

IAJ 2nd Delivery

1. Strategy used for the Cluster Graph  
     
   The Cluster Graph Gateway Distance Table was created by running the Node Array A\* algorithm for each gateway, using all existing gateways (even itself, though this may be redundant) as endgoals. This allows us to have a complete array of values without any null indexes.  
   The Quantize function of Cluster Graph was implemented by comparing the local position of the given node with the min and max values of every cluster until the following condition is true:  
    *cluster.min <= node.position <= cluster.max*  
   This means that any node that is not within the range of a cluster is not associated with a cluster (this will be important for the Gateway Heuristic).  
   Note that if a node is within range of two clusters, only the first cluster found by the iteration is considered.  
   For the Gateway Heuristic, we consider that if both nodes belonged to the same cluster or if any one of them did not belong to any cluster, we calculate the Euclidean Distance between them (as suggested by the faculty). In case the nodes belong to different clusters, we calculate the values of:

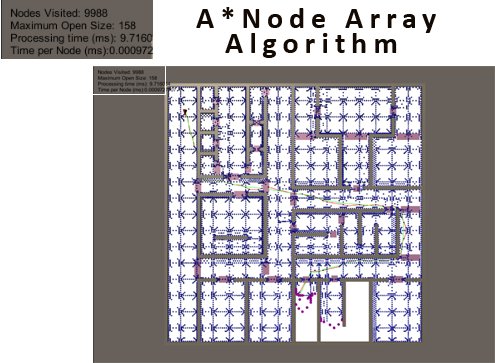
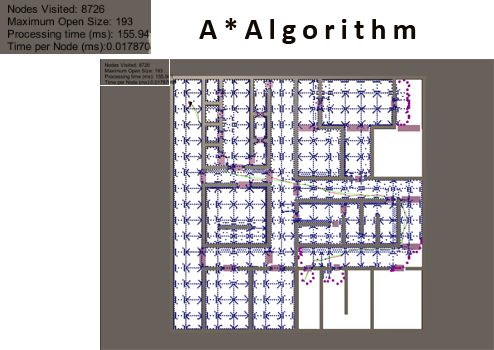
*h(startNode, startGateway) + h(startGateway, endGateway) + h(endNode, endGateway)*

, for all gateways of both the start and goal node, compare them, and return the minimum value.

# 2.

# 3. Comparing the pathfinding algorithms

To test the performance of both versions of the A\* Algorithm, we ran each of them separately using approximately the same Goal position. The results were the following:



The number of Nodes Visited and Maximum Open Size are very close between the two algorithms, and so we can dismiss the fill difference as irrelevant.

The processing time of the A\* algorithm with Node Arrays is multiple times smaller than with unordered lists, and even though the number of Nodes Visited is higher in the Node Array version, the processing time per node is still much lower.

The faster processing time is a result of the much shorter seek time for any given node, which in turn makes the testing of its state much faster aswell. However, the Node Array A\* algorithm must expend extra memory to store the pre-processed node graph.

So, in short, if you have the memory to expend and want faster processing time, the Node Array version of the A\* algorithm is much better.