DRONE EXPRESS - GLOBAL OBSTACLE AVOIDANCE

1 INTRODUCTION

Obstacle avoidance is a crucial element in Autonomous Unmanned Aerial Vehicles (UAVs) operations. Drone collisions are capable of not only destroying the drone but also causing damage to objects and people on the ground. Obstacles, in general, fall into two categories - local obstacles and global obstacles. Local obstacles are defined as dynamic objects which suddenly appear in the UAV's path such as birds in flight, other drones, or unknown objects. Global obstacle are defined as stationary objects that do not change their location frequently such as tall trees or cell towers. They also include areas that are known and categorized as prohibited or potentially dangerous for UAV flight including restricted air space as well as places on the ground that would incur significant damage in case of a crash.

Obstacle avoidance is accomplished by including, in the UAV's overall navigation system, Obstacle Avoidance Algorithms that prevent collision with both global and local obstacles. They mitigate potential collisions by changing flight path in real time and re-route the UAV as necessary while in flight. The algorithms are highly sophisticated and require considerable computer processing power which is not usually available onboard standard UAVs. Also the extra weight and power requirements of the processor impact to some degree both flight time and payload capabilities. For that reason most UAV designers choose to process these algorithms on the ground portion of the overall navigation system or the Cloud, not on the UAV itself. The problem with this approach is that it relies on having a constant, low latency wireless connection between the UAV and the ground network. Low latency is critical because once an obstacle is identified even the smallest amount of latency in notifying the UAV could result in a collision.

<u>Drone Express has, instead, chosen to process both global and local obstacle avoidance algorithms on the UAV itself</u>. For that purpose each of its drones includes a high-computational embedded system-on-module (SOM) co-processor with GPU. Analysis has shown that the extra weight and power consumption of the SOM will be almost negated by the reduction in traditional obstacle sense and detect hardware that is normally needed on a UAV.

2 GLOBAL OBSTACLE AVOIDANCE ALGORITHM

Global obstacle avoidance requires several different feeds of data including cell tower locators, topographical maps, geographical maps and airspace maps. **A shortest route path finding algorithm, known as A*, is then used to produce**

a flight path that avoids these obstacles. The initial flight path is then loaded into the drone before take-off. If the drone deviates from the plan during flight it will calculate a new route and update the flight plan. This method of adaptive flight planning reduces the risk of an in-flight collision and limits the damage from a crash including the loss of life.

3 A* PATHFINDING ALGORITHM

The objective of A* is to find the shorted path given a graph of nodes with defined **start** and **end** locations. The shorted path algorithm used by Drone Express is based on the A* algorithm which is fundamentally based on Dijkstra's algorithm. In these scenarios, each *node* is defined from the beginning. A node can be thought of as a location and all possible *edges* (paths) to different nodes. Each edge has a specific *cost* or number that references a certain action or work associated between two nodes.

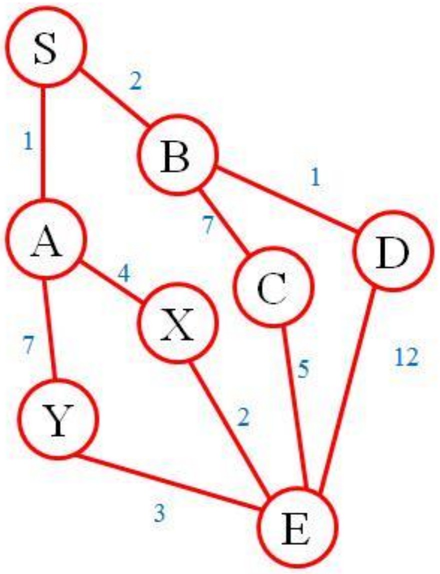
In this case, the cost of an edge can be thought of as the distances between two nodes, but this isn't always true for other implementations. This is the *edge cos*t. In Dijkstra's algorithm this is the only cost. However, in A* there is another cost, *heuristic cost*. This is defined as the cost from a specific node to the end node. In our case it is the distance from a specific node to the end node. When iterating through each node, the total cost of the node needs to be calculated it. The *total cost* of the node is defined as the sum of all edge costs from the start node to the current node, plus the heuristic cost.

