**Chapter 3: Research Question and Methodology**

CHRISTEN FORD, Baldwin Wallace University

1. **RESEARCH QUESTION**

Path planning in a multi-level, three-dimensional, dynamic environment is not easily solved. This has led to many different solutions to this problem that were created for specific application domains within this problem. Many solutions are not cross compatible. A solution for a specific application of this problem may not work for another, and in fact, this is often the case. Is it possible to adapt real-time rapidly exploring trees to better suit the needs of dynamic path planning in three dimensions? The algorithm should be able to compute paths on the fly Given this knowledge, we propose a method for solving the multi-level, three-dimensional, dynamic path-planning problem that will be generalized to apply to various application domains.

1. **METHODOLOGY**

This research will utilize the Project Malmo framework for the test environment for this research. Initially established by Microsoft Research on June 1st, 2015. Project Malmo is a framework that utilizes the Minecraft game environment for creating artificial agents. It was chosen as Minecraft is a discretized, constrained environment, where in-game agents have predictable behaviors. In addition, Minecraft contains both a world generator and the ability for players to create their own worlds. As the focus of this research will be on developing the path-planning agent and not the world it will navigate, it makes sense to utilize a prebuilt framework.

The agent must be able to navigate a set of small to medium sized predefined worlds with the goal of navigating to the goal destination. Before it may be utilized on randomly generated worlds, the agent must meet the following criteria for all testing worlds:

1. The agent shall be able to navigate diverse, multi-leveled terrain, some of which is hazardous to the agent such as lava.
2. The agent shall be able to navigate around opposing agents, avoiding them when necessary.
3. The agent shall not exceed greater than 25% deviance from the optimal path to the goal. As each world contains a predefined layout and goal, the only contributing factor to deviance from the optimal path shall be opposing agents and hazardous terrain.

To further strengthen statistical analysis of the path planning algorithm, it shall be run across each map a total of 1,000 times. This should enable us to filter out any bias introduced into the population via a small sample size. Because the algorithm will be run on the provided NVIDIA GPU cores, time constraints on the actual testing process are not a concern.

As Minecraft is a complex, three-dimensional multi-level environment, preprocessing must be performed before the agent may path the world. This thesis shall utilize a recently released real time variant of rapidly exploring random trees (RT-RRT\*). RT-RRT\* creates an elliptical sample space to create the tree from. To do this in the Minecraft environment, we will leverage the distinct unit boundaries created by the (x, y, z) world system. Where each unit is called a block, and is exactly one meter in length, width, and height and is exactly sixteen pixels.

A connected hierarchical tree can be constructed at each z-level of the world, with transition points being single blocks that connect one z-level to the next. After this stage, three-dimensional RT-RRT\* will be able to function as it normally does in two-dimensions.

The machine utilized for this research is an MSI GT62VR 7RE with the following hardware:

* Windows 10 Professional
* Intel Core i7 7700HQ @ 2.8GHz
* 16 GB of DDR4-2400 Memory
* 480GB Intel 535 Solid State Drive
* NVIDIA GeForce GTX 1070 8GB

In addition, Project Malmo allows for writing agents in Java, C#, Python, and C++. However, for optimization and compatibility with all of Project Malmos’ features and NVIDIAs’ Compute Unified Device Architecture (CUDA) application programming interface, this thesis shall utilize the Visual C++ 2015 programming language.

1. **PLAN**

Preceding the actual implementation of the thesis, a few preliminary steps must be accomplished first, these include:

1. Install and configure Project Malmo on the development machine.
2. Create a data structure to represent the ellipsoidal tree of the search space.
3. Upscale RT-RRT\* to three dimensions.
4. Parallelize RT-RRT\* to work across multiple GPU cores.

After these steps are accomplished, the pathing algorithm will be hooked up to Project Malmo so that it may retrieve information from the game environment and send pathing information back to the agent playing the game. At this point, data collection can begin. In this stage, the pathing algorithm shall be run per the parameters outlined in the “Methodology” section. Once the algorithm is successful per the success criteria, it shall be allowed to run on several randomly generated worlds to gauge effectiveness (per the same pathing criteria for the predefined worlds) in times of true uncertainty. Results from testing on the predefined worlds shall be used as a baseline shall be used to gauge how well the agent performs on the random worlds.

This thesis will produce a three-dimensional version of real-time rapidly exploring random trees. It shall utilize Project Malmo as a testbed. To verify the algorithm, it will utilize several predefined, user-created worlds to ensure compatibility with the Minecraft environment. After this point, the algorithm shall be utilized on several randomly generated worlds. To ensure repeatability, the random world generated will be seeded with known Minecraft seeds for popular Minecraft worlds.

1. **PUBLICATION POSSIBLITIES**

This thesis shall be publishable in various journals and conferences that deal with artificial intelligence. Several publication possibilities include:

1. Journal of Artificial Intelligence Research
2. The Annual Symposium on Computational Geometry
3. The International Conference on Motion in Games
4. Artificial Intelligence: An International Journal
5. AI & Society: Journal of Knowledge, Culture, and Communication
6. Applied Intelligence: The International Journal of Artificial Intelligence, Neural Networks, and Complex Problem-Solving Technologies