**Analising applicable techniques that can be used to navigate an agent across a dynamic environment.**

**Peter Bush (2017)**

**School of Computing and Digital Technologies, College Rd, Stoke-on-Trent ST4 2DE**

1. Introduction

A dynamic environment is one that is always changing and therefore in a game the agent must find a way of noticing changes to their environment and work out paths that avoid obstacles at a low cost and is computationally effective to calculate. A\* is a static based path-finding technique where the path will be drawn up from the start taking in the agents location and its target, however as noted by *McCabe, H. Graham, R. & Sheridan, S. (2003)* *“If a dynamic object then blocks the path the agent would have no knowledge of this and would continue on as normal and walk straight into the object.”* This would therefore would mean that the AI would have to try and recalculate a new path around the object adding to the CPU overhead. This is where the need to create more dynamic approaches to path finding to allow for the obstacles in the world to be avoided earlier on, therefore reducing the operating costs.

2. Extended Distance Propagation Algorithm (EDP)

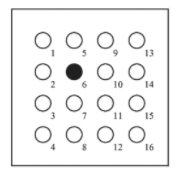
This approach utilities the abilities of A\* but also labels each point on a map to either be a traversable node (*Free Space*) or and obstacle and only allowing the agent to occupy a free space on the map. *Ji, S. Yang, L. (2012)* illustrate a diagram *Figure 1* which shows one single node that the agent would have to avoid. EDP also uses 16-Adjacency test on nodes to guarantee the most optimal path is generated in a reduced time using the expanded search area. *Ji, S. Yang, L. (2012)* noted that searching *“is only related with the size of the map”* and each node describing its row and column and assigned value to specify if it is clear or not. Due to the small amount of information needed and the search cost relating to the maps size this algorithm remains O(n) where n is the number of grids of the grid map keeping it viable for the use in games due to its low use cost. When it comes to generating the path to follow EDP utilises an approach that is commonly found in real time path finding where any dynamic obstacles are treated as static objects and the path is constantly re-planned when the dynamic obstacles move. The general principal of EDP is that the algorithm spreads out from the target node, If the node that is checked contains an obstacle it is given a specific ID value and the algorithm will test other nodes, storing their minimum cost to the target (Using a diagonal distance heuristic). Once the algorithm reaches the agents nearest node the agent will move to the node within the 16-Adjacency test where the minimal cost is the lowest, repeating until it arrives at the target node or one of the obstacles move, in which case the agent will have a new path generated from its current node to the target again updating the nodes between to make sure they still don’t contain any obstacles.

Figure 1: Regular square grid

with barrier at location 6

3. Dynamic A Star Algorithm (D\*)

The creator of the D\* Algorithm Stentz, A. argues that by creating a global path from all known information then attempting to locally circumvent any obstacles, recreating the path when it is completely blocked off (Which is similar to the process taken with EDP) would allow for a complete approach to creating a dynamic obstacle avoiding path but *Stentz, A. (1994)* notes “they are also sub optimal in the sense that they do not generate the lowest cost path given the sensor information as it is acquired and assuming all known, a priori information is correct. This is why Stentz, A. created the D\* Algorithm which works with a map under any state, be it full or empty or containing partial information about the environment and claims to be able to similar but far more efficient than the brute force optimal re-planning approach which is used in the EDP Algorithm. Just like in the A\* algorithm D\* maintains an OPEN

as noted by *Ji, S. Yang, L. (2012) “Extending the DP algorithm that builds on the observation that the distance between any two free spaces is defined to be the minimum Euclidean length of all paths joining the two points through non barrier adjacent neighbors.*

**References:**

1. McCabe, H. Graham, R. & Sheridan, S. (2003) *Path-finding in Computer Games* [Online] Available from: <http://arrow.dit.ie/cgi/viewcontent.cgi?article=1063&context=itbj> [Accessed: 15/04/2017]
2. Stentz, A. (1994) *Optimal and Efficient Path Planning for Partially-Known Environments* [Online] Available from: <http://ieeexplore.ieee.org.ezproxy.staffs.ac.uk/document/351061/> [Accessed: 17/04/2017]
3. Ji, S. Yang, L. (2012) *Extension of Dynamic Programming Algorithm in Robotic Path Planning* [Online] Available from: <http://ieeexplore.ieee.org.ezproxy.staffs.ac.uk/document/6394746/> [Accessed: 16/04/2017]