Faculty Of Computer Science,
Dalhousie University, Canada.

CSCI 1108
Intro to Experimental Robotics
Mid-Term Project Report
Group-9

Names	Banner ID
1. Anantha Suraj Patnaikuni	B00845171

<u>Abstract</u>:-

In our mid-term project, we programmed the Thymio to perform particular actions such as moving when the forward button is pressed and stopping when the backward button is pressed, following the black line with a specific speed in less time and detecting as well as avoiding the green blocks, which are located at different positions on the course track, with the help of the proximity sensors ranging between zero to four and the ground sensors. We also added some constants like BLOCK, SPEED, STOP, FOLLOW, AVOID, and many more to make the robot start, stop, move, follow, avoid, and sense.

<u>Introduction</u>:-

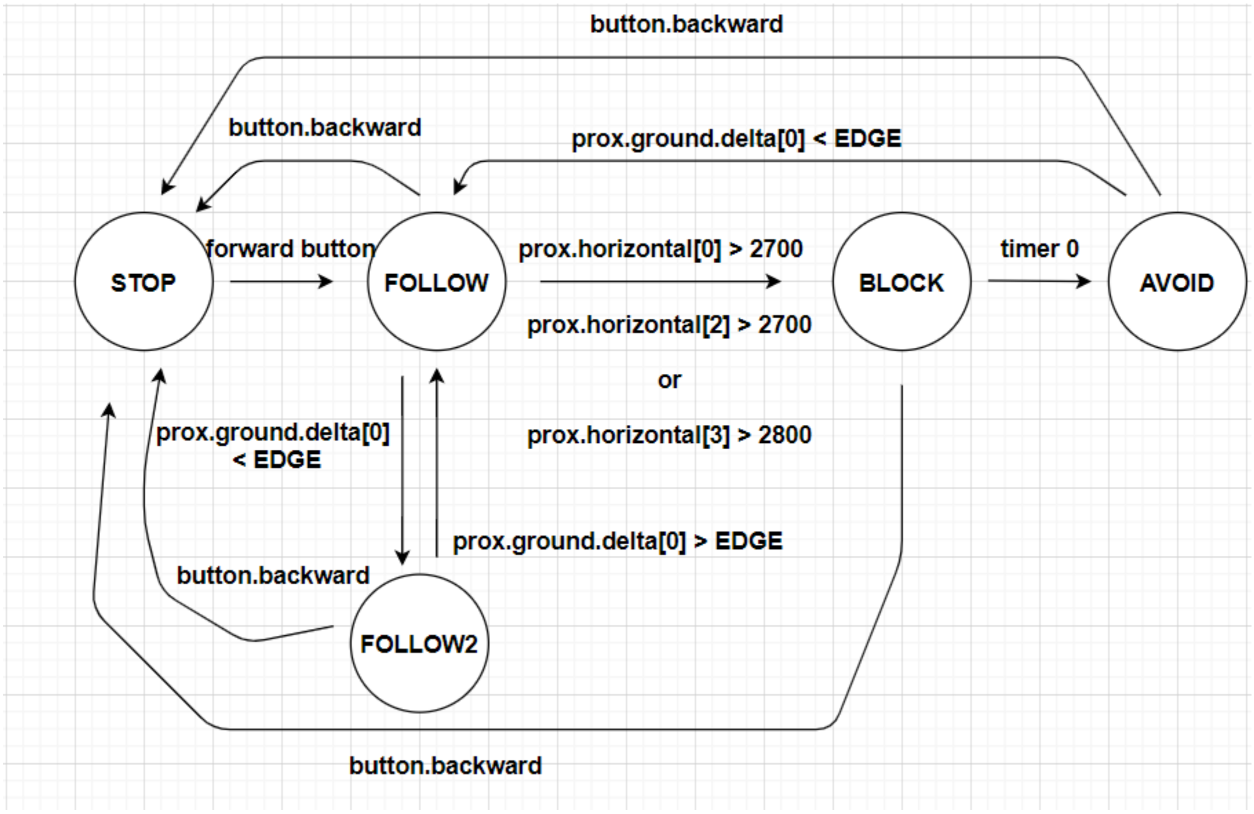
This course is intended to introduce us to the basics of robotics and understanding the basic elements that make up a robot i.e. actuators, controllers, and sensors. So, to have a deeper understanding of the same, we are using a simple form of a robot system, which is a mobile robot. Here we use a Thymio II robot. On this robot, we were assigned a task to program it so that it completes a course track made up of black tape and several hurdles(blocks) placed at different positions along the track. Coding the Thymio robot to follow a line and avoid an obstacle is relatively easy compared to deciding the value of speed and turn period parameters as it involves some kinematics. For example, we need to have a proper value of speed so that the robot does not skid off its expected line of motion while it turns around an obstacle. Another major problem we encountered during our project was controlling the robot's speed while taking turns and also at sharp-edged. We needed to adjust the turn time period along with speed as much time could not have been wasted in turning around the obstacles.

