**RaLaZaBa ELECTRONICS**

**Weekly Report**

**Team Members:** Ali AYDIN, Anıl AYDIN, Enes AYAZ, Nail TOSUN, Selman DİNÇ

**Advisor:** Lale ALATAN

|  |
| --- |
| **Done**   * We tried to control the mouse on MATLAB and we drew the path of mouse without using any rotation angle. We just drew on xy plane. The result can be seen in Figure. * We tried to obtain rotation angle by using gyroscope, but gyroscope only gives data when it moves against to ground. Due to that, we decided to use compass or two mousses to sense rotation of robot. * We created a simulation environment for shape and center detection. The maps and shapes are in the section of “Shape and center detection simulation” * After that, we tried to use measured data for shape detection, but our measured data needs to be cleaned in some aspects, thus our algorithm is not working well yet for real data. We will try to improve our algorithm erroneous data. |
| **To Do**   * We will start to write conceptual design report on Wednesday. * We will try to obtain rotation angle by using two mouse or a mouse and compass. |
|  |

**Shape and center detection simulation**

To test our first algorithm for shape and their center detection, we created simple maps with no error using paint. The maps that we tested are shown in from Figure 1 to Figure 4.

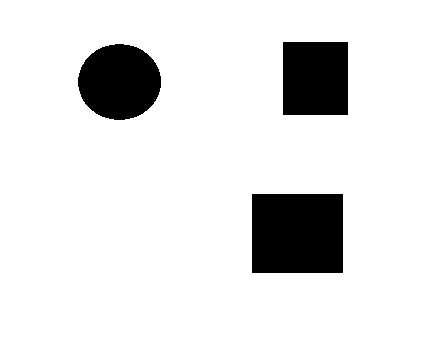
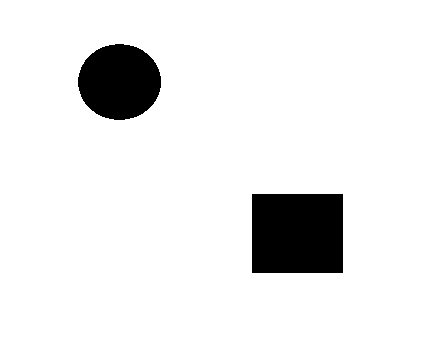
 

Figure : Test Map1 Figure : Test Map 2

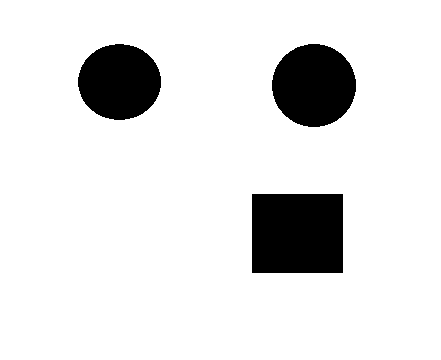
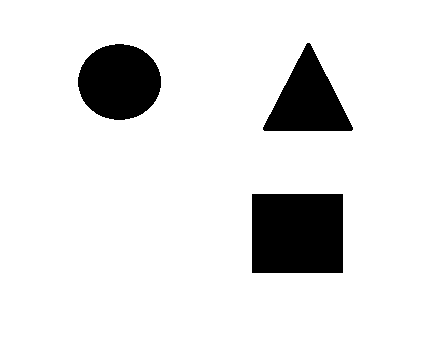
 

Figure : Test Map3 Figure :Test Map4

We created a simulation environment such that, we loaded these maps to a MATLAB script and we selected 3 points for sensing these shapes. From these 3 points, we created lines with different slopes and the intersection points represented the lidar scans. Our simulation environment is seen like in Figure 5. Then, we obtained these points and plotted. The intersection points for 4 different map configurations are shown in from Figure 6 to Figure 9.

