Aalborg University

Vision, Graphics and Interactive Systems

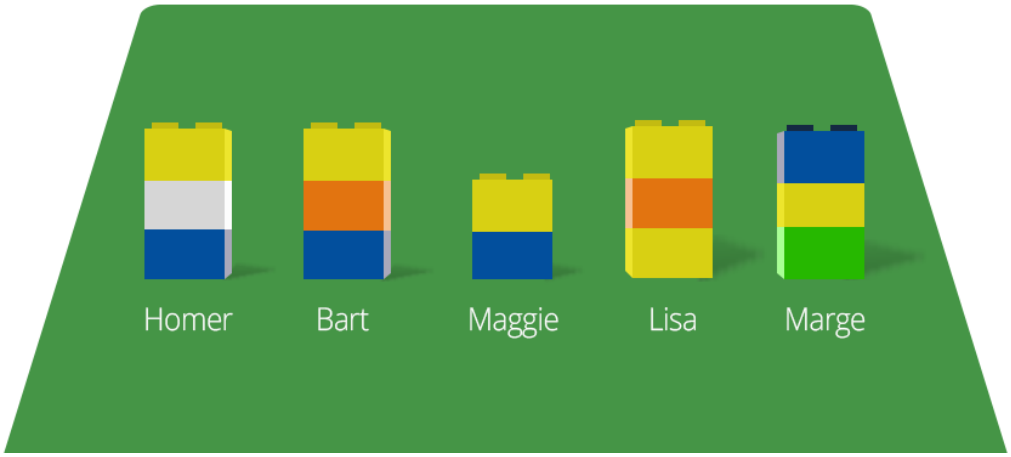
8TH Semester - Group 842

Robot Vision

Simpsons Figures with Lego Duplo Bricks

Students:
Andreea-Daniela Ene
Yanis Guichi
Daniel Michelsanti
Rares Stef

April 14, 2016



Contents

1 Introduction **3**
1.1 Problem 3
1.2 Equipment 3
1.3 Setup 4

2 Design and Implementation **6**
2.1 Acquisition (Image Capture) 7
2.2 Camera Calibration 7
2.3 Undistortion 8
2.4 Bricks Detection 8
2.5 Path Generation 9
2.6 Robot Movements 10

3 User Interface **13**

4 Test and Results **14**

5 Conclusion **15**

1 Introduction

This report has been created by Group 842, which consists of four students in the 8th semester of the Master Degree in *Vision, Graphics and Interactive Systems* from Aalborg University. This mini-project has been developed in the Spring Semester 2016 for the *Robot Vision* course. The aim of this report is to describe the steps that have to be taken in order to build Simpsons' figures with Lego bricks, using a KUKA robot.

1.1 Problem

The purpose of the mini-project is to build the main characters from the american sitcom called *The Simpsons* out of Duplo Lego Bricks, using a KUKA robot. The rules for building those figures are the following:

- Each character contains 3 bricks, except for Maggie, who is built only with 2.
- All bricks are randomly placed on a table next to the robot. The bricks should not overlap.

In order to complete this task, a system to build the Lego figures has to be developed. The main tasks that need to be achieved are:

- identifying the type, color, and location of the bricks on the table.
- determining the path that the robot has to follow.
- grasping the bricks.
- placing the bricks on another plane, in a predefined order.

1.2 Equipment

The materials and the tools that we can use are the following:

- 14 Duplo Lego bricks (left part of Figure 1), as follows:
  - 3 x 4 for Homer, Marge, Lisa, and Bart.
  - 2 x 1 for Maggie.

- Black A4 paper, for placing the bricks in order to build the figures.
- Calibration pattern attached to the black paper to calibrate the camera.
- Lego green board (254×254 mm), to place the final figures.
- Logitech HD Pro C920 webcam, that will be used to get the initial positions of the bricks.
- KUKA AGILUS KR 6 R700 SIXX Industrial Robot having a gripper, to move the Lego bricks. It has 6 degrees of freedom and 6 joints (right part of Figure 1).
- Computer with MATLAB, to connect the camera and to control the robot.