Background:-

Most important aspect of the project was to identify where the different sensors were placed on the robot and which sensors should be used to detect the black tapes and the obstacles. Robotic sensors mainly give the robot a path to move and to sense the objects according to the environment. To program the Thymio robot we used the IDE named Aseba. This language is simple and easy to understand and follow. It has various commands to read sensor values, which are divided into two types: ground proximity sensor values and horizontal proximity sensor values. Moreover, we had to measure the ground sensor values by calculating the difference between the amount of light reflected and the amount of ambient light, which is called delta. By this, we had to fix an approximate value, called the threshold, above which the robot would change its line of motion. After doing this, we had to introspect about the physics of movement. For example, the second hurdle in the course track needed the robot to make a sharp turn. We had to adjust the speed of the wheels and the threshold value of one of the sensors so that it could turn around the obstacle without colliding with it(refer Figure 1). Also, at some obstacles, the robot was making a large turn, which resulted in extra time being taken by the robot to come back on its track.

Program Description:-

The program starts by declaring variables and constants which will be used throughout the program. We have declared *min, max, mean, state* variables and some constants for *SPEED* and for other the various states the robot is in when it executes different events. Initially, we keep the motor speed to zero and the state to STOP which has a constant value of 1. The robot will remain in the same state when the backward button on the robot is pressed (*onevent button.backward*). When the forward button on the robot is pressed(*onevent button.forward*) the motor speed is set to *SPEED* i.e. equal to 800mm/sec. And the state is changed to *FOLLOW*. The robot starts moving in a straight line with the target speed of 800mm/sec. The next part of the program explains how the proximity sensors are programmed which made our robot move exactly the path which was expected from us during the competition day. Here we set the robot to different states based on the readings of the sensor. Lines 53 to 62 of the code explains as to why the robot follows the black line. When the ground proximity sensor(prox.ground.delta[0]) is less than the constant value *EDGE*, the left motor moves with *SPEED* and the right motor is set to zero (results in robot turning right) and the state is changed to *FOLLOW2*. The lines 58 to 62 do the same job but here the robot turns left and the state is changed back to *FOLLOW*. Thus the robot continuously switches between the two states until it encounters an object. Lines 27 to 40 perform the same task for three different horizontal proximity sensors [0], [2] and [3]. When the sensor readings are above their respective threshold values, the left and right motors rotate in the opposite direction so that the robot turns immediately towards the left and the state of the robot is set to *BLOCK*. Also of the 5 horizontal sensors, the sensor having minimum, maximum and the mean reading of all the sensors is stored in *min, max* and *mean* variables respectively. The [0] sensor is for avoiding the 2nd hurdle(refer figure 1), [2] for the objects directly in front of the robot and the [3] is for the 3rd hurdle(refer figure 2). So in lines 42 to 52, when the state is *BLOCK* and *max*=0 (i.e. nothing in front of the robot) the robot moves along a straight with speed 620mm/sec much less than the constant *SPEED* 800mm/sec for a time period of 250 milliseconds and the state, is set *AVOID*. When the timer turns zero (lines 64 to 67), the speed of the right motor is halved as compared to the value of left motor speed. This makes the robot turn right so that it can regain its path (the black line). In lines 48 to 51, when the [0] proximity ground sensor is less than the *EDGE* value and the state is *AVOID* the left motor moves at constant *SPEED* (800mm/sec) with no speed of right motor. After this, the state is set to *FOLLOW2* so that the robot sets the speed of the right motor to SPEED and consequently robot starts following the black line again by switching between the two states repetitively as seen in lines 53 to 62.