Beginning at the **start** node, A* explores that node by iterating through each of that node's *children*. Each node has associated edges, or paths to other nodes. Those can be thought of as its children where the original node is considered the *parent*. When a child node is created its total cost is calculated. If the child node has already been created as the result of another path, its total cost from the previous path is compared to the new total cost from the current path. The child with the minimum total cost is kept and the other is discarded. The child is then added to a priority queue that is based on a minimum binary heap. The need also keeps record of all the nodes that preceded it.

Once the parent calculations are finished, a new node needs to be processed. Since the objective is to find the shortest path, the next node to expand is the one with the least cost. That node is popped from the priority queue and the iteration of its children starts.

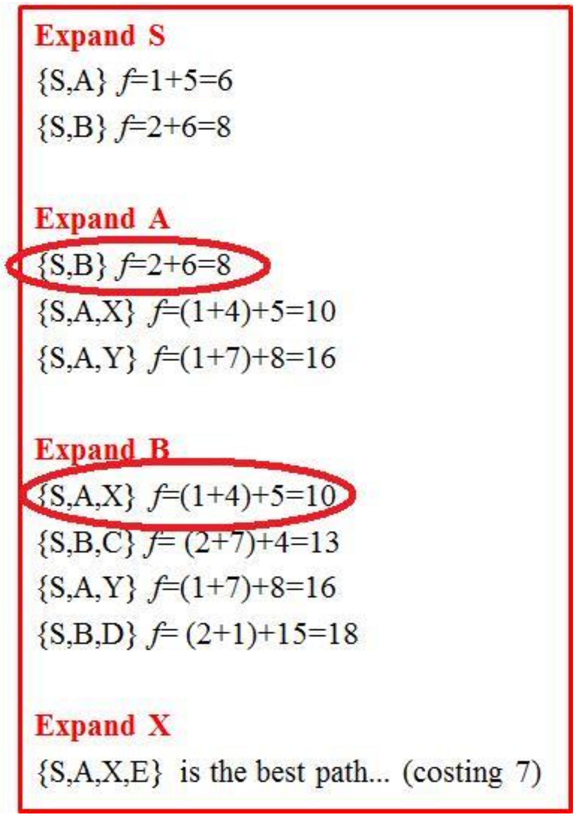
The algorithm continues in this fashion until the node that is popped from the priority queue is the **end** node. The nodes keep track of all the nodes in the path prior to it. Therefore, to reconstruct its path from start to finish, A* traverses through the end node to the start node and reverses the list.

This methodology is depicted in the image below:



Values for h:

A:5, B:6, C:4, D:15, X:5, Y:8



Expand S

{S,A} *f*=1+5=6

{S,B} *f*=2+6=8

Expand A

{S,B} *f*=2+6=8

{S,A,X} *f*=(1+4)+5=10

{S,A,Y} *f*=(1+7)+8=16

Expand B

{S,A,X} *f*=(1+4)+5=10

{S,B,C} *f*= (2+7)+4=13

{S,A,Y} *f*=(1+7)+8=16

{S,B,D} *f*= (2+1)+15=18

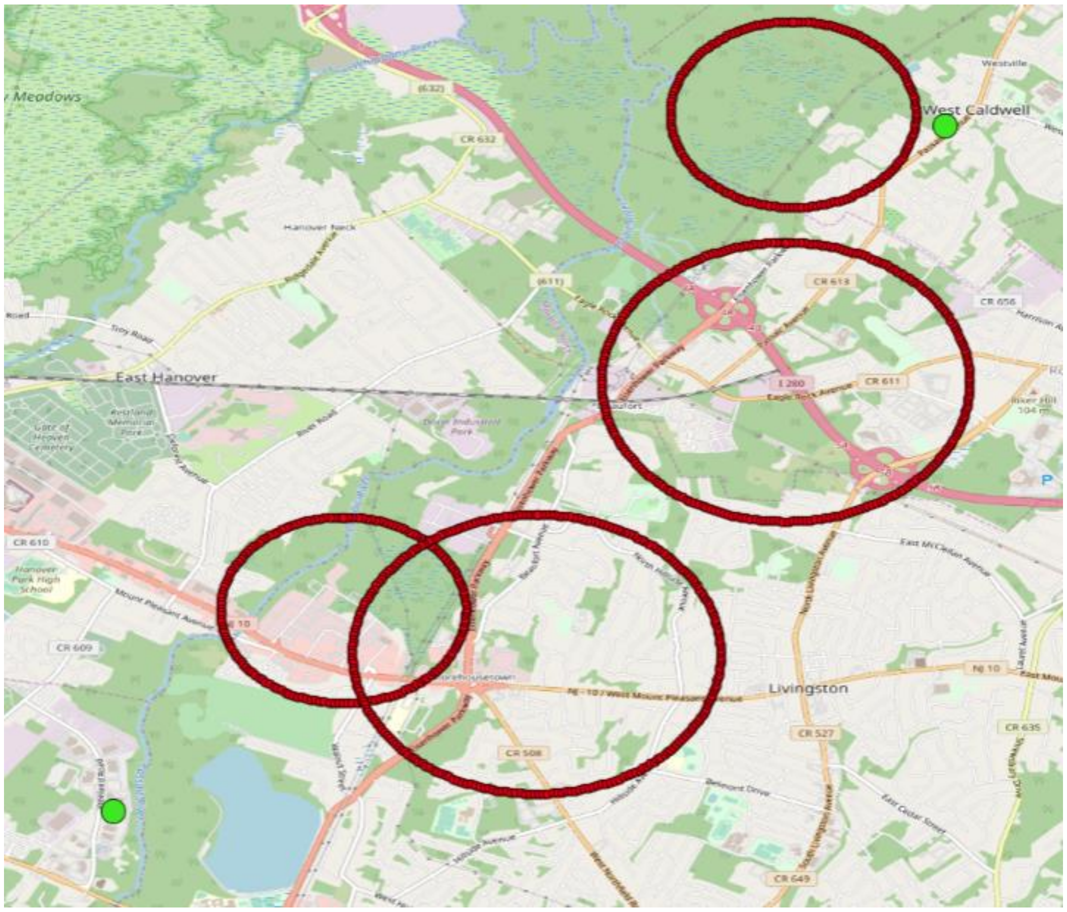
Expand X

{S,A,X,E} is the best path... (costing 7)