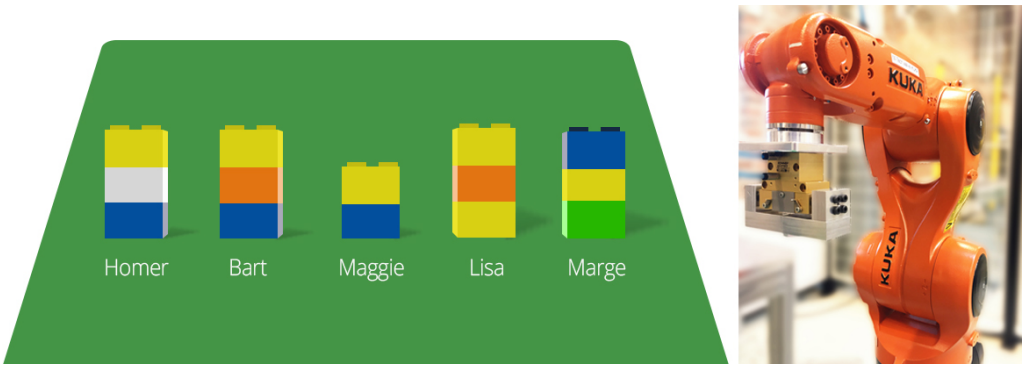


Figure 1: *Simpsons' figures and KUKA Agilus KR 6 R700 SIXX.*

1.3 Setup

In order to move the robot and find the positions of the bricks, a camera has been placed above the table, where the bricks and the checkerboard are. The bricks are placed on a black paper to make the detection easier and limit the light reflection. We used MATLAB on the computer to move the robot through the provided interface. The setup can be seen in Figure 2.

