Report 3

This week the task was to program our robot to identify the sites and make the corresponding movements. In order to do that, we had to change both the robot and our code.

Structural changes

Sites can be identified by triggering the light switch and computing the frequency with which the light flashes. To be able to recognise the change in the lights, we attached the two remaining light sensors to the robot. We placed them to the front of the robot at exactly the same height as the lights are above the ground, which is 15.4 centimetres. We decided to add both of the sensors to the front of the robot, thus even if the robot is not perfectly aligned to the middle in front of the switch, one of the sensors will be triggered regardlessly. The light sensors are 3.6 centimetres apart, hence they are at approximately dividing the width of the robot in a 2:1:2 ratio (see Figure 1 for a picture of the robot).

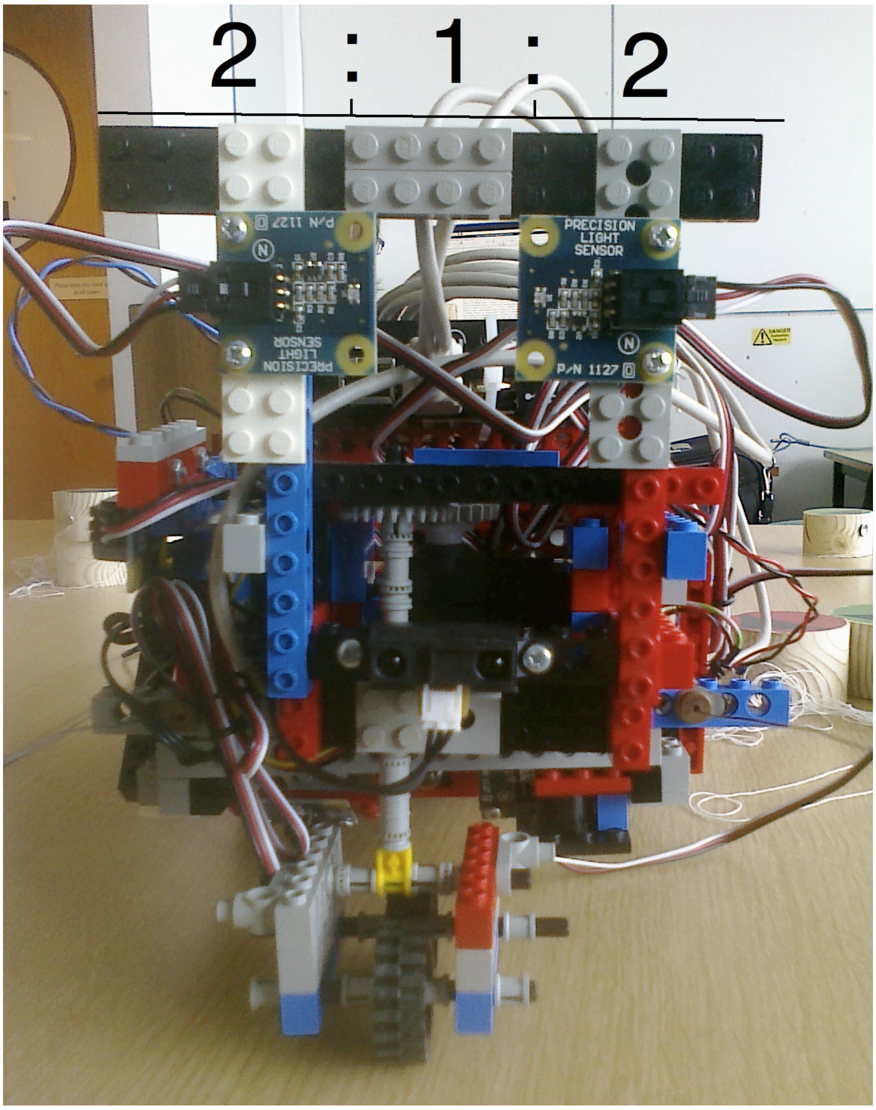

Figure 1: image of the front of the robot showing the ratio of the light sensors

Report 3

Software design

To prepare for this week's task, we first had to fix the remaining bugs in our code. Such a bug was the implementation of recognising the dark floor, as we found that our algorithm did not work equally well in different lighting conditions. To avoid this, an adaptive threshold was put in place by averaging the initial input from the bottom light sensors. The program recognises dark floor when values, that are at least 12% greater than the average are sensed, which we have found sufficient through testing. Furthermore, the threshold for the front IR sensor was raised to make sure that it does not prohibit moving on to a site when a wall triggers the sensor, however, the black floor has not been perceived. To make testing easier, the program now also contains several debugging options to minimise the amount of text printed out while running. The options are: behaviour, movement, sensor, and setup with each of them focusing on different parts of the program.

To implement the required task, we had to overcome the issue of inconsistency among the elapsed time between new signals. This is due to the fact that the sensor only sends a new signal if the sensed value changes. This way it is hard to measure the frequency of the light, which is given by the number of flashes in one second. To solve this issue, we set a threshold to the top light sensors and count the number of times our main function loop is executed while the sensory input is above the threshold. The main control loop is called every 50 milliseconds and we calculate the frequency as dividing 20 by the number of times the main control loop was executed. The first frequency that the robot acquires has to be excluded as it presents the time elapsed from the start to the first flash.

After the robot has detected the frequency of the light and identified the site, the function 'dance' is called which does the correct moves given the frequency. To accommodate for minor errors, we use upper and lower bounds on the frequencies in the dance function. For example, if f is the sensed frequency, we consider f to be 0.5Hz if $0.25\text{Hz} < f < 0.75\text{Hz}$, or f is 6Hz if $5\text{Hz} < f < 7\text{Hz}$. This way we can make sure that even if there is a small miscalculation in the frequencies, the correct movement will be made.

Evaluation section

The robot occasionally measured light frequencies correctly, however, for the higher frequencies the reliability rapidly diminished. However, when the frequency was detected correctly, the robot performs the expected movements. There is still need to do more testing and improve the method for measuring frequencies. Partially the problem may lay in the fact that the robot does not fully trigger the switch of the light source which is a problem we would like to solve by making structural changes next week.

Code

The full source code is available online at: https://github.com/gampleman/Robot-IAR.