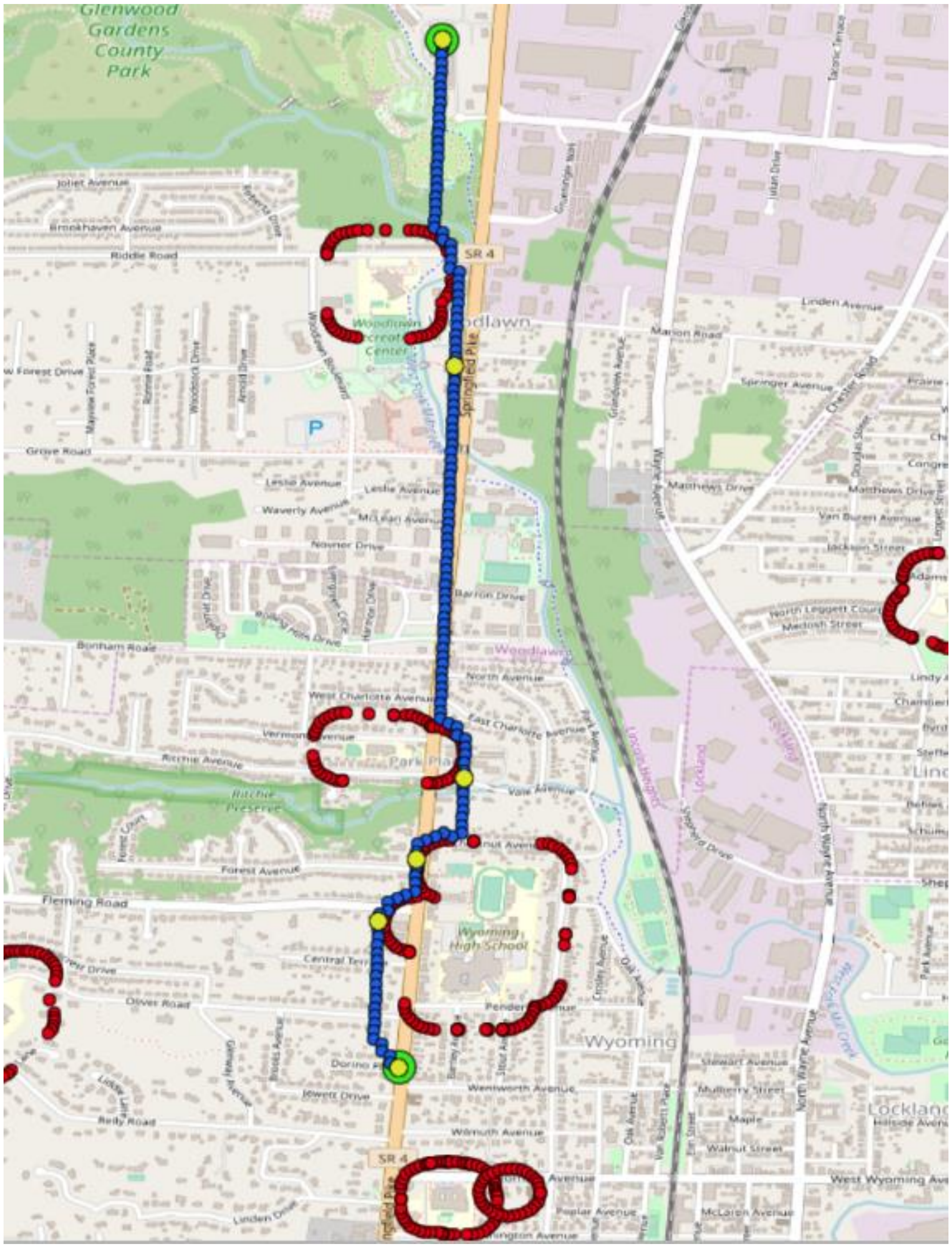
4 DRONE EXPRESS PATH FINDING ALGORITH

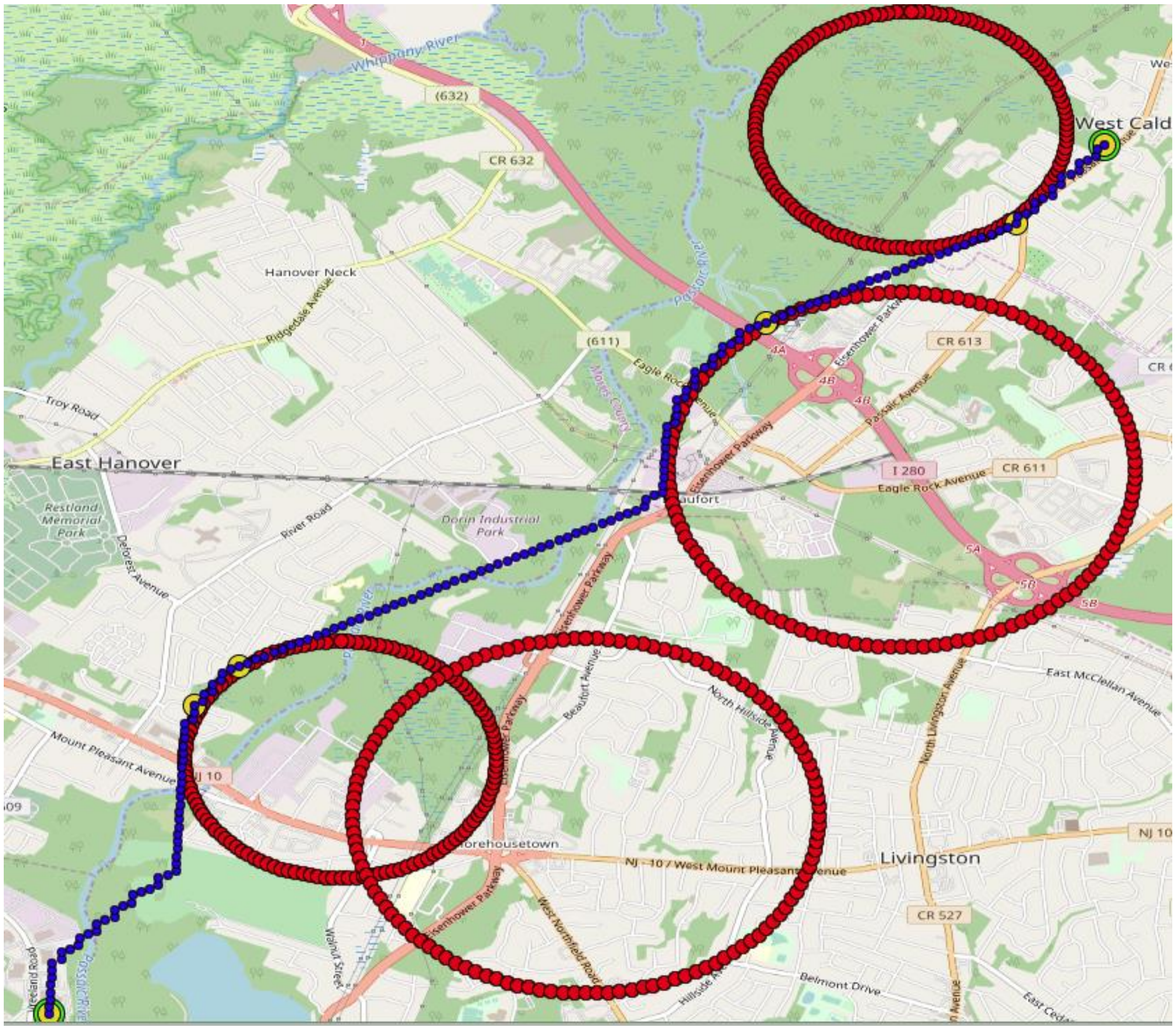
In the current use case, only the **start** and **end** *locations*, and obstacles are defined. These end point *locations* are not *nodes*, and have no edges associated with them. Therefore, there are no paths connecting the two end locations with each other nor any intermediate node. In the algorithm, the end point locations are converted into nodes and intermediated nodes/edges are created.

Intermediate nodes creation is the main bulk of the algorithm and remains proprietary. The algorithm takes into account both time complexity and, because often times the program will run on the drone with limited memory, space complexity as well. After the intermediate nodes are created, the algorithm traverses through path selection starting with the beginning node just like A*. With each node activated, cost is calculated. In our implementation, cost is determined by several influences such as edge cost, destination heuristic, and other contributing factors.



