|  |
| --- |
| Figure 1: Spiraling Search Method |

The first search method, dubbed the “Spiraling Search Method” (figure 1), consisted of starting at the lower left corner and of going around the search field in a square spiral motion. Every lap the robot would shorten its search area by 10 centimeters (to avoid missing any cubes between the spirals) and continue scanning in front of it until it would either find the targeted cube or reach the middle.

|  |
| --- |
| Figure 3: Snake Search Method |

|  |
| --- |
| Figure 2: Possible Cube Positions in a Square-Type Search Method |

This method despite sounding good at first glance offered many shortcomings- first one being the time it would take to complete. The distance to travel is simply too great. Another shortcoming would be the difficulty of coding such a method as we would have to consider a multitude of scenarios each under very varying conditions- when a cube is reached, the robot must check the colour and then round it to continue on its path; a plausible unfortunate scenario is when there is a cube next to the other undetected one, the only way to avoid the secondary cube would be to place a secondary-side sensor for object rounding. Though how would the robot also know it is detecting two cubes so it can check the second one? Furthermore, when a cube is detected it can be in one of 8 positions in a square-type search method (figure 2), it can either be on a side of the square paths or close enough to the corner that rounding the cube completely would put the robot off course. Finally, the cube locations would also have to be stored to avoid detecting the same cube multiple times.

The second search method is the “Snake Search Method”- consisting of going through the grid in a zig-zagging 90-degree pattern (figure 3) with each path being separated by a 10cm distance. Seeing as in theory this method uses the same detection type with a front facing ultrasonic sensor just like the “Spiraling Search Method” it brings in fact the same problems. Issues arise when the cubes are next to each other, and once again the multiple possible cube positions when it comes to the path complicate the pathing algorithm by a huge amount. Though, this method also greatly reduces the odds of find the same cube twice and reduces the path length.

|  |
| --- |
| Figure 4: Zigzagging Search Method |

A third search method was as well proposed, consisting of going through the field in a full zig-zagging pattern (Figure 4). The robot would detect the cubes in front of it just like in the previous two methods and would round the detected cubes from their left using a secondary ultrasonic sensor. This searching method proves itself to be a slight improvement over the 3rd method time-wise as the path distance is reduced. Though it may prove to be even more difficult to code, seeing as the path is not constituted of 90° simply angles it is harder to verify at which point on the path to robot is, since its pathing will be often broken when rounding the cubes that it will have detected in from of it. Furthermore, the same issue arises as with the last two methods, it being when there is a cube behind another.

|  |
| --- |
| Figure 5: Square Driver Search Method |

The final method, which is the “Square Driver Rounding” consists of going around the outer perimeter of the search area with an ultrasonic sensor pointing inwards which will detect cubes that are within the halfway point of the search area. Once a cube will be found the robot will go to it, verify its color and return to the outer perimeter rounding. This method that we attempted to implement offers a huge improvement in terms of reliability and time over the other 3 methods. The pathing is much shorter and if a cube is behind another the robot will detect it when it is going along another side of the search area. To avoid having to verify a cube multiple times the cube’s coordinates are stored in an array (the cube’s coordinates are an estimate, seeing as the robot can be looking at a corner of the cube the worst-case scenario would be that the cube’s center is away and so their location will be stored as circles of a radius of taking in consideration other possible unexpected errors (the number is to be further tweaked). The single downside of this method is when 5 cubes are to be placed in a cross like pattern which would avoid detection of the middle cube.