Multi-Agent Systems Project

Shane Cooke – 17400206

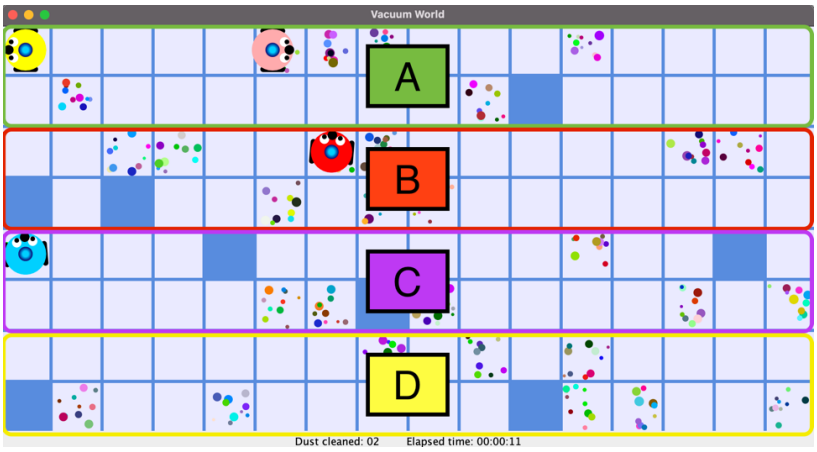
Introduction

The project that I have created for this assignment involves four VacBots (Decco, Harry, Henry and Lloyd) and their task to clean all of the dust from the VacuumWorld that they reside in. In order to complete this project, I used ASTRA programming language along with Java for custom external modules. The README file that I submitted with the project contains all of the details on how to run the project, and also all of the details about the file structure of the project.

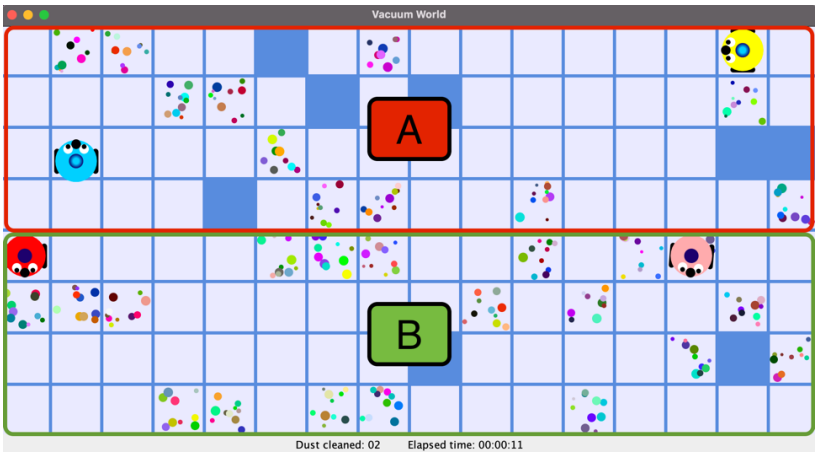
Plan

My plan for this project was to create a system where one VacBot acted as the "Leader" or "Main" VacBot and dictated to the other VacBots where they should and should not focus on. At the very start of a run, each VacBot would send the Leader VacBot a message containing which area of the grid that it had spawned into. The Leader VacBot would then take in this information, and dynamically tell each VacBot which area of the grid that it should focus on cleaning. The Leader would ensure that each area had VacBots evenly distributed in it, meaning that one area would not be overly focused on compared to other areas.

Initially I had intended on each VacBot focusing on an area two spaces deep or two Y coordinates deep, as shown in the diagram below:



However unfortunately due to time constraints and other factors, I had to settle on each VacBot focusing on an area four spaces deep or four Y coordinates deep, as shown in the diagram below:



Implementation

To implement my plan, I first decided to elect Decco as the Leader VacBot. Upon running the application, each Minion VacBot instantly sends a message to Decco with the area of the grid that they have spawned into ("top" or "bottom"). Decco then receives these messages, and after deciding which area of the grid he will focus on cleaning, sends each Minion VacBot an instruction saying which area of the grid that they should focus on cleaning ("top" or "bottom").

Decco assigns Minion VacBots areas by taking into account the area that each VacBot spawned into, and how many VacBots are currently occupying an area. If a VacBot spawns in the "top" area, Decco is more likely to assign him to the top area, and if a VacBot spawns in the "bottom" area, Decco is more likely to assign him to the bottom area. If Decco has already assigned two VacBots to one area, he will assign the remaining VacBots to the opposite area, regardless of where they spawn into the grid. This ensures that two VacBots are focusing on each area of the grid at all times.

Each VacBot will focus on the area of the grid that was assigned to them by Decco. While they are occupying the area that they have been assigned to, the light on their head turns on, and when they stray outside of this area, the light on their head turns off. When a VacBot strays outside of the area that they have been assigned to, they will be instructed to return to their area and continue cleaning. VacBots can still clean outside of their area for efficiency purposes, however the vast majority of the cleaning they do will be inside their own assigned focus area.