After the main algorithm is complete, the granular *full path* is found. Depending on the inputs, it might look like the image below. However, this full path may not be the shortest possible route for a drone to take because of proprietary optimizations done earlier. It is often close, but there can be further optimization done in order to skip over unnecessary nodes. A new *smooth path* is constructed by calculating which nodes can be skipped while still ensuring that the path is valid and avoids obstacles. The smoothed path is returned and the main program is finished. Examples of calculated flight paths are shown below. The green circles are the two end points, the red circles are the obstacles, the blue line is the full granular path, and the yellow is the smoothed path waypoints for the drone.


Glenwood Gardens County Park
SR 4
Springfield Pike
Woodlawn
Wyoming High School
Wyoming
Park Place
Ritchie Preserve
Lockland


Whippany River
(632)
CR 632
Hanover Neck
Ridgedale Avenue
(611)
Eagle Rock Avenue
Troy Road
East Hanover
Restland Memorial Park
Deforest Avenue
River Road
Dorin Industrial Park
Eisenhower Parkway
CR 613
Passaic Avenue
I 280
CR 611
Eagle Rock Avenue
West Cald
East McClellan Avenue
North Hillside Avenue
Beaufort Avenue
Mount Pleasant Avenue
NJ 10
Morehousetown
NJ - 10 / West Mount Pleasant Avenue
Livingston
CR 527
North Livingston Avenue
West Northfield Road
Walnut Street
Belmont Drive
Hillside Avenue
NJ 10
Passaic River