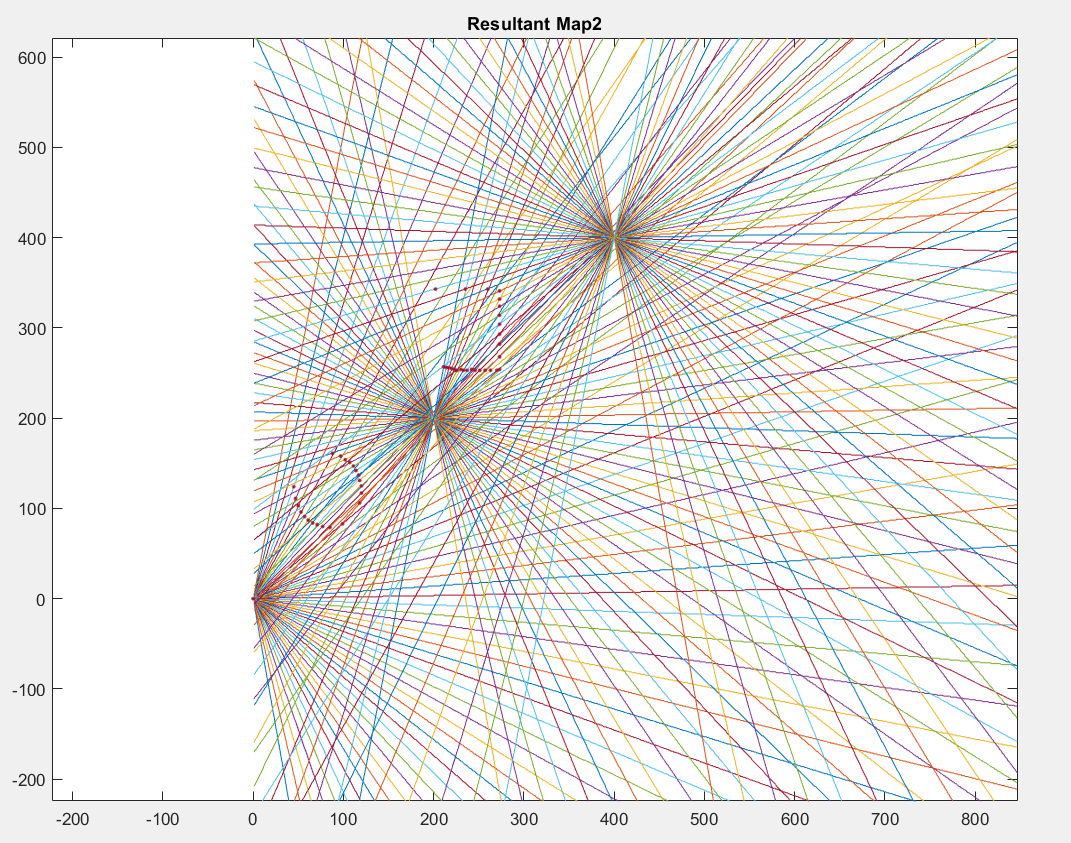


Figure : The simulation environment

After intersecting lines and our objects, some parts of objects are not shown which will be the case in real data. The missing parts of the objects can be seen in resultant maps. We must come up with this problem first. To do that, we combined x and y coordinates of resultant map data and converted these data to an image to be able to process and create objects.

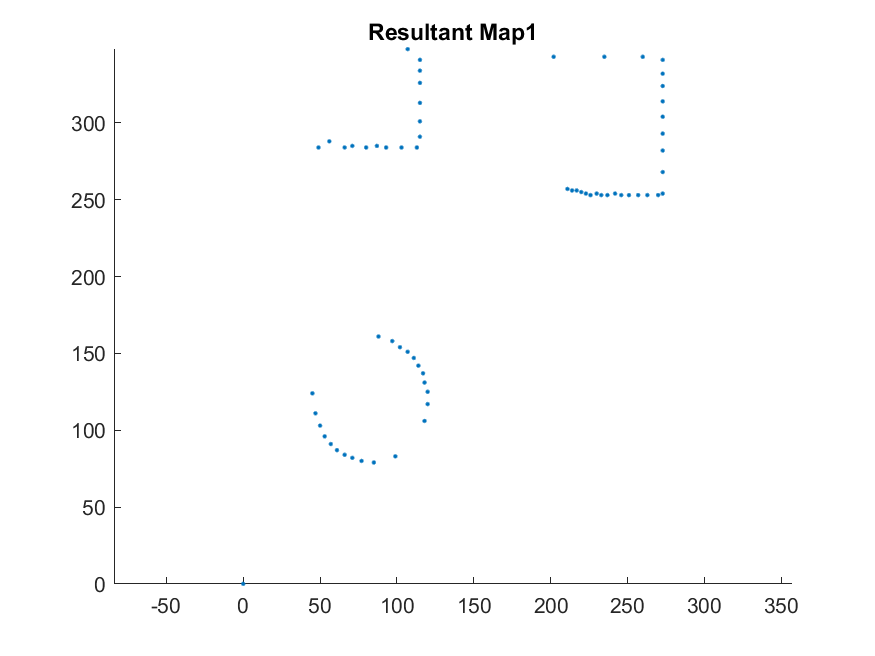
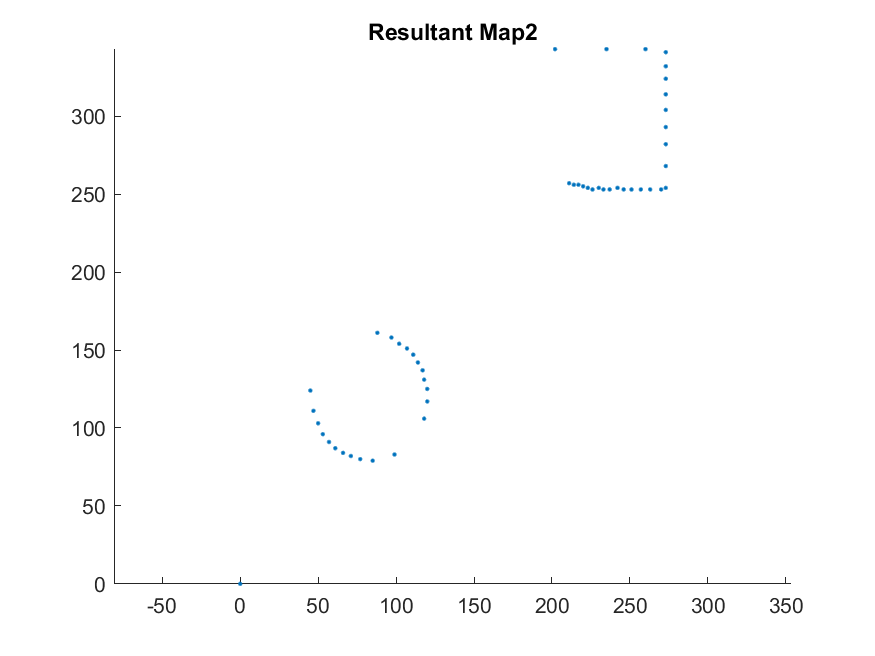
 

