods or levels of payment for benefits or services under those programs.
6. Taste and food quality evaluation and consumer acceptance studies, if wholes wholesome foods without additives are consumed or (ii) if a food is consumed that contains a food ingredient at or below the level and for a use found to be safe, or agricultural, chemical or environmental contaminant at or below the level found to be safe, by the Food and Drug Administration or approved by the Environmental Protection Agency or the Food Safety and Inspection Service of the U.S. Department of Agriculture.

NOTE: If the application is to be reviewed by the Institutional Review Board as exempt, one copy is sufficient. Complete the Proposal Submission Form and include the consent form.

Brief Biography

Dr. Sharad Sharma
Associate Professor
Department of Computer Science
14000 Jericho Park Road, Suite 317
Bowie State University
Bowie, Maryland 20715, U.S.A.
ssharma@bowiestate.edu

Ph: (301) 860-4502
Fax: (301) 860-3979

Dr. Sharad Sharma is an Associate Professor in the Department of Computer Science at Bowie State University. He has received a Ph.D. in Computer Engineering from Wayne State University, Detroit, MI in 2006 and an M.S. from University of Michigan, Ann Arbor, MI in 2003. He has won the "Outstanding Researcher Award" in 2011 and 2013, "Outstanding Faculty Award" in year 2012, "Outstanding Publication Award" in year 2010, and "Outstanding Young Faculty Award" in year 2009 at College of Arts and Science at Bowie State University. Dr. Sharma is the Director of the Virtual Reality Laboratory at Bowie State University. The laboratory applies virtual reality and augmented reality as a tool for learning, training, and education. Dr. Sharma's research focus is on modeling and simulation of multi-agent systems for emergency scenarios. His work is motivated by the need of research in real-time agent navigation for reaching a goal in emergency situations like evacuation.

Kolawole Ogunlana
Ph.D. Candidate Student
Department of Computer Science
14000 Jericho Park Road
Bowie State University
Bowie, Maryland 20715, U.S.A.
ogunlanak0415@bowiestate.edu

Ph: (301) 633-8205

Kolawole Ogunlana is a Ph.D. candidate student in the Department of Computer Science at Bowie State University. He has received an M.S. in Computer Science from Catholic University of America (CUA) in 2006 and a B.S. from University of Maryland, College Park in 2003. He has published many papers under Dr. Sharma, Director of the Virtual Reality Laboratory, at College of Arts and Science at Bowie State University. The Virtual Reality Laboratory applies virtual reality and augmented reality as a tool for learning, training, and education. Kola's research focus is on modeling and simulation of multi-agent systems for emergency scenarios, especially in a building. His work is motivated by the need of research in real-time agent navigation for reaching a goal in emergency situations like evacuation, saving time and money.