Transition diagram of the robot function

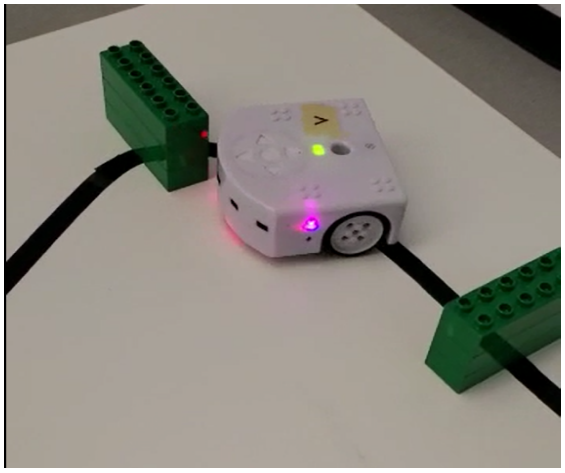


Figure-1 Robot at Second hurdle

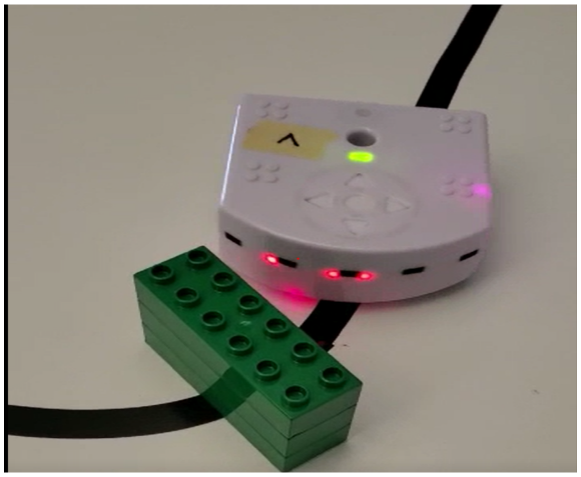


Figure-2 Robot at Third Hurdle

<u>Results</u>:-

On our presentation day, we had gone through some rough testings on our robot in order to make it work better like, increasing the speed of the robot a bit more apart from its actual speed which we kept in the initial stage and also by adjusting the speed of the robot at sharp edges so that it can move slowly without deviating from the black line. In our first attempt, we made the speed of the robot to 550mm/sec. With that speed, our robot passed all the hurdles without dislodging any of the green blocks in 48 seconds. In our second attempt, we adjusted the edge values and as well as we increased the speed of the robot to 650mm/sec and even in this attempt also our robot passed all the hurdles without hitting anything within 47 seconds. Finally, in our last attempt, we tried to increase the speed to 800mm/sec and we did not change any of the values apart from that. During this last run, our robot passed 80% of the hurdles without dislodging anything, but at the last block it started rotating by deviating from the black track. Due to this reason, we were disqualified in our third attempt. After that, we got to know that we have to adjust the edge value as well whenever we are increasing the speed of the robot. When we increased the speed value of the robot in our second attempt it did not make any difference to the robot edge values. But when we made the speed of the robot from 650 to 800mm/sec, it made a big impact to the edge values of the robot, which caused the robot to lose its line in the third attempt.