Figure : Lidar Scan Result for Map1 Figure : Lidar scan result for Map2

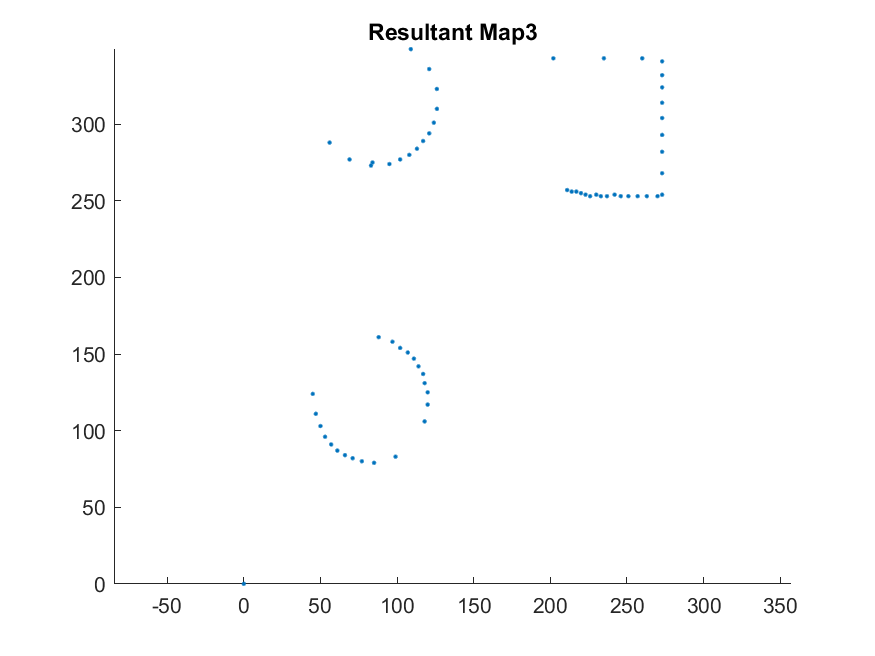
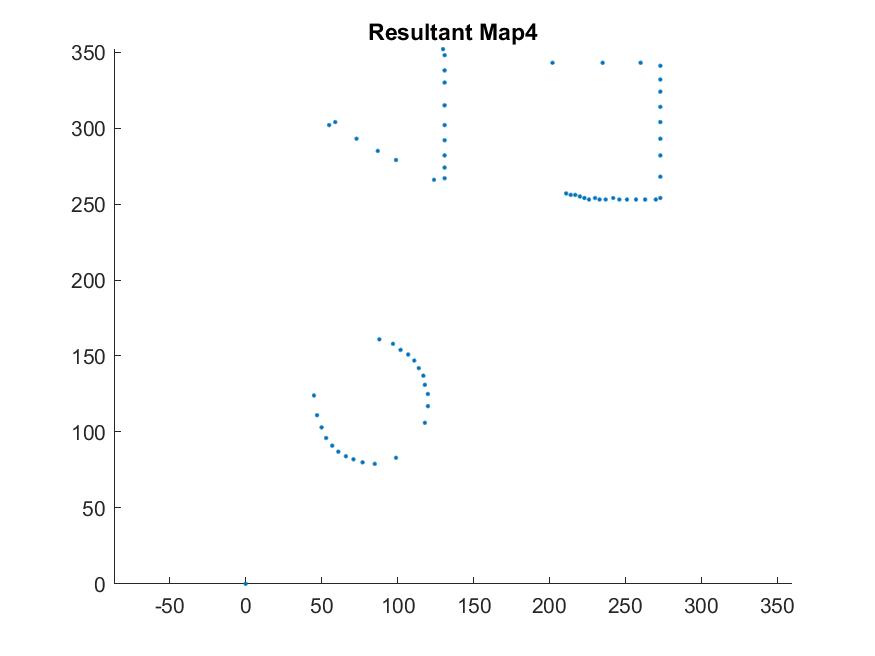
 

Figure : Lidar Scan result for Map3 Figure : Lidar Scan Result for Map4

Thus, after simulation, we converted resultant maps to images and made threshold operations. After thresholding, we found contours in the image and using some constraints such as the area of contours and the length of contour arrays, we created 3 different cases for rectangular, circular and triangular objects. The resultant maps from the missing data are shown in from Figure 10 to Figure 13.