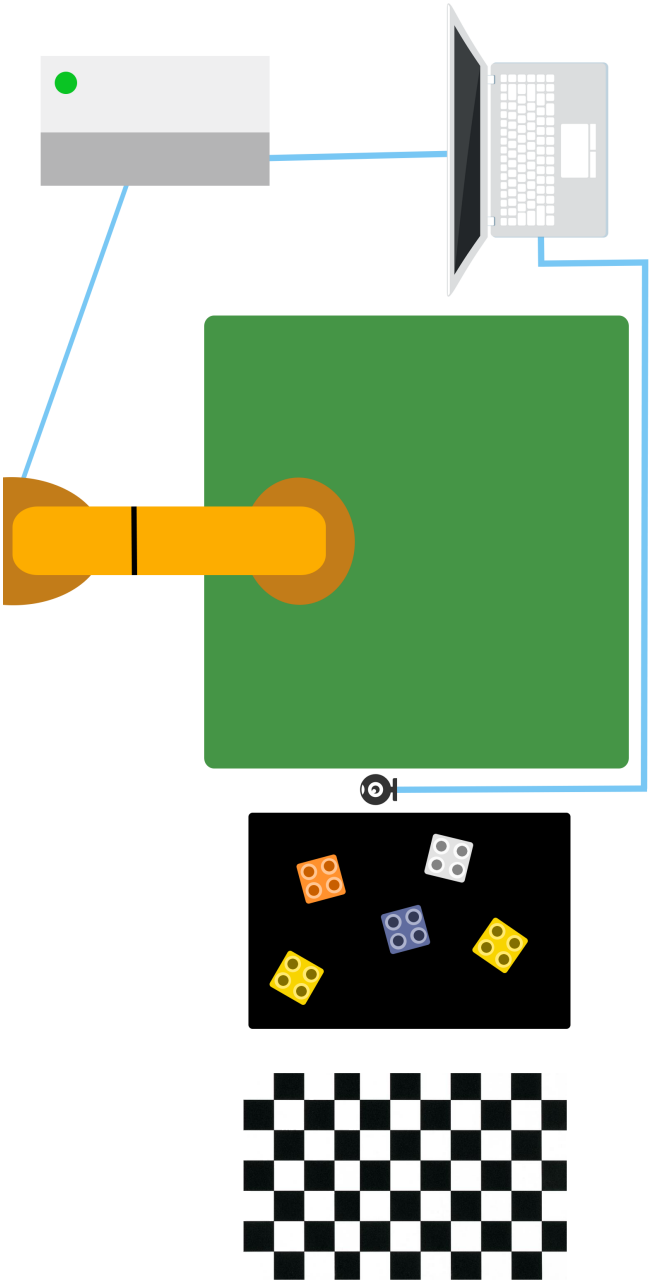


Figure 2: *Setup.*

2 Design and Implementation

The entire implementation has been divided into smaller steps. The flowchart in Figure 3 shows the stages needed to build the Simpsons' figures.

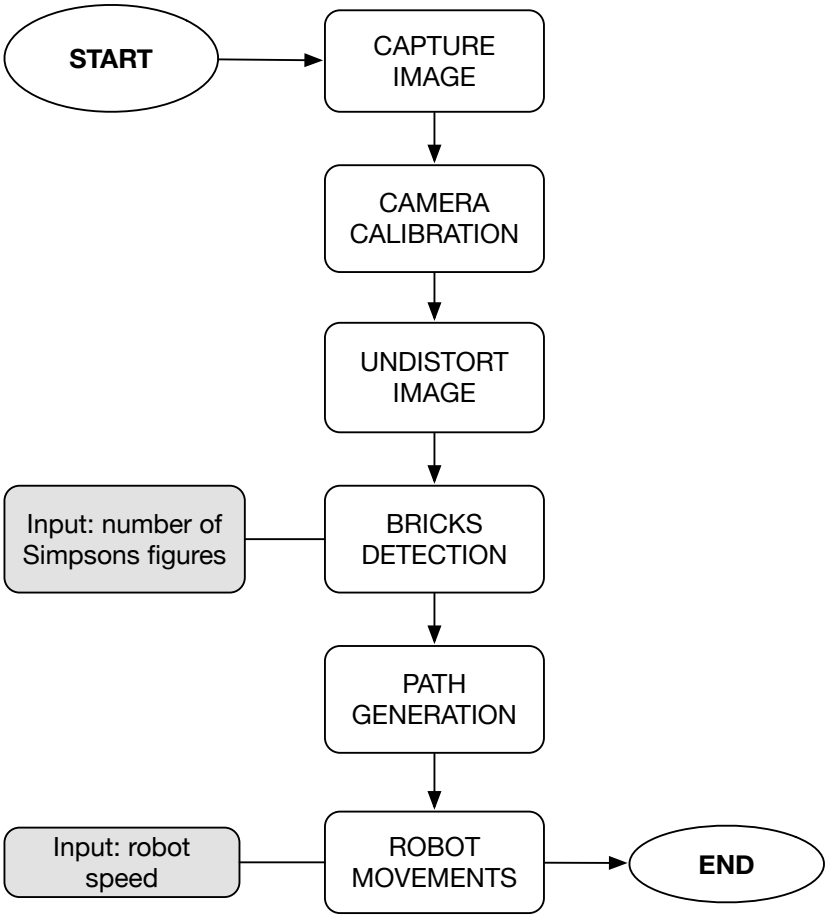


Figure 3: *Implementation Flowchart.*

After the image is captured by the webcam connected to the MATLAB interface, it has to be calibrated and undistorted in order to detect the correct positions of the bricks. Given the number of the Simpsons' figures that have to be built, the program should generate a path for the robot to follow. After setting the speed, the robot should start picking up the objects and place them on the Lego green board.

2.1 Acquisition (Image Capture)

After getting the stream of the camera and setting the trigger mode to manual, we capture 30 snapshots in order to avoid any focus issues. Detecting the checkerboard is needed to get the positions of the bricks in the world coordinates. If the checkerboard is detected, the image will be shown in the GUI (Figure 4), otherwise an error message will appear, suggesting to move the camera.

