Autonomous Mechatronic Systems Report

Group 5: Xu Fei, Carlos Fuentes, Qiu Yinan

Abstract:

Automation of mechanical environments is getting more important every day, reducing production times, eliminating mistakes and increasing productivity in multiple industries.

This course has focus on giving us a basic knowledge in sensor and actuator integration in a mechatronic system. We built and programmed an autonomous robot, and it was tested during an examination and a competition. The robot was built using limited number of Lego parts, sensors, actuators, color-camera and one DSP board. The test environment consist on a 2 x 2 meters playground with 6 boxes.

As a result of the project, we will explain how a simple design becomes faster and more reliable than a complex one for this task.

1. Introduction

For this project we are using Lego Technic. Technic is a line of Lego interconnecting plastic rods and parts. The purpose of this series is to create more advanced models with more complex movable arms, such as machines with wheels, in addition to the simpler brick-building properties of normal Lego. Technic sets are often characterized by the presence of special pieces, such as gears, axles, pins, and beams. Some sets also come with pneumatic pieces or electric motors.

Lego Technic seemed to be the perfect tool for our project, although, we didn't use electric motors provided by lego, but usual motors, we still used gears, pins, chains and other tools from Technics. The whole project can be divided into three parts. First, the image process part, the second part is the construction, and the last part is strategy. We will describe them in deep in the following point.

The purpose of the robot is to turn at least 3 of the 6 boxes in the playground, so their face with a 1 painted on it, looks to the robot side. For this