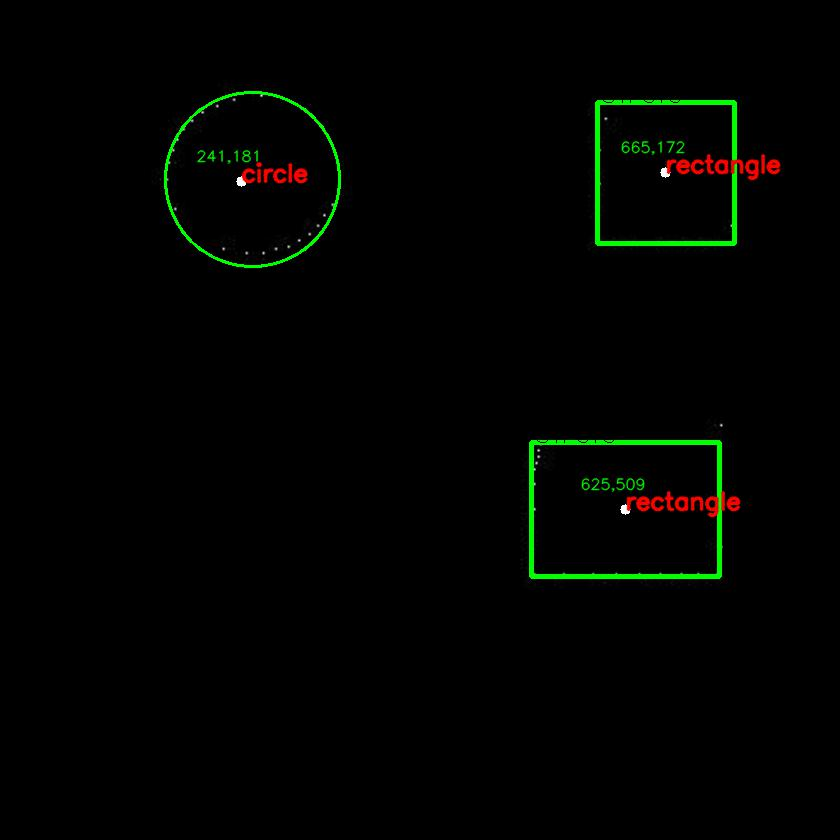
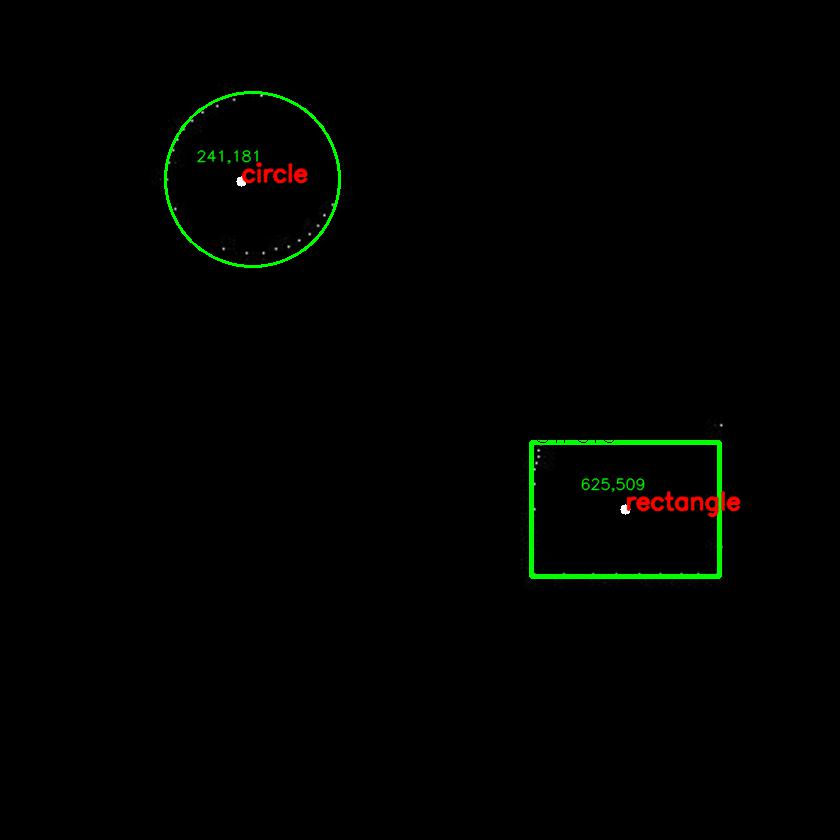
 

Figure :Resultant shapes and centers for Map1 Figure : Resultant shapes and centers for Map2

