CSCE 274 Section 2 Fall 2018 – Project 1 – John Esco, Adam Kenvin, Changxuan Yao

Description:

Our program is consisted of three parts. The first part is the connection interface, which contains all methods required in Task 1, including connecting to the serial interface, sending of commands, reading of data, and closing the connection.

The second part is the other interface for the robot, which contains all methods required in Task 2, including controlling the state of the robot, reading the state of the buttons, and sending a drive command. The second interface imports methods from the first (connection) interface in order to read data (uses opcode 142 and calls packet 18) and send drive command (uses opcode 145).

The third part is the drive code for the robot. It imports both the connection interface and the robot interface. It first uses the connection interface to close the connection and then open it to establish a connection to iCreate 2. Then, it uses the robot interface to start iCreate 2 and set it to passive and safe mode. After that, there is an infinite while loop that breaks when button is press, and thus proceed to performing the drive. The actual drive code is located in a for loop that runs a total of six times, which covers the six sides of the hexagon. During each straight drive and turning drive, a for loop is implemented to check button press, and adds 1 to a constant x that contains the times the button is pressed. If the button is pressed, the robot will stop when it reaches the current goal vertex and the while loop will start and wait until the next button press to continue executing. When the clean button is pressed again, this while loop breaks, and iCreate2 proceeds to the next action. After the for loop is completed, the robot is set to stop using the drive() method and setting all its parameters to 0. At the end, the stop() method is called from the robot interface, and the close() method is called from the connection interface to close the connection.

Evaluation:

Our program worked for the most part but has some notable bugs. The readButton() method was unreliable and highly dependant upon the individual iCreate 2 that it is used on. When the robot cooperates, the readButton() method returns the correct values corresponding to the correct button presses. The drive() implementation works flawlessly. On a taped out hexagon, the iCreate perfectly traced the shape. However, due to the previously stated inconsistency with the readButton() there were issues with getting the iCreate to stop and start again. This was remedied by implementing a for loop which read the button every 0.015 seconds for the full duration. If the clean button was pressed then a global variable would be set to true and the iCreate would stop. Once this was implemented, the iCreate functioned correctly and would stop or start with each press.

Allocation of effort:

Each member of the group contributed equally to the final product and met frequently. Below is a more specific breakdown.

Adam Kenvin:

Helped with coding the connection interface and the functionality of the clean button stopping the drive function at the next vertex.
Changxuan Yao:
Helped to code the drive() function. Calculated the values for the parameters correctly resulting in a near perfect hexagonal shape. Contributed to the project report.
John Esco:
Helped to code the readButton() method which returned the correct button being pressed. Contributed to the project report.