While each VacBot is inside their focus area, they will move around almost freely. If a VacBot comes across an obstacle or another VacBot in its path, it will be instructed to turn. The direction a VacBot will turn is random, in order to ensure that all parts of the area are eventually cleaned. To ensure that no VacBot gets stuck in a never-ending loop or gets trapped in one part of the grid, there is a one in twelve (1/12) chance that while no obstacle or VacBot is in the way, the VacBot will still turn. Without this feature, I would often find VacBots getting trapped in one part of the grid, or getting stuck in a never-ending loop, and the "random" turn signal prevents this from occurring.

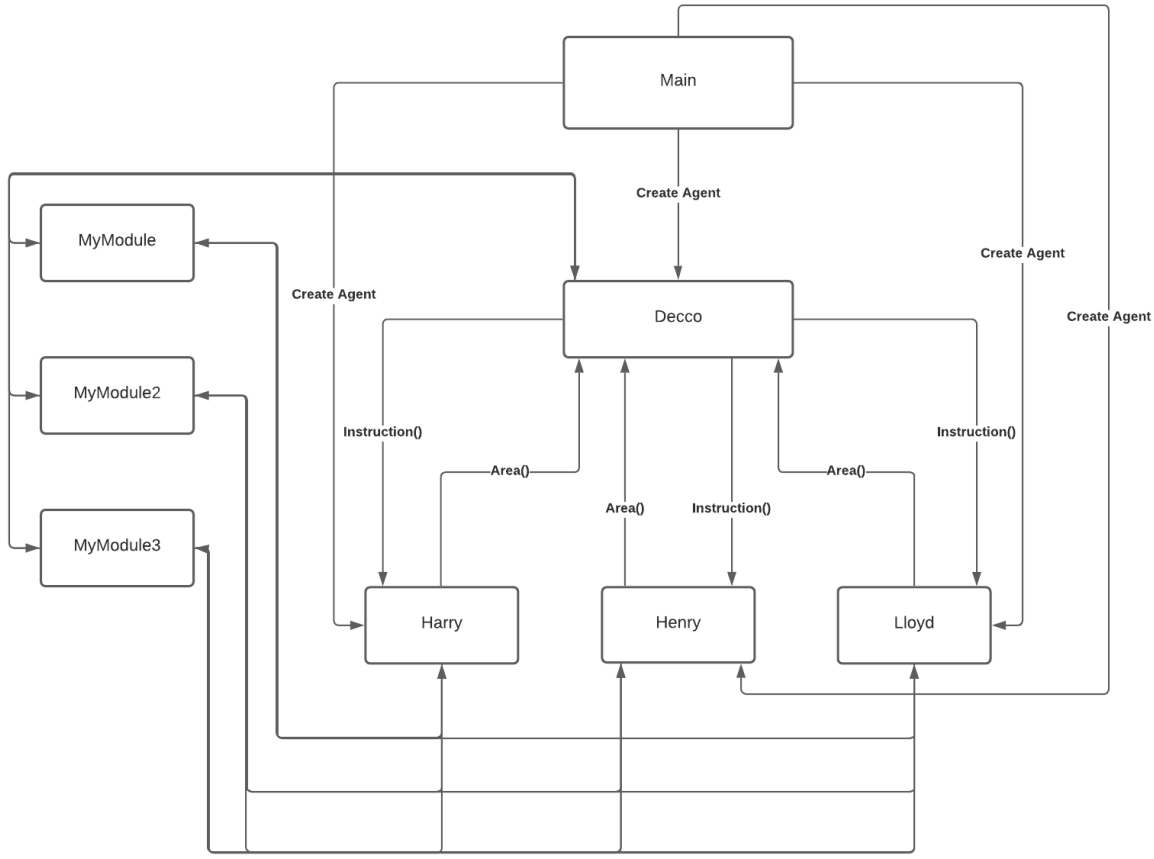
I decided to use three custom Java modules in this application (MyModule, MyModule2 and MyModule3) and each carries out an important function. MyModule is used to generate a random turn direction ("right" or "left") and is vital to ensure that the VacBots cover all areas of the grid. MyModule2 is used to pause the application for a set amount of time and is vital in order to allow the percepts time to update and to allow the VacBot to adjust to these percepts. MyModule3 is used to generate a random number which is used to give the VacBot random turn signals and again, this is vital to ensure that the VacBots cover all areas of the grid and don't get trapped.

System Architecture

Now that the main premise behind the application is understood and the main behaviours of the VacBots are understood, I will explain the system architecture of the application.

As you can see in the system architecture diagram below, the application begins with the Main class creating each agent (Decco, Harry, Henry and Lloyd). Once these agents are created, the three Minion VacBots send the Leader VacBot a message that contains the area of the grid that they have spawned into. Once the Leader VacBot has received these messages and decided which area of the grid that each VacBot will focus on, he sends an instruction message back to each Minion VacBot. This is not a one-time message, and this communication channel between the Leader VacBot and the Minion VacBots remains open for the entirety of the runtime of the application.

Each of the four agents queries the custom Java modules whenever it is necessary. The agent will specify the input and the Java module returns the required answer or function, which is why two way arrows can be seen indicating the connection between the agents and the custom modules.