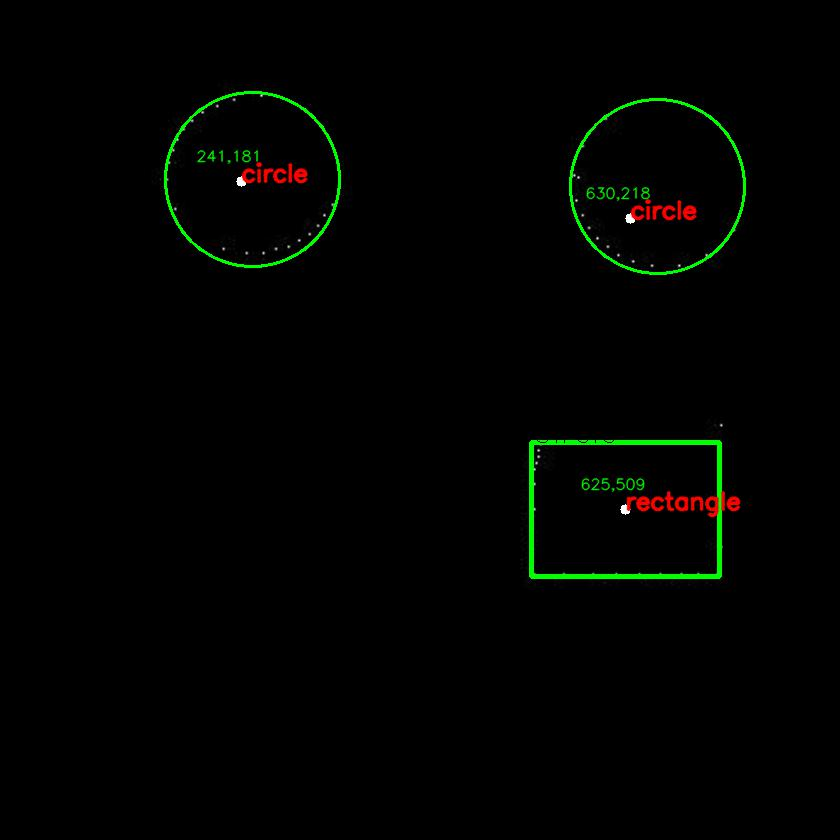
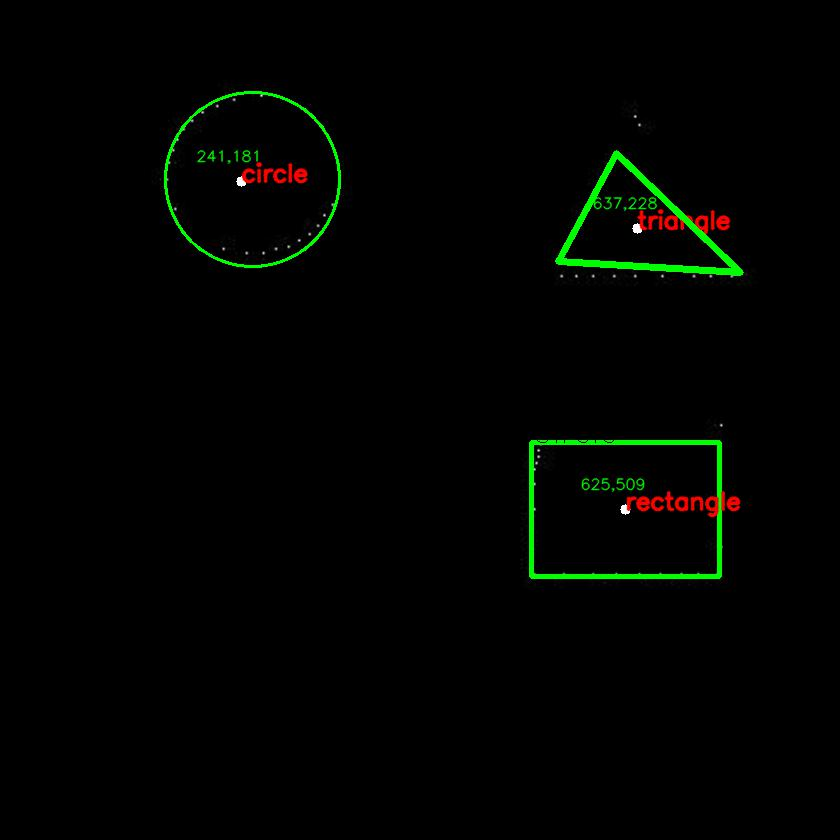
