FLORIDA TECH

FLORIDA'S STEM UNIVERSITY

RMC - CSE Milestone 2

Liam Sapper

Contact & Meeting Information

- CSE Project Member: Liam Sapper - lsapper2020@my.fit.edu
- Faculty Advisor: Dr. Marius Silaghi - msilaghi@fit.edu
- Client: FIT's Robotic Mining Competition team (RMC), and by extension, NASA (the host of the Robotic Mining Competition).
- Head of RMC project:
  - Sidney Causey (scausey2021@my.fit.edu) - Aerospace Engineering
- Meeting Times: Wed. 4pm-5pm; Fri. 3pm-3:30pm

Progress Matrix

TASK	COMPLETION %	TO DO
1. Implement a simulator	100%	none
2. Design test vectors for main requirements to be verified by simulator	80%	Try to create numeric measurements to be checked
3. Look up documentation of involved hardware	30%	Need to look up documentation for chosen encoder, motors, and other sensors
4. Research relevant algorithms for autonomous tasks	100%	none
5. Develop navigation system	30%	Move past pseudocode/logic phase, start building/testing base software

Task 2

- Waypoints are set beforehand
- Test Vectors:
  - Waypoint storage system is functional; waypoint information stored and retrieved
  - Set first waypoint at 0,0
  - When navigating, acknowledge waypoint reached
  - Turn in correct direction of next waypoint
  - Start moving forward towards next waypoint
  - Stop navigation once it reaches last waypoint

Task 3

- Hardware documentation: computer, motor, encoder, other sensors
- Raspberry Pi for computer, attempting to obtain one
  - Works with Python and can connect to Arduino
- Encoder: Counts number of rotations of a motor.
  - Calculating distance with info from encoder: distance = (motor rotations) * (wheel perimeter/gear ratio)
  - Works along with Internal Measurement Unit, keeping track of acceleration of bot

Task 4

- "Robot Navigation by Waypoints", 2008, Yang Wang, David Mulvaney, ian Sillitoe, Erick Swere
- Made use of both reactive behavior and deliberative planning

- Currently working on deliberative
- Want to add reactive later on

```
procedure EP/N
  begin
    if there exists a previous population with relevant paths then
      input the previous population P
    else
      initialise P
    end if
    evaluate P
    while the termination condition is not reached do
      use the operator probabilities to select an operator O
      select parent(s) for the operator
      produce offspring by applying the operator O to the selected parents(s)
      evaluate the new offspring
      replace the worst member(s) of the population P by the new offspring
      select the best individual p from P
      every n^th step
        if the algorithm is operating in an online manner and p is feasible then
          move one step along the path determined by p while sensing the environment
          modify the values in all individuals to a new starting position
          if there is any change needed to the existing plan then
            update the object map
          end if
          evaluate P
        end if
      end every
    end while
  end procedure
```

```
procedure vertex planning algorithm
  begin
    if there exists a previous population with relevant paths then
      input the previous population P
    else
      initialise P
    end if
    evaluate P
    while the termination condition is not reached do
      use the operator probabilities to select an operator O
      select parent(s) for the operator
      produce offspring by applying the operator O to the selected parents(s)
      evaluate the new offspring
      replace the worst member(s) of the population P by the new offspring
    end while
    select the best individual from P
  end procedure
```


FLORIDA TECH
FLORIDA'S STEM UNIVERSITY

Task 5

In [3]:
```
class Node:
    # For doubly linked list
    def __init(self, next=None, prev=None, head=None, tail=None, data=None):
        self.next = next
        self.prev = prev
        self.head = head
        self.tail = tail
        self.data = data

class RobotNavigation:
    Node currentNode = new Node
    def __init__():
        # Set first waypoint xy coordinates (0,0), and direction/angle.
        # Node headNode = new Node
        # headNode.data = [0,0,angle(0?)]
        # currentNode = headNode
        # NavGui thisGui = new NavGui

    def getPosition():
        # Get current position.
        # x =
        # y =
        # angle =
        #

    def setWaypoint():
        # Focus on manual first
        # When markButton = pressed, mark current xy from data given as waypoint

    def autoNavigate():
        # Follow waypoint route starting from 0,0
        # loop; while (not at last waypoint)
            # find angle for straight shot to next waypoint
            # turn robot towards waypoint
            # start moving towards waypoint
                # check x and y
                # check angle
                # if current x and y = next waypoint x and y
                    # stop
                    # mark waypoint as reached, get x + y for new waypoint
                # else
                    # continue moving
```

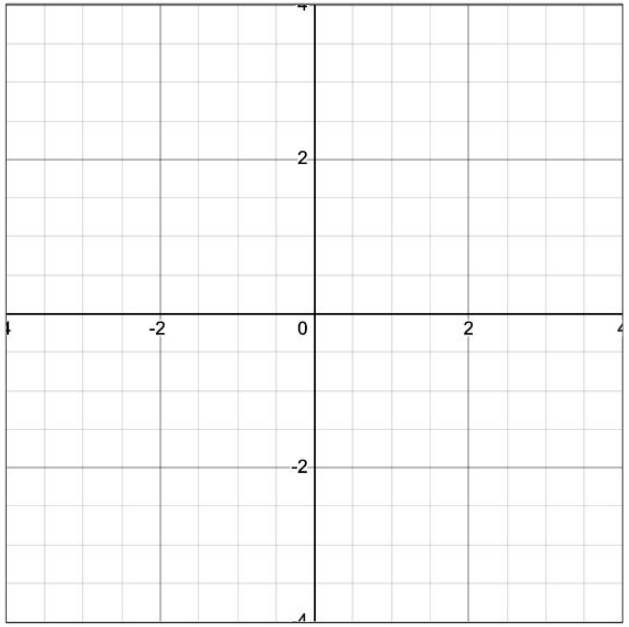
In [2]:
```
class NavGui(QMainWindow):
    # self.resize(width, height)
    # self.setWindowTitle("Muck Navigator")
    # self.loadGui

    def loadGui():
        # Set up arena/grid view
        # Set up waypoint list
        # Set up "Set Waypoint" button (might want a physical button for this though)
```

Task 5

WAYPOINT	XPOS	YPOS
1		
2		
3		
4		

MARK WAYPOINT

AUTO NAV



FLORIDA TECH
FLORIDA'S STEM UNIVERSITY

Milestone 3 Plan

TASK
1. Implement code in simulator that passes vectors
2. Implement unit tests for verifying simulated code
3. Continue researching algorithms for autonomous tasks, look up libraries for selected algorithms
4. Implement/Adjust any missing/existing techniques and tasks

M3 Task 1

- Move past pseudocode phase
- Webots simulator provides motor simulation and other libraries
- Replace simulation w/ proper outside connections

M3 Task 2

- Goes along with task 1
- Break everything up into smaller problems
  - Test separate parts of functions
  - Test complete functions separately
  - Test to make sure functions work together

M3 Task 3

- Current findings seem acceptable
- How many people have improved on this strategy? Are there more efficient methods?
- What can I do to improve the current algorithm?

M4 Task 4

- Look into more hardware documentation of confirmed motors/sensors
- Confirm I am being given correct information from sensors to use in software
- Once again, move past pseudocode phase

FLORIDA TECH

FLORIDA'S **STEM** UNIVERSITY

Thank you