TEL 412

Final Project 2016-2017

PC – Camera Car localization

Team : Kyriakos Psarakis

Eleftherios Tzagkarakis

Marios Vestakis

Konstantinos Skivalakis

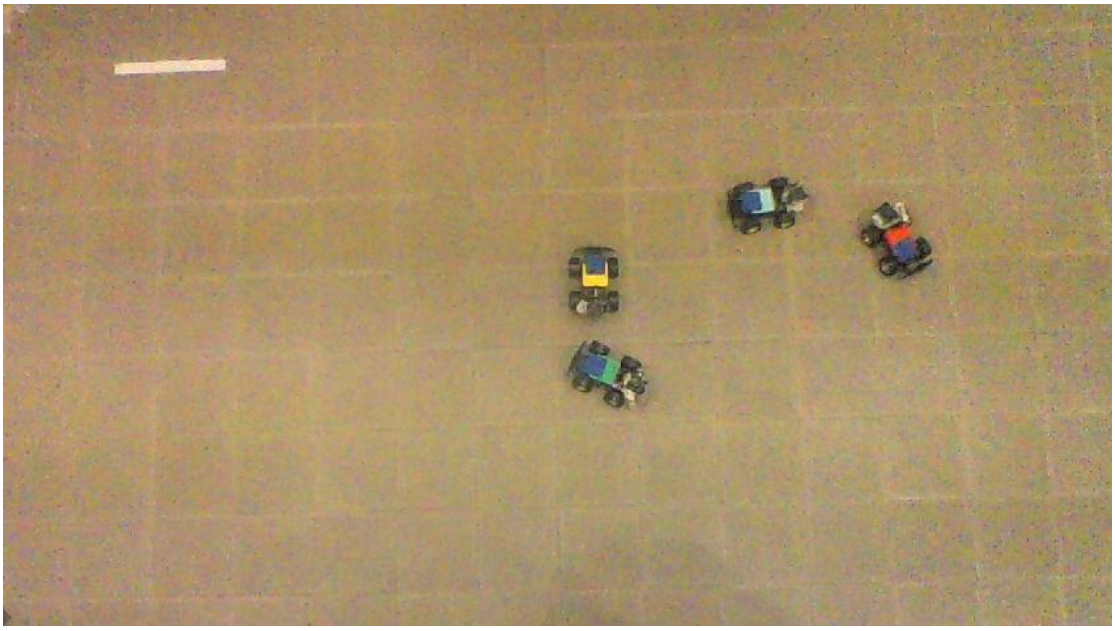
Introduction:

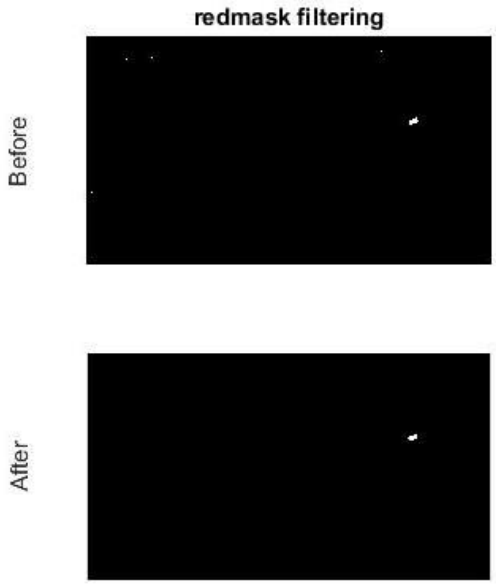
This project was part of the undergraduate course TEL 412 that took place in the Technical University of Crete School of Electronics and Computer Engineering. The goal of the project was to coordinate the movement of 2-4 cars in the 2-D plane in order to create specific shapes. In order to achieve that goal we used a camera for the car localization, the Silicon Laboratories microcontroller C8051F320 with the embedded radio Texas Instruments CC2500 (nodes) for matlab to master node serial communication and master node to the car-slave nodes communication.