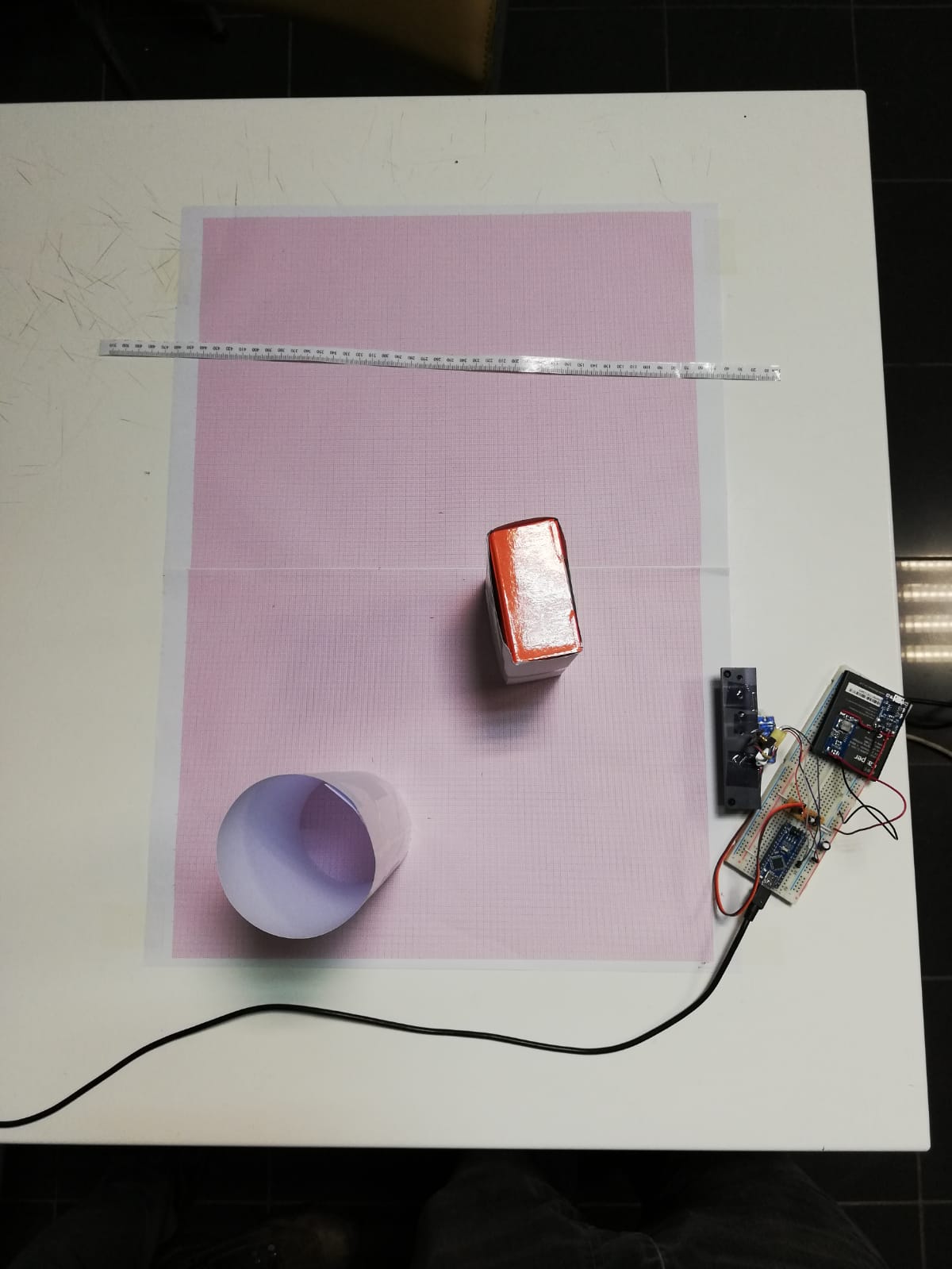
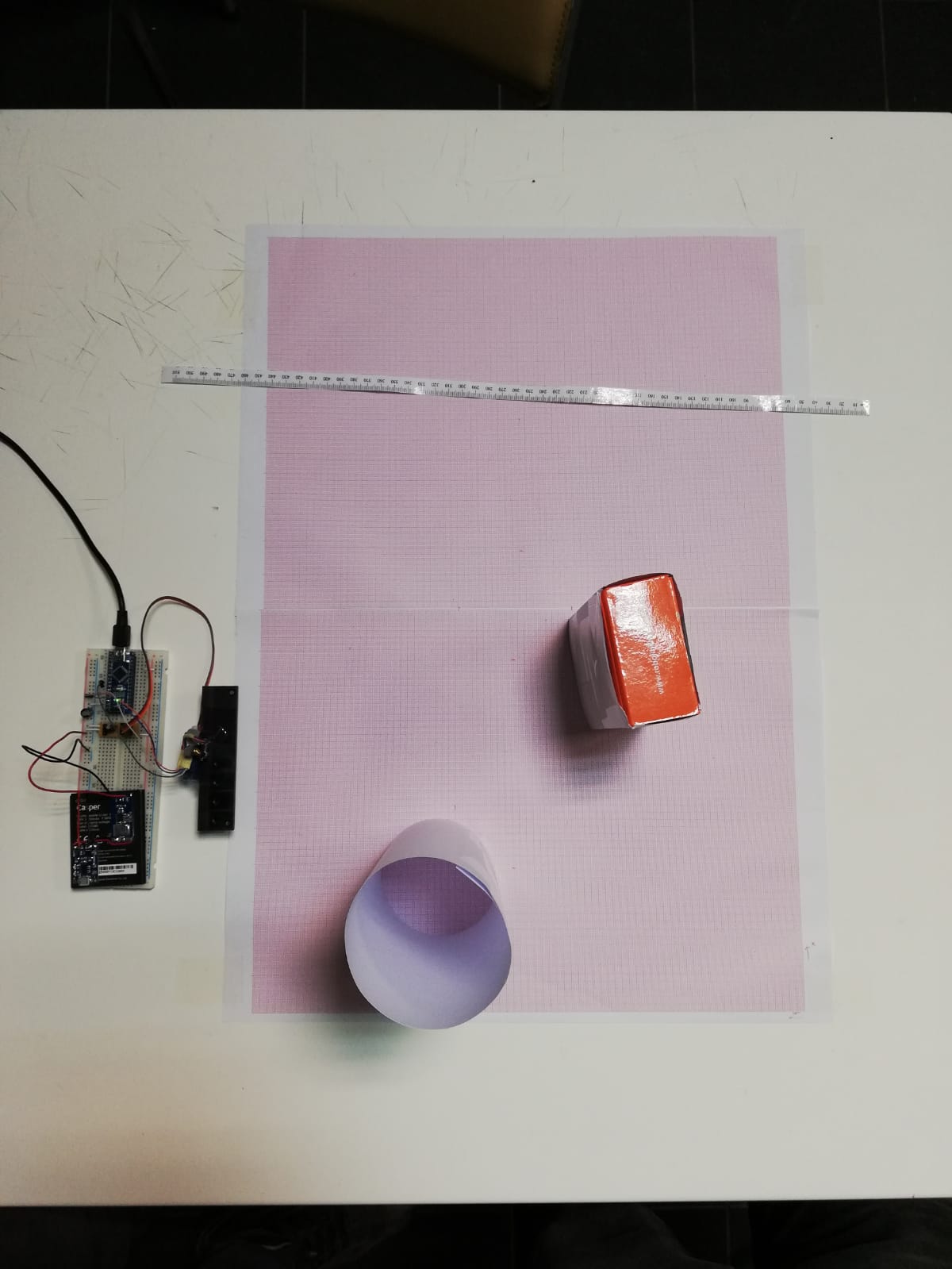
 