<u>Conclusion and Future Work</u>:-

Overall, we made our robot work in a way as per the instructions given to us by our instructors like moving on the black line, passing the obstacles by deviating when it senses the hurdles and completing the task in the least time. Due to this reason, we have achieved an A+ grade in the competition. If we would get more time, we would make changes to the edge values of the robot because as we made the speed of the robot to 800mm/sec in our third attempt and we did not bother about the edge values, which made a huge impact on robot functioning during the run. So, if we put the appropriate edge value to the speed we set in our third attempt (800mm/sec) we would have completed the task in about 40 seconds. We will improve this if we get more time.

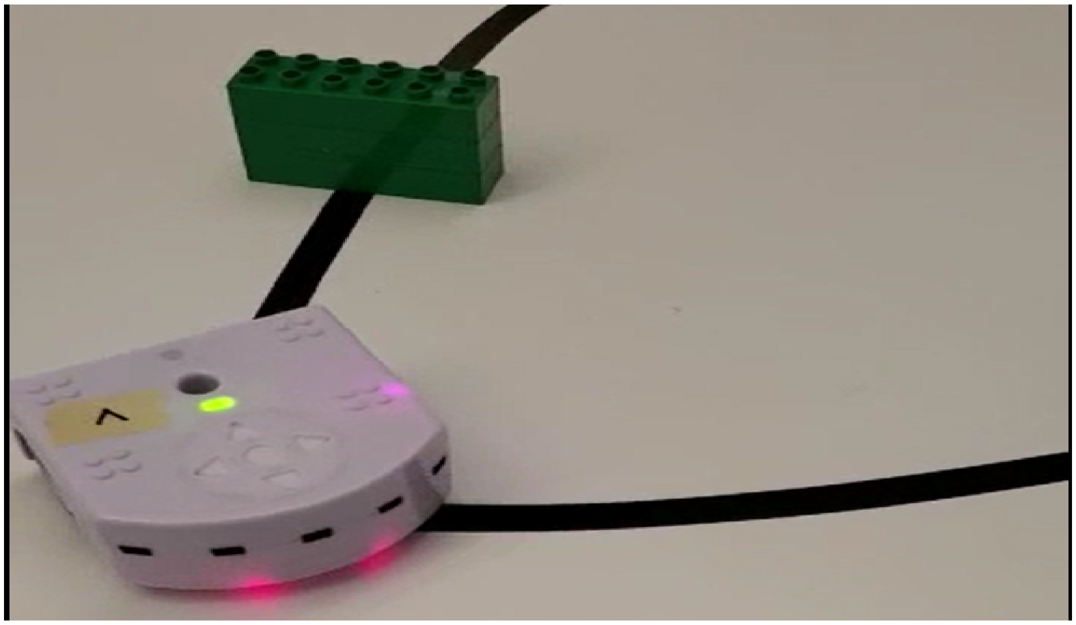


Figure-3 Robot turning at the sharp corner of course track after the fourth hurdle.

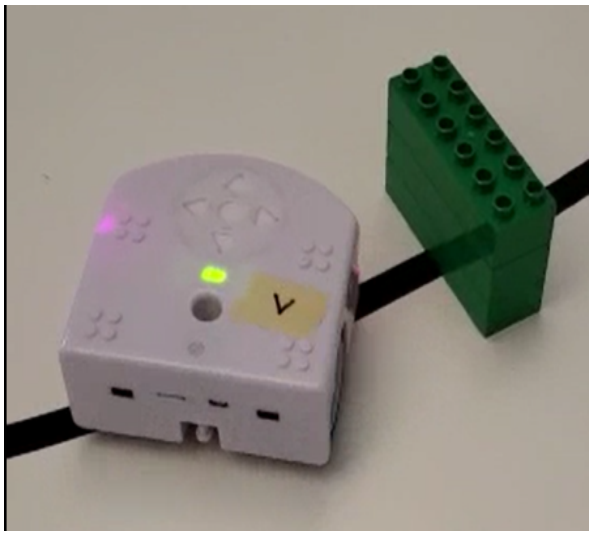


Figure-4 Robot deviating the last block.