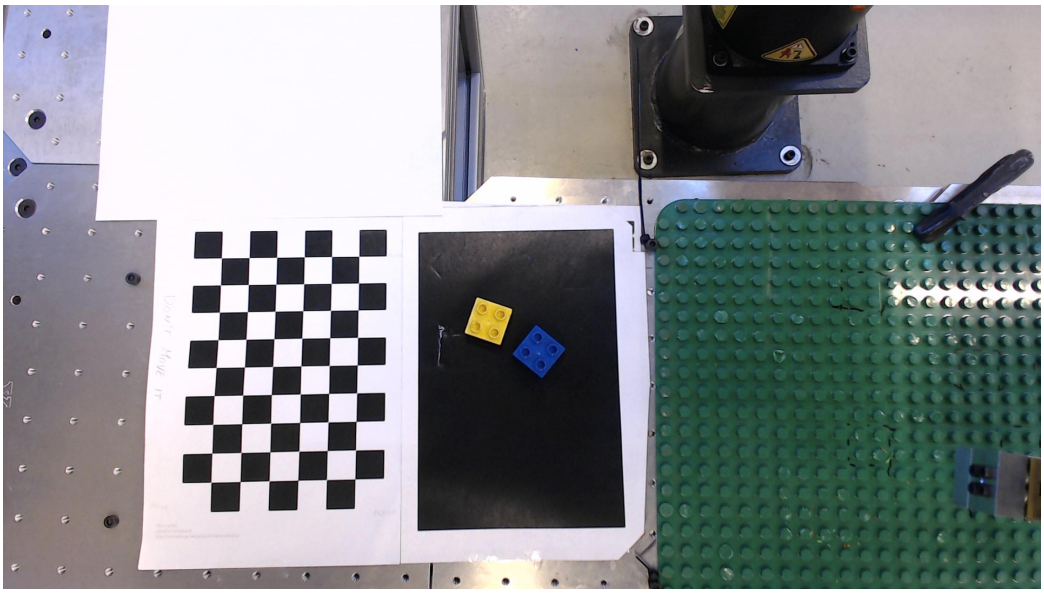


Figure 4: *Image acquisition.*

2.2 Camera Calibration

In order to perform the calibration, the *Computer Vision System Toolbox* is used. Having a set of 23 calibration images, we detect a calibration pattern and we generate the coordinates of the corners of the squares with respect to one of the corner that is the origin of the checkerboard frame, knowing that the size of a square is 23.5 $\times$ 23.5 mm. The calibration allows us to get the camera parameters (intrinsic and extrinsic) that will be used to undistort the image and calculate the world coordinates of the center of the bricks.

2.3 Undistortion

The distortion of the image caused by the camera can be reduced by using the intrinsic parameters (Figure 5).

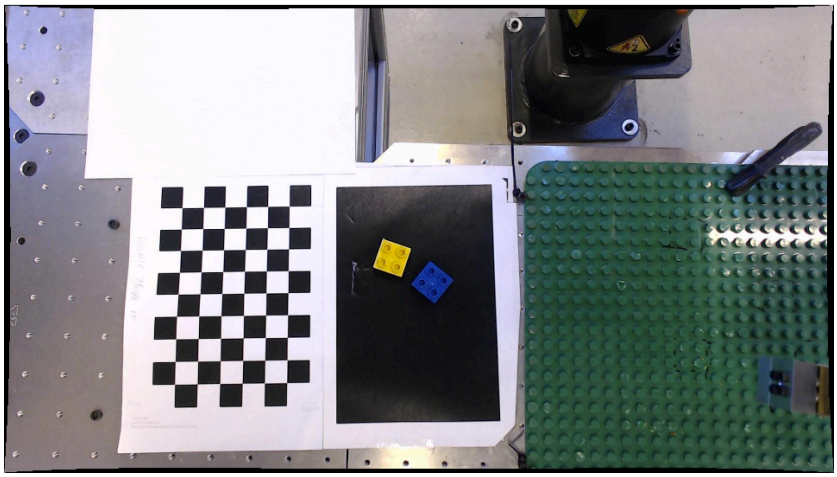


Figure 5: *Undistorted image.*

2.4 Bricks Detection

The image is converted into HSV and HSL, in order to avoid any influence of light and to detect the white bricks that are visible.

