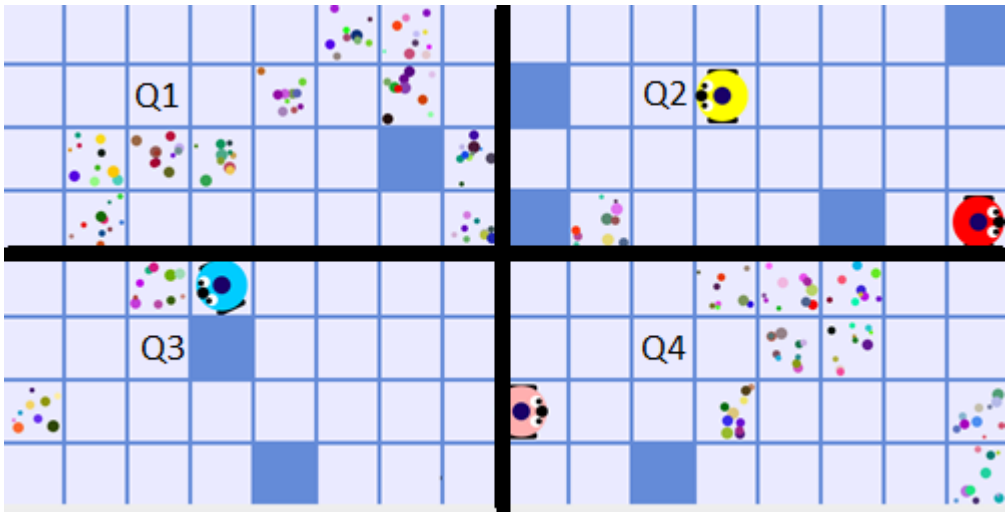
COMP30240 Assignment 2 “Vacuum World” Report

This group project was completed by a team consisting of Kealan McCormack, Lukasz Filanowski, and Gerard Colman. See README for student numbers. In this assignment we were tasked with creating multiple agents to clean up a virtual world called “VacWorld”. We used the Astra programming language to accomplish this. Over the course of this report, I will discuss: Our General Coordination Method, Bot Movement, Bot Seeking Dust, and Bot Obstacle Avoidance.

General Coordination Method

A coordination method in the scope of this project is essentially how the bots work together to clean the board most effectively. A good coordination method should have methods for avoiding obstacles & avoid other bots as well as cleaning dust. We went through multiple different iterations of our Coordination method. The first iteration was quadrant based. Essentially each bot would have its own quadrant to clean and would go down every 2nd row as a bot can see to its front left, front forward, and front right.

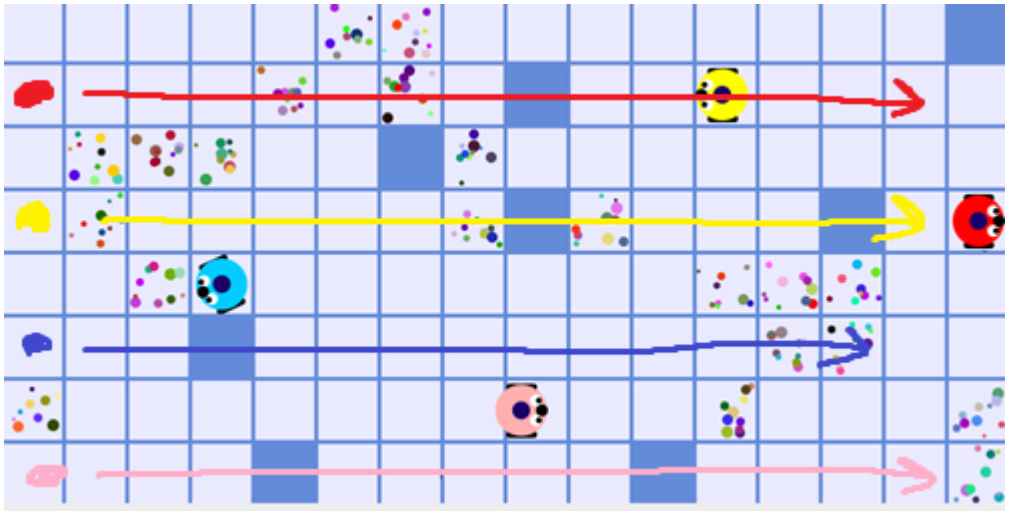
See diagram below for demonstration:



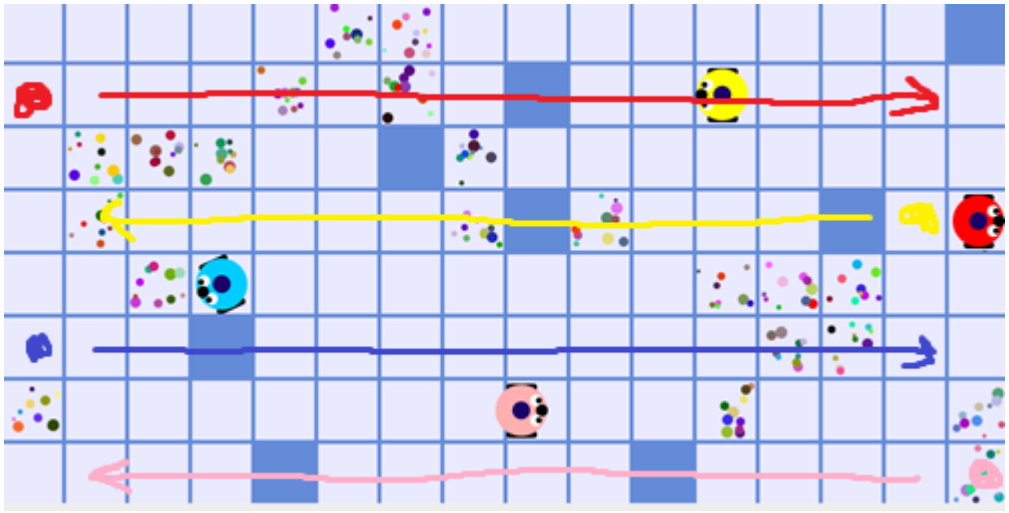
We encountered a few issues with this method. Mainly it was too complicated to implement. Another issue was the bots kept colliding with each other and finally it was too inefficient as there were too many calls to the movement function.

The next coordination method we tried was a “Row Method”. The premise of this method is that each bot can see to its front left, front forward, and front right.

So, the bots would centre themselves in the middle of 3 row wide segment and sweep from one side of the board to the other as seen in the diagram below.



The main issue with this method is that the moment there was an obstacle they would collide and break. The solution to this was to have them start on alternating sides as seen the diagram below:



In the end, we decided to go with the Row Method, starting on alternating sides.

Bot Movement

Our bots movement is split into 6 “movement” functions and “moveForward” function and a “goForward” function. The 6 movement functions. The 1st movement function simply calls movement and is a generic function. The 2nd is like an initializer after the bots have moved to their starting positions. This function sets the x to 15 in order to make it sweep across the board. The next function is meant for when the bot is “below” its y coordinate. And then calls moveForward(). The next movement rule is meant for when the bot is “above” its y coordinate. The two movement rules are for when the bot is aligned on the Y to its target by not X. It moves towards the X. two rules for two directions. The moveForward function simply rotates the bot to the direction it is going to move and then

checks if the bot is on dust and if so cleans it. This moveFoward function then calls the goFoward function which firstly seeks dust. Then querys the forward string and moves it forward one if its empty or has dust on it. If the forward square contains dust we clean it. If theres an obstacle we call the obstacleAvoid function.

When we call seekDust() we pass in the square to the left, the right, in front, forward right, forward left, and here square. We then check if these contain dust. If so we clean it, and move back to the original position.

In the obstacleAvoid method we check where the obstacle is move, move out and move forward until theres no obstacle and move back.