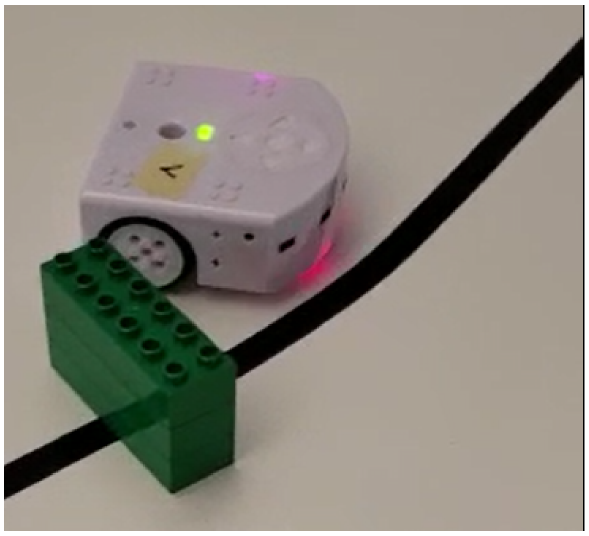
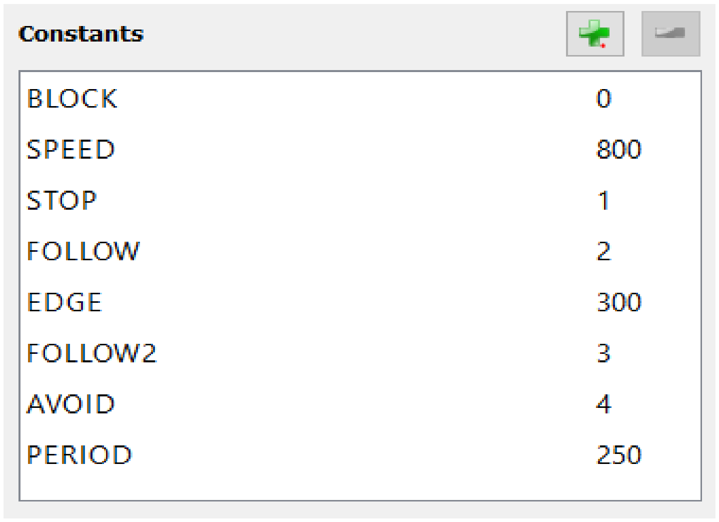


Figure-5 Robot retrieving the line hurdle after avoiding the hurdle.

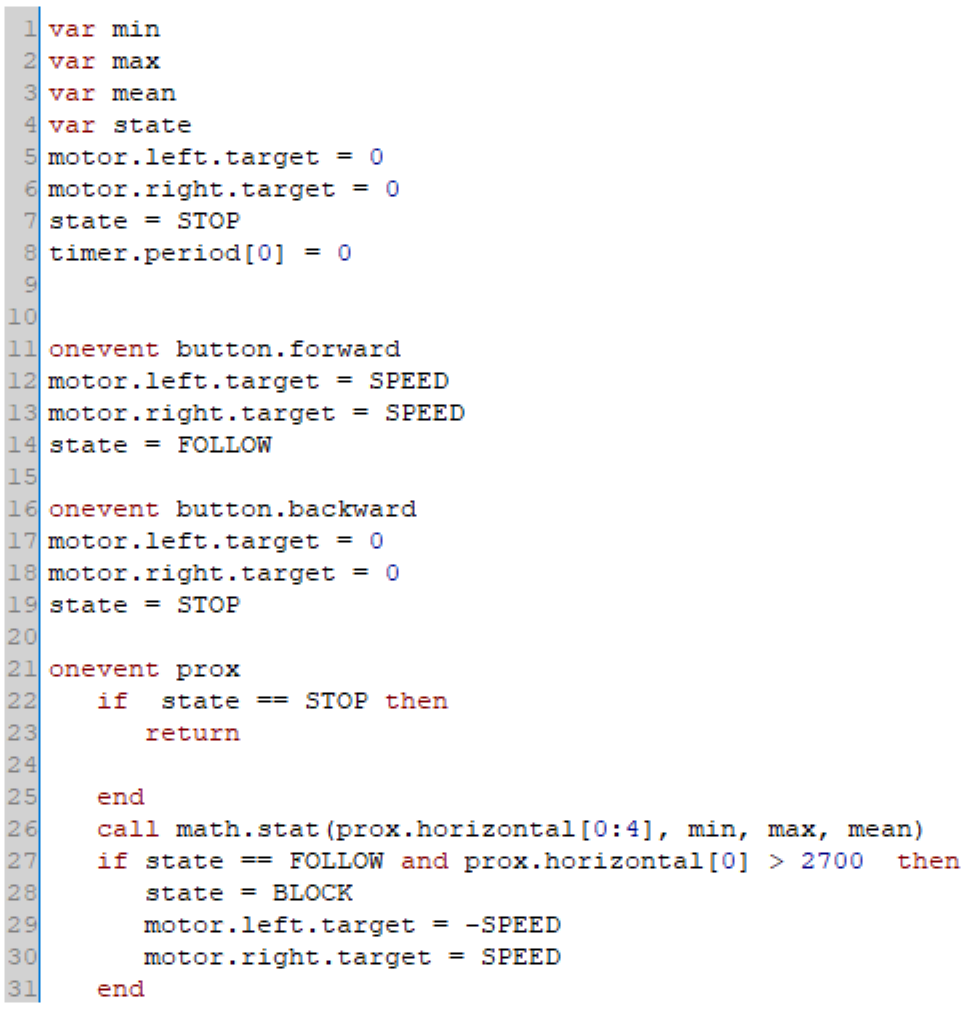
<u>The program to make the robot move and avoid all the Blocks</u>