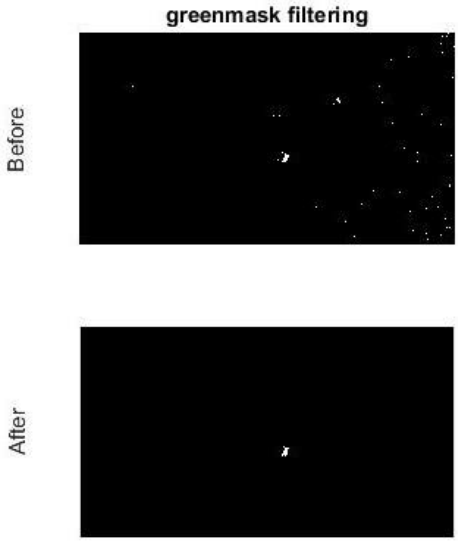
Part I : Color Segmentation

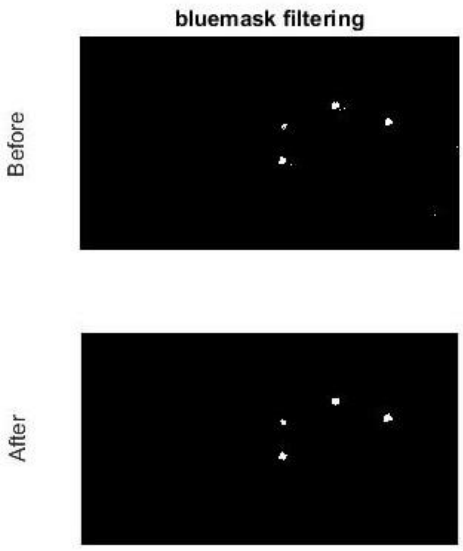
The first part of the project is to segment the different color stickers in an image in order to define the cars position and the direction it is facing (red, green, yellow, cyan) stickers for each cars ID and blue stickers to define the direction they are facing. We stay in the RGB color space because we didn't want to add any more complexity to our algorithm and in the testing we had good results in less than a second of execution time. We used thresholding and morphological filters in order to isolate the stickers and get rid of the noise. The following figures show the segmentation procedure before and after filtering with morphological filters.

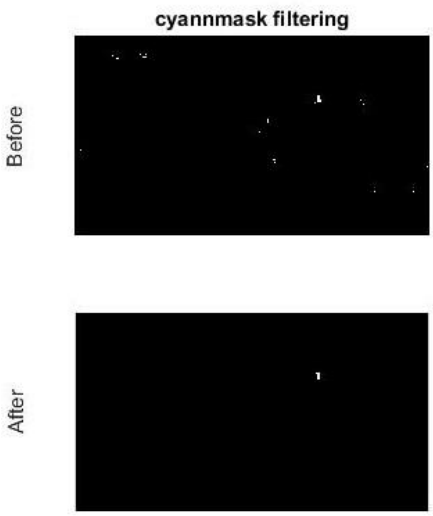
An example image that we use in this color segmentation result

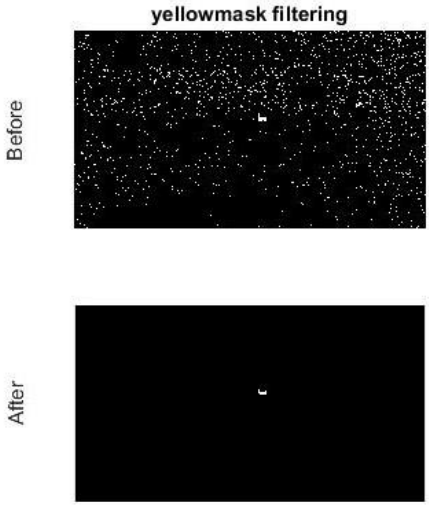







bluemask filtering
Before
After


cyannmask filtering
Before
After


yellowmask filtering
Before
After

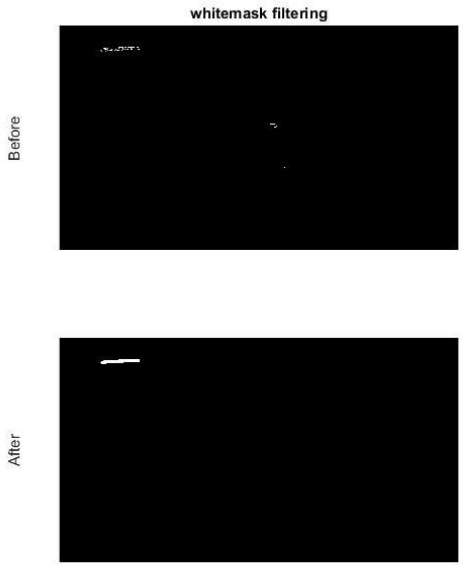
Part II: Calibration

In part II we want to relate the pixel distance in our image with “real world distance” (cm). So, we lay a white ruler vertically or horizontally (the ruler was 30cm). We can find out how many cm is one pixel with the equation:

[how many cm is one pixel] = 30 / [pixel length of the ruler]

This is scalable in any environment.

In the following figure we show the segmentation procedure of the white color in grayscale order to isolate the ruler with morphological filters. We know that the white color is very close to the 255 (max) in grayscale so we introduce a threshold in 220 and keep only the pixels above it. Then we close the image in order to connect the separated pixels of the ruler and after that we open the image so that we are rid of the remaining white color noise in the image



Part III: Target Definition

We ask from the user to give us the number of the cars and after they choose what shape they want to form. The possible options are the following:

- Two cars, with which they can form 1)a horizontal line, 2)a vertical line, 3) the main diagonal, or 4)the secondary diagonal of our picture space.
- Three cars, for a triangle.
- Four cars, which give the user the opportunity to select between 1) square, 2) diamond and 3) horizontal line.

We predefine all the points that define a shape to specific pixel coordinates.

Part IV: Distance and angle calculation

In order to find the angle, which our cars will rotate, we use vector theory from linear algebra. Firstly, we defined the direction vector of the car by joining up the ID-sticker center with its closest blue sticker center. Then we defined second direction vector which starts again from the ID-sticker center and ends in its specified target, which the routing algorithm provides us. We found the angle (1) and the distance (2) from the above mathematical formulas.

$$\theta = \arccos\left(\frac{\vec{v}\vec{q}}{\|\vec{v}\|\|\vec{q}\|}\right) \quad (1)$$

Euclidean distance:

$$\mathrm{d}(\mathrm{v},\mathrm{q}) = \sqrt{(\mathrm{v}_1 - \mathrm{q}_1)^2 + (\mathrm{v}_2 - \mathrm{q}_2)^2} \quad (2)$$

We also multiply the y axis by minus one in order to transform the image plane to Cartesian plane. Finally, we use the methodology of rotation and translation to find if the rotation is clockwise or counterclockwise.

Part V: Routing algorithm and collision prevention

I)Calculate all distances (between cars and targets).

II)Find the Max and note the target to whom this max belongs to.

III)Find the Min distance from a car to the target noted in II) and assign this car to the target.

IV)Start over from step I) until all cars are assigned to a target.