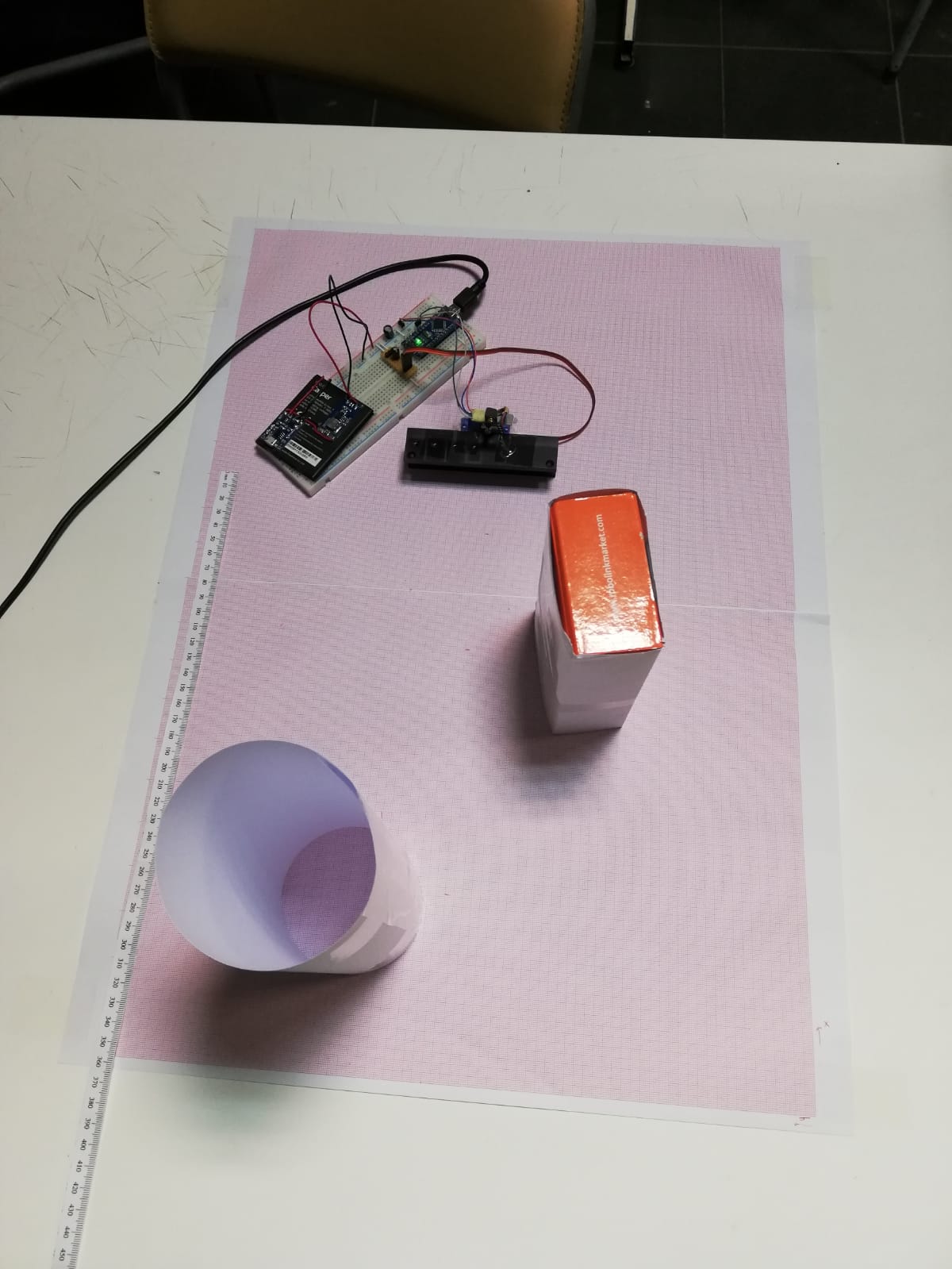
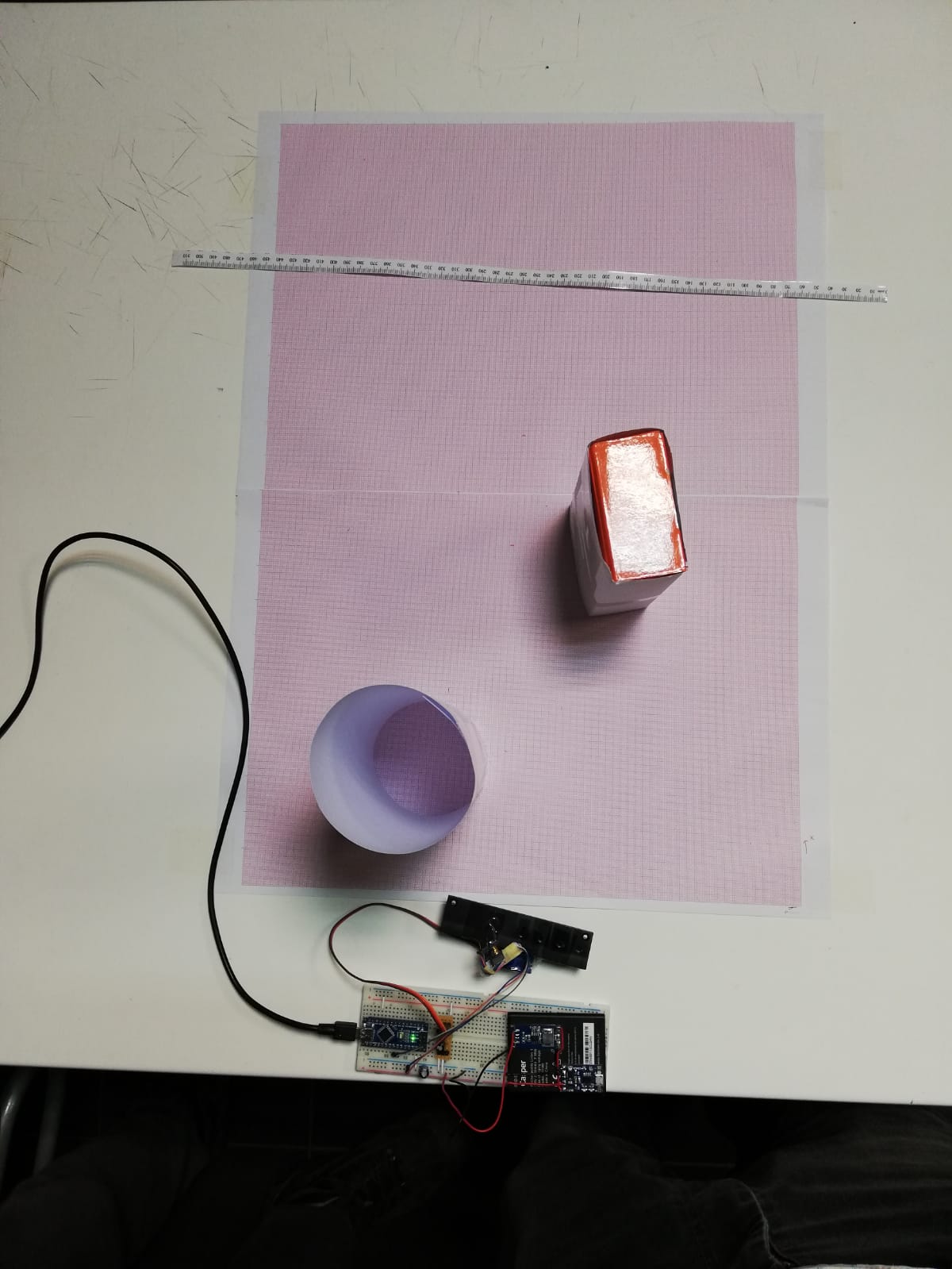
Figure :Resultant shapes and centers for Map3 Figure :Resultant shapes and centers for Map4