Results and Reflections

Upon finishing my application I am quite happy with the results. The application can clean the entire grid in as short as 7 minutes, with an average completion time of around 9-10 minutes. Decco works very well as the Leader VacBot and always ensures that the Minion VacBots are assigned to the best area, and that no more than two VacBots will be assigned to one area. The messaging between VacBots is extremely fast and no messages are lost in transit.

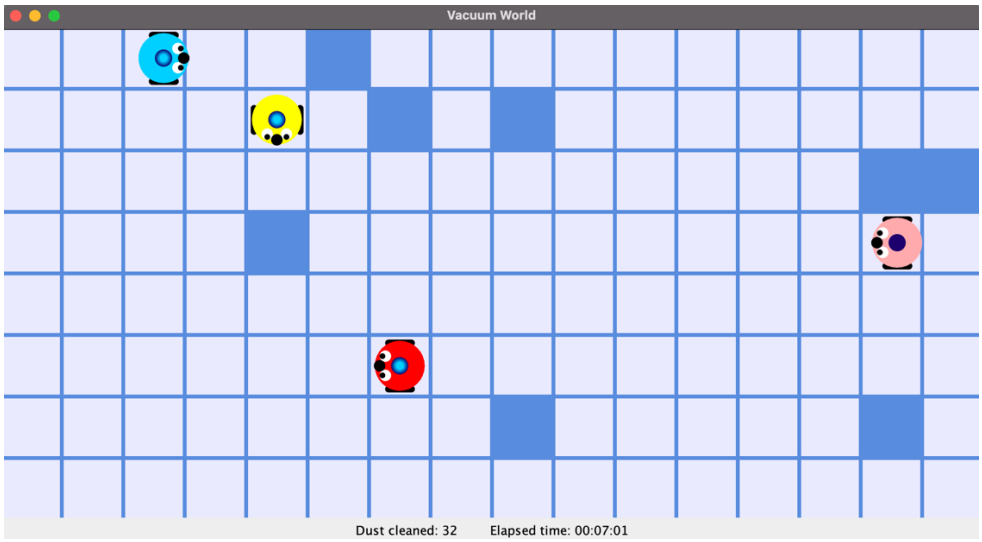
The VacBots behave almost exactly how I would like them to, the only thing I would like to improve is their reaction time to the percepts. Sometimes an Agent can be quite slow to realise that it is outside of its focus area, however it will always return if it strays. The VacBots don't get stuck in never-ending loops or get trapped, and always seem to cover a very good ground area. They respond well to obstacles and other VacBots, and make the necessary adjustments to their path to continue cleaning.

If I could start this project again, I would like to make each VacBot cover a single area two spaces deep or two Y coordinates deep. Unfortunately due to time constraints for this project, I had to settle for a four space deep or four Y coordinate deep area. I would also like to make the VacBots more responsive to their environment, for instance actively searching for dust in the grid or intelligently choosing which direction to turn and travel.

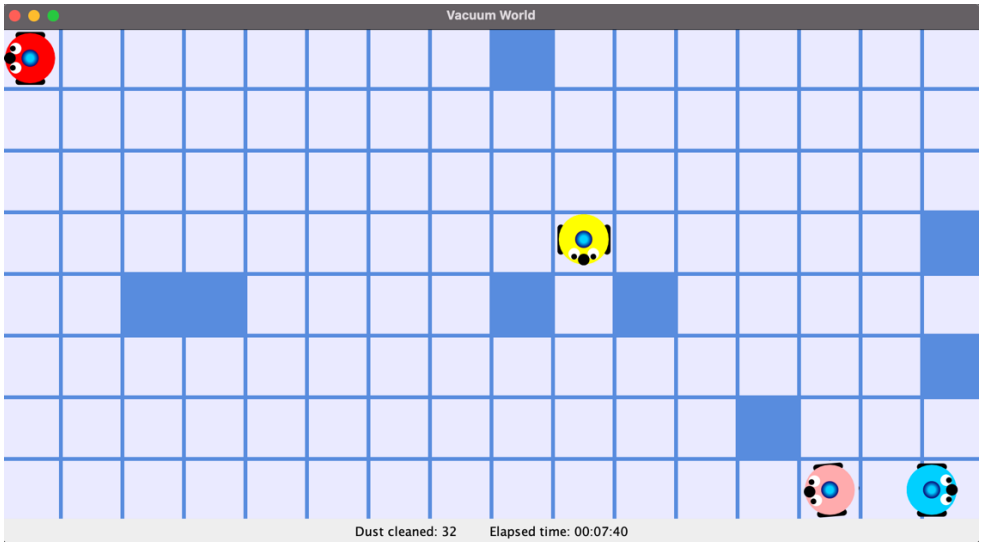
Overall, I am very happy with how this project turned out. The application runs well and never crashes, the grid get cleaned entirely by the VacBots, and the Agents are communicating and coordinating to complete the goal. Below I will include some screenshots of successful completions.

Completions

Time: 00:07:01



Time: 00:07:40



Time: 00:07:11