Constants Declaration:-

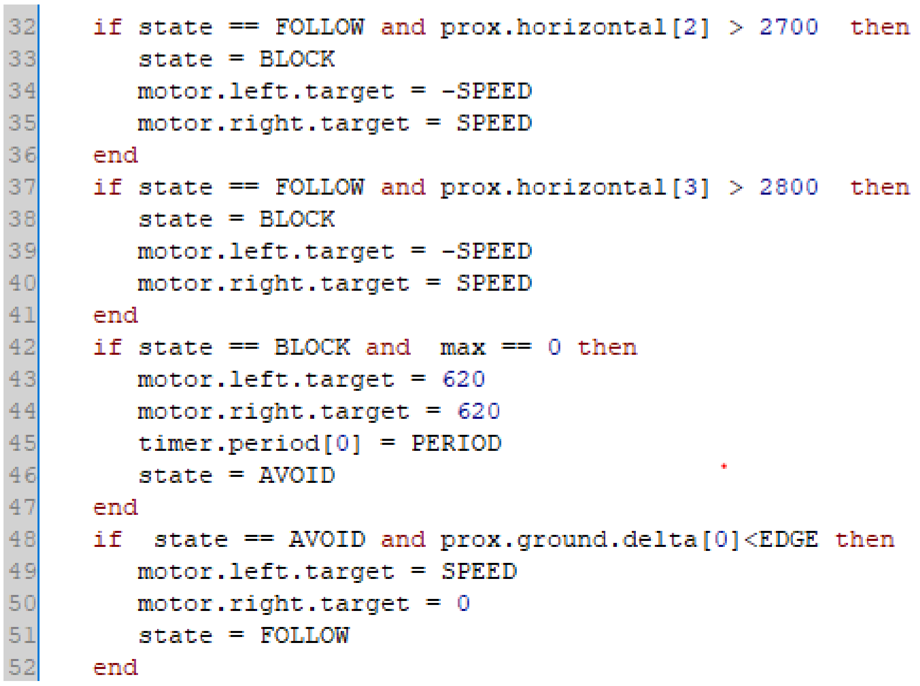


Constants

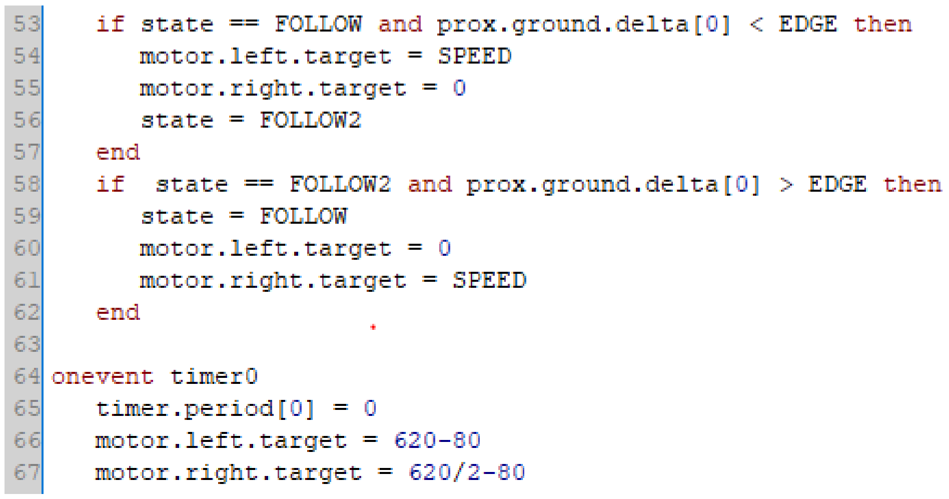
BLOCK	0
SPEED	800
STOP	1
FOLLOW	2
EDGE	300
FOLLOW2	3
AVOID	4
PERIOD	250



```
var min
var max
var mean
var state
motor.left.target = 0
motor.right.target = 0
state = STOP
timer.period[0] = 0


onevent button.forward
motor.left.target = SPEED
motor.right.target = SPEED
state = FOLLOW

onevent button.backward
motor.left.target = 0
motor.right.target = 0
state = STOP

onevent prox
   if  state == STOP then
      return

   end
   call math.stat(prox.horizontal[0:4], min, max, mean)
   if state == FOLLOW and prox.horizontal[0] > 2700  then
      state = BLOCK
      motor.left.target = -SPEED
      motor.right.target = SPEED
   end
```



```
   if state == FOLLOW and prox.horizontal[2] > 2700  then
      state = BLOCK
      motor.left.target = -SPEED
      motor.right.target = SPEED
   end
   if state == FOLLOW and prox.horizontal[3] > 2800  then
      state = BLOCK
      motor.left.target = -SPEED
      motor.right.target = SPEED
   end
   if state == BLOCK and  max == 0 then
      motor.left.target = 620
      motor.right.target = 620
      timer.period[0]  = PERIOD