Yet, we have some problems about our algorithm which is detection of the triangular object is not working well. We will solve this problem by defining the triangle dimensions to the software as we know them.

**Measurement Data**

We set the measurement environment to test our mapping algorithm with real data. The setup, Figure 14-17, has 4 different viewpoints with respect to object. These data are merged by using an algorithm that takes a position of sensor( 2 Dimension) and angle between stationary coordinate system and sensor data. The output of algorithm is illustrated Figure 18.

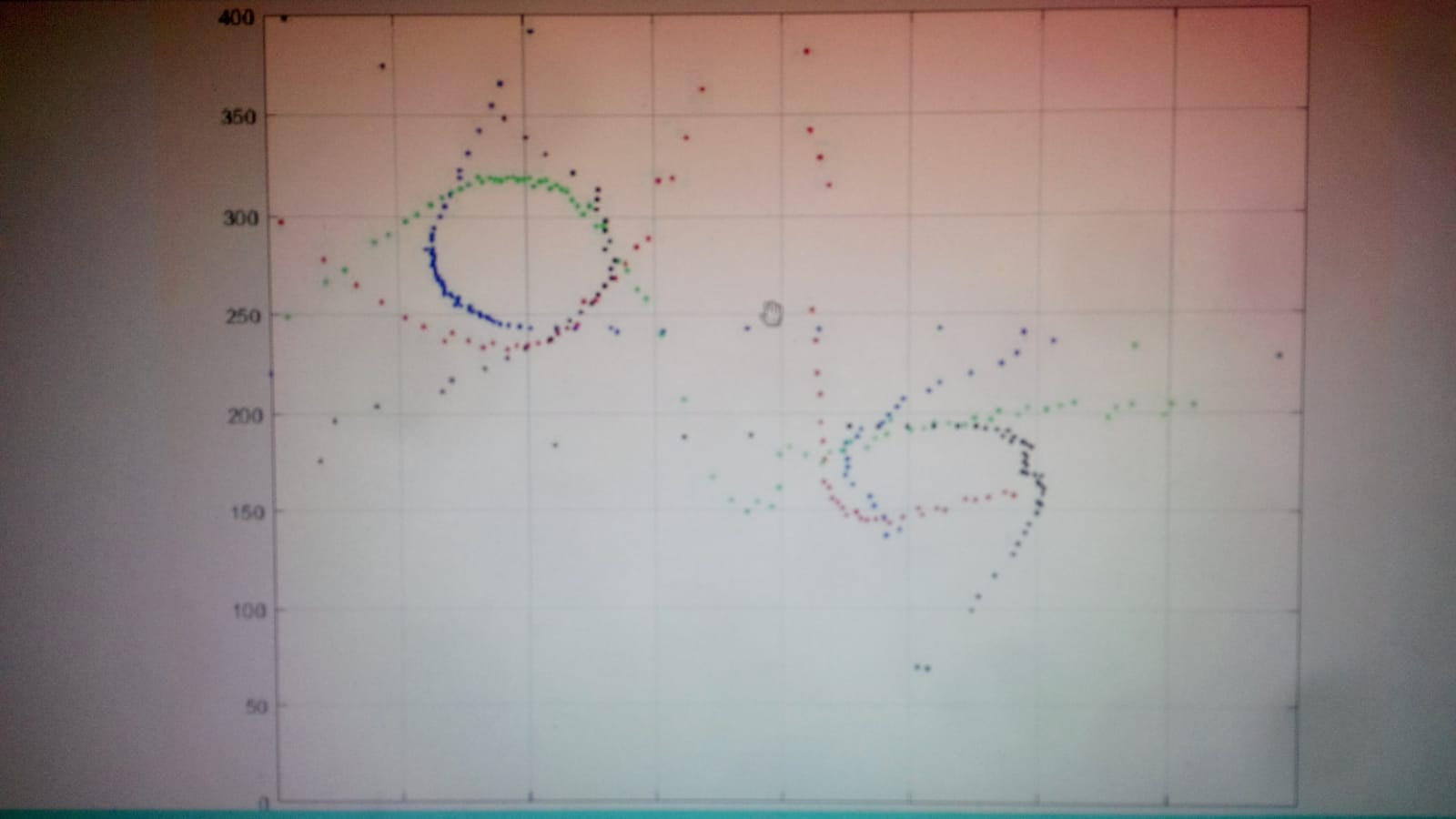


Figure : Test Result

**For Mouse**

We connected mouse to arduino and to obtain a graph, we take data on serial port by using matlab. Therefore, as seen figure xxxxx, we track mouse on the graph. To obtain tracking line, we drew vector between two point that sensed from mouse.