All channels have to be split separately by using multiple thresholds in order to detect the colours. We chose the thresholds according to the values from the image, which are directly recovered using the MATLAB built-in tool. In this way we obtain a binary image where the bricks and the areas that satisfy the conditions imposed by the thresholds are in white.

However, we want to detect only the bricks, so we have to use some features to exclude the other parts. We decided to use the values of the width, the height, and the ratio between them. For this, we use the *bwconncomp* function to get only the connected components of the image. Then, we use the *regionprops* to get the bounding boxes and keep the BLOBs that satisfy the previously mentioned conditions. The result is an image that only contains the detected bricks.

Once the bricks are detected, we used the *BoundingBox* and the *Centroids* fields of the *regionprops* to draw and get the coordinates of the bounding

boxes and of the centre of each brick. The previously described steps of the detection can be seen in Figure 6.

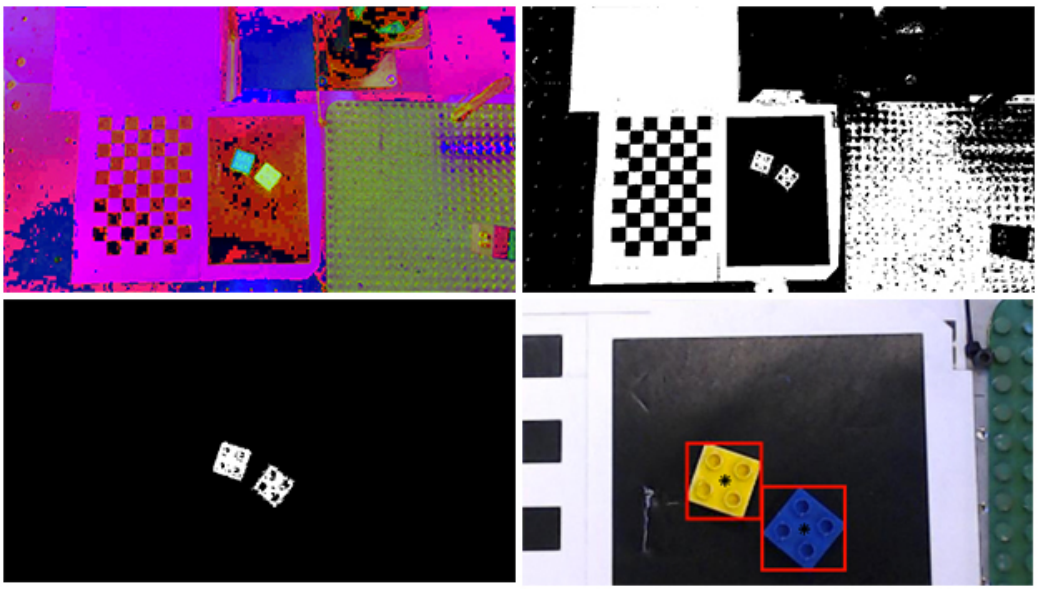


Figure 6: *Brick Detection Steps. Upper left: Image in HSV space. Upper right: Binary image. Lower left: Binary image only with bricks. Lower right: Bounding boxes and centroids.*

After computing the centres, the *Extrema* field is used to get the corners of the brick and to compute the orientation angle that rotates the robot tool as follows:

$$\alpha = rad2deg(\arctan(a/b)),$$

where α, a and b are the parameters shown in Figure 7.

Once the angle and the position of each brick have been computed, the path that the robot arm should follow can be determined.