      state = AVOID
   end
   if  state == AVOID and prox.ground.delta[0]<EDGE then
      motor.left.target = SPEED
      motor.right.target = 0
      state = FOLLOW
   end
```



```
   if state == FOLLOW and prox.ground.delta[0] < EDGE then
      motor.left.target = SPEED
      motor.right.target = 0
      state = FOLLOW2
   end
   if  state == FOLLOW2 and prox.ground.delta[0] > EDGE then
      state = FOLLOW
      motor.left.target = 0
      motor.right.target = SPEED
   end

onevent timer0
   timer.period[0] = 0
   motor.left.target = 620-80
   motor.right.target = 620/2-80
```

<u>References</u>:-

1. wiki.thymio.org. (2020). The Aseba language - Thymio & Aseba. [online] Available at: http://wiki.thymio.org/en:asebalanguage [Accessed 6 Mar. 2020].

2. draw.io. (2020). Flowchart Maker & Online Diagram Software. [online] Available at: https://www.draw.io/ [Accessed 6 Mar. 2020].
3. google.com. (2020). [online] Available at: https://www.google.com/search?q=dalhousie+logo&sxsrf=ALeKk01uCabpp9Z5OtASijLI8SfqtrvFIA:1583505468395&source=lnms&tbm=isch&sa=X&ved=2ahUKEwiE9MXHiYboAhXxYt8KHezXDGgQ_AUoAXoECAwQAw&biw=1536&bih=754#imgrc=UXi0DjgRRaSZwM [Accessed 6 Mar. 2020].