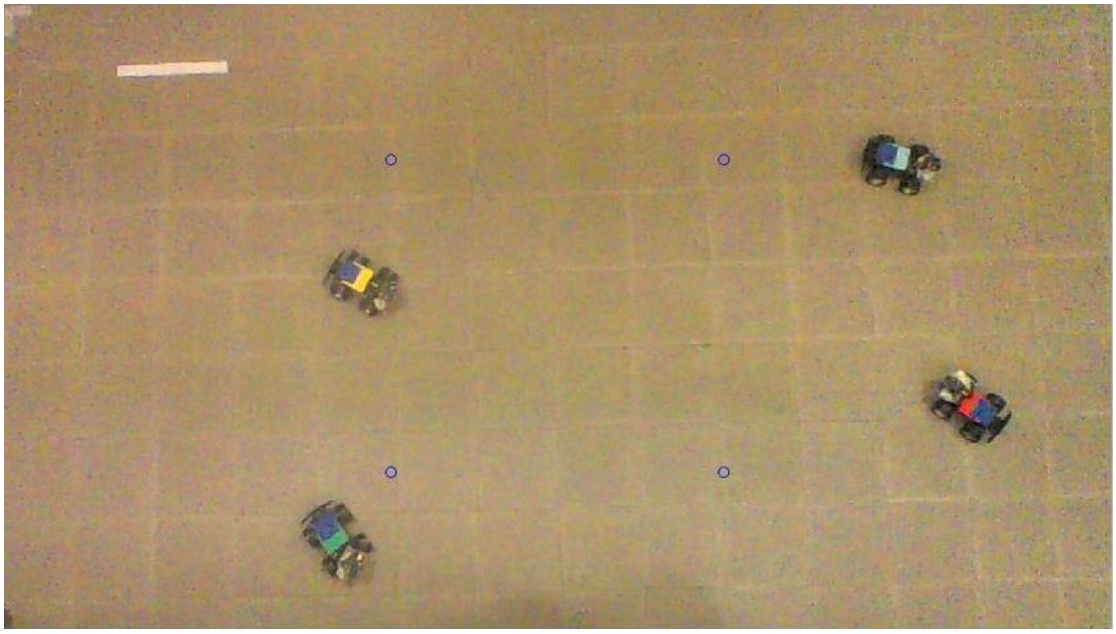
The idea is simple, measure all distances between cars and targets e.g., with 4 cars and 4 targets we have a total of 16 distances to measure (4 for each car between all 4 targets), what we do next is compare all the distances and keep the biggest. After having done that we note the target to whom this max distance belongs to and then we assign to this target the car that is closer to it. This process is iterative until we get 4 pairs (car, target).

Part IV: Matlab to Serial Communication

In this part we need to send the data to the cars , so we send a first package with the car id that the data is targeted to and after that the angle and distance follows . This procedure continues until we have sent the data to all the cars or if the car that the data was aimed at has reached its position.

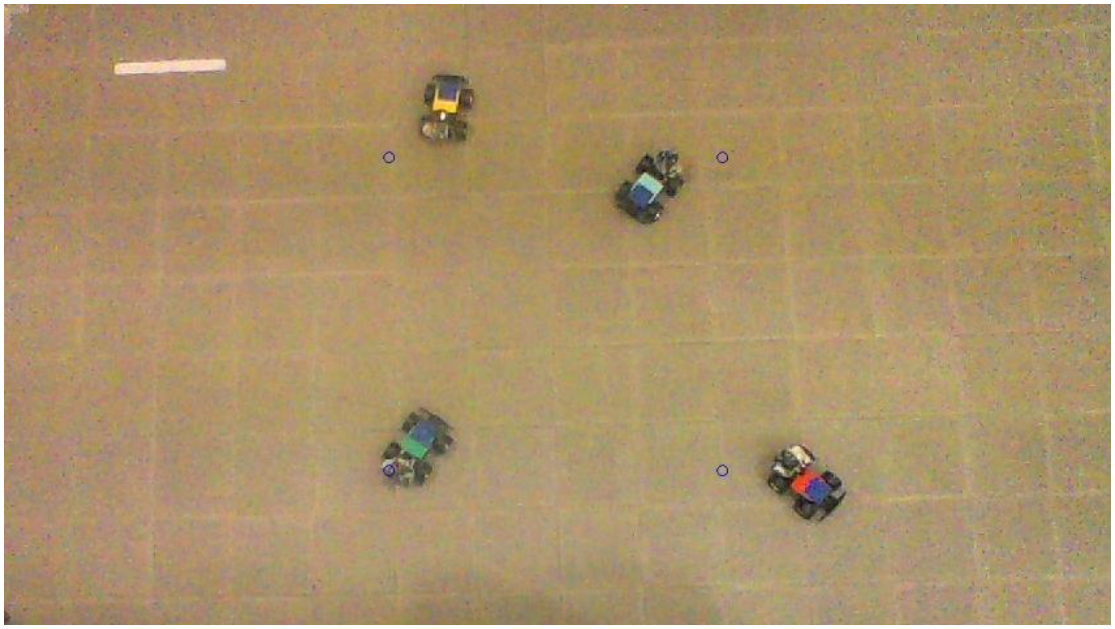
Execution:

Car position and targets at the start of the execution:



1rst Run of the algorithm:

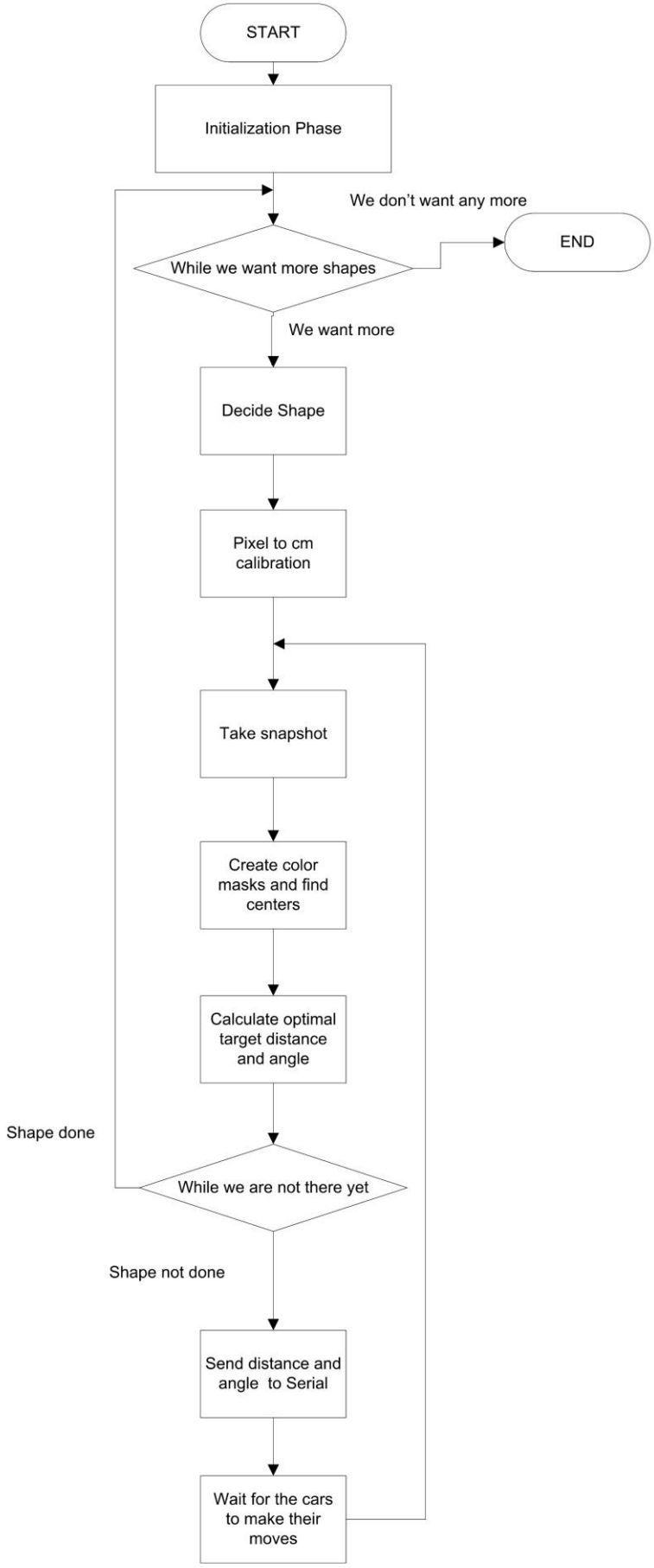
2nd Run of the algorithm:



3rd Run of the algorithm:

The cars have reached their targets within a range (accuracy of the positioning) so the algorithm stops.

Algorithm datapath:



Summary:

We conclude that for the cost of our implementation (cheap teleguided cars that was “hacked” in order to receive angle and distance instructions, a cheap webcam and simple to use color segmentation methods) we have acceptable results (we achieve the selected shape in 3 runs of our algorithm)

- Cost of our implementation
    - cheap teleguided cars that was “hacked” in order to receive angle and distance instructions
    - cheap webcam
    - simple to use color segmentation methods
- Good results
    - we achieve the selected shape in 3 to 5 runs of our algorithm
    - the whole algorithm runs in less than 2 seconds (except for the serial to master communication)