2.5 Path Generation

Having the variable *col_needed* with the required number of bricks for each color, we sort the list according to each level of the figures. If the bricks are

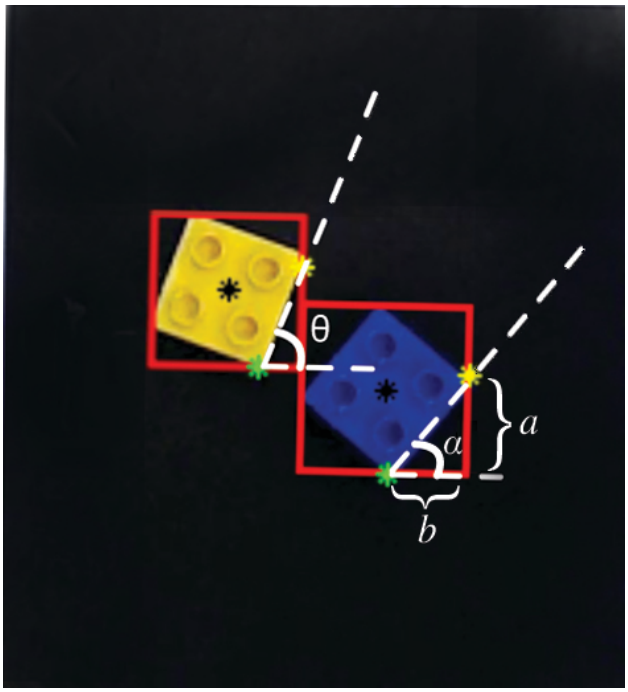


Figure 7: *Orientation determination.*

not enough for building the previously set figures, an error message will be displayed.

The matrix that contains the positions of the bricks is sorted in a way that the robot will start grasping the bricks by level, and not by stack. After having generated this matrix, the robot can sequentially access all the elements and build the figures.

2.6 Robot Movements

Three different frames were used: world frame, green board frame, and checkerboard frame. The green board frame has been placed so that all its axes are parallel to the world frame. The checkerboard frame is rotated by 180 degrees around the x axis with respect to the world frame. In this way we can calculate the coordinates with respect to a frame by using a simple transformation (Figure 8).

In order to move the robot, the following methods of the MATLAB interface have been used:

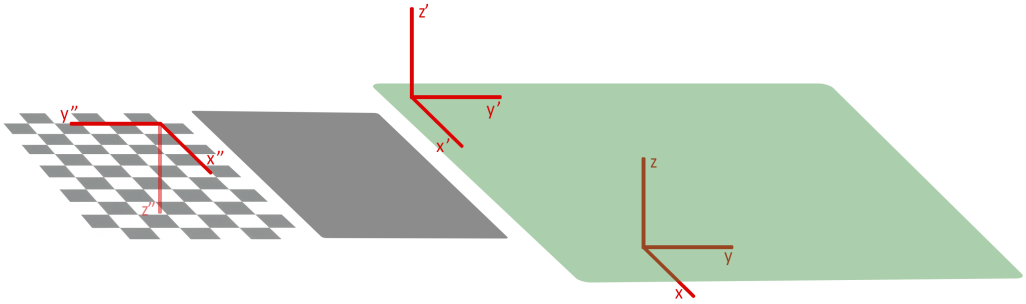


Figure 8: *The 3 Frames World Coordinates(x, y, z), Green Board Coordinates(x', y', z'), Checkerboard Coordinates(x", y", z").*

- *moveLinear* and *moveJoint*: moves the robot either linearly or by joints.
- *getPosition* and *getJoints*: returns either the robot positions or the angles of the joints.
- *openGrapper* and *closeGrapper*: opens and closes the gripper.

A home position of the robot has been chosen in order to avoid interfering with the image capture. Between picking up the bricks and placing them on the green field, an intermediary position called *switch point* has been set. The robot moves to that position each time it prepares to pick up a brick or place it on the green board.

When the robot attempts to grab a brick, a joint movement is performed to rotate the gripper to its orientation. The grab is performed with the robot moving down on the brick and once it is in position, the gripper will close. In the next step the brick is lifted up, the gripper is rotated to the previous orientation and the tool moves to the switch point. In order to avoid any impact with other bricks that are already placed and stacked, the robot performs an upper movement called *security height* in the switch position, then moves over the green board to place the brick. With the placement being completed, the robot moves back into the switch position and afterwards proceeds to repeat the steps above in gathering and placing another brick. If there are no more bricks to extract and place, then the robot proceeds back into the home position. The process is represented in Figure 9.

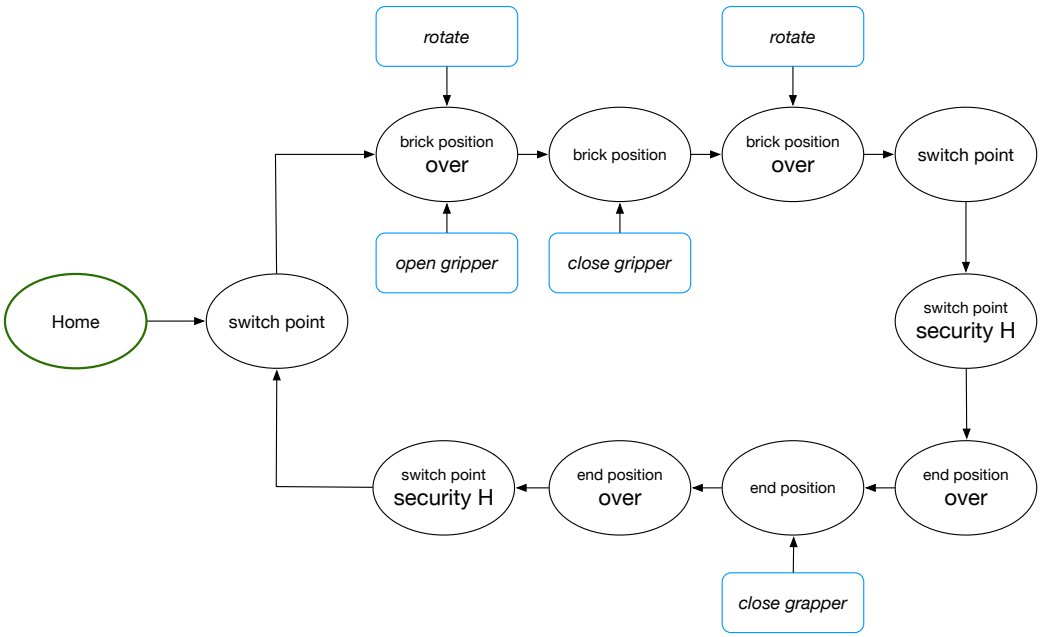


Figure 9: *Robot Movements Flow.*

3 User Interface

Based on the previously explained process, a Graphical User Interface (GUI) was implemented to allow the user to perform all the steps. The design of the interface can be seen in Figure 10. The menu is placed on the left side of the window and the center-right space is used for the processed images.

The user can interact with the application by using the menu on the left and select the Simpsons' figures to build. In addition, it is also possible to select the speed of the robot movements with the help of a speed slider. The Start button is used to upload the settings to the robot that starts the building process.

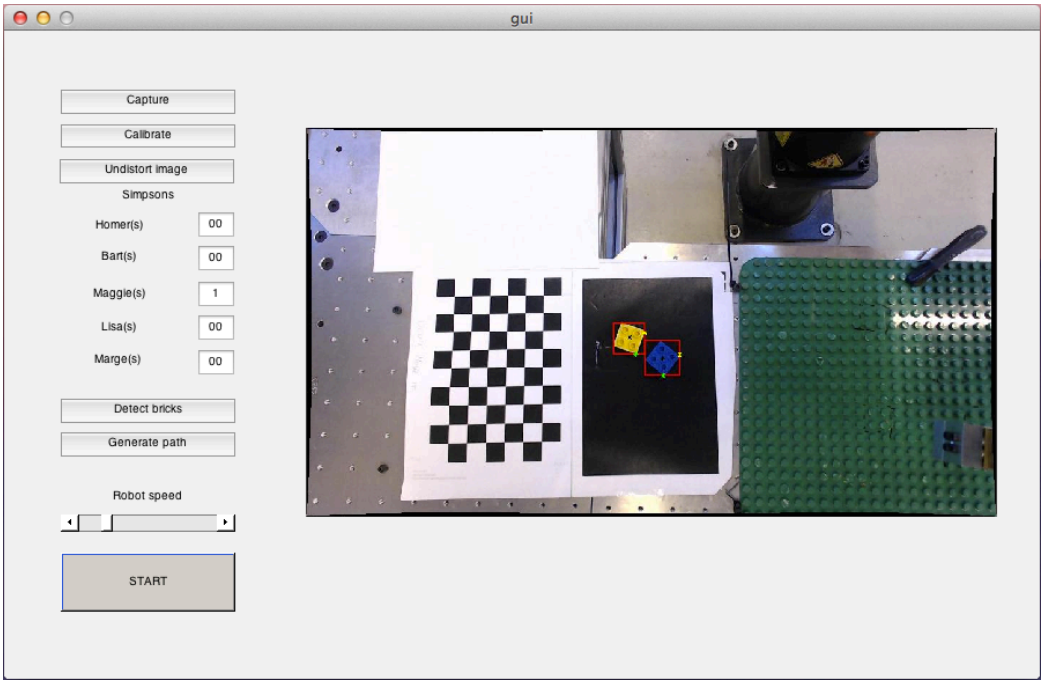


Figure 10: *Graphical User Interface.*

4 Test and Results

The purpose of the tests that have been performed is to move the robot to the right locations above the bricks and on the desired space on the green board. The height of the robot has been fixed, for both when it had to grab the bricks and to place them on the board, in order to avoid collision.

The program was firstly tested on bricks with one of the edge parallel to the x-axis of the world frame, in order to check that the brick detection and the computed coordinates were correct (Figure 11).

Some difficulties have been encountered with the gripper, mainly during the rotation, because when we change the orientation of the tool with the provided interface, also its position changes. We fixed this issue by using the *moveJoint* function to only move the joint relative to the gripper, avoiding any further computation of the trajectory and hence any undesired translation.

In order to test that this issue has been fixed, the bricks for building Maggie have been randomly placed on the board. After getting the positions of the bricks and calculating the angle, we used the joint movements for rotating the gripper, in order to be able to pick them up.

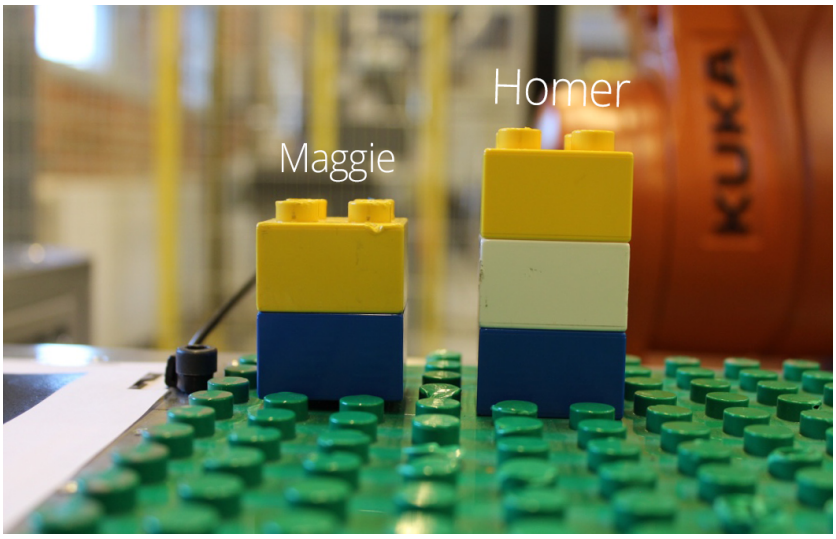


Figure 11: *Maggie and Homer Figures.*

5 Conclusion

In this report, the steps to build the main Simpsons' characters with Duplo Lego Bricks, using a KUKA robot have been described. The program works by acquiring the positions and the colors of Lego bricks with the help of a camera and builds Simpsons' characters. The user has the option to select the type and the number of characters. During the brick extraction process, some difficulties with the gripper were encountered, caused by the position change that was performed in order to rotate the tool and pick up the bricks. However, this issue has been solved by rotating only one specific joint of the robot instead of modifying the orientation of the tool. The final result is a functioning program that sorts Lego bricks into the desired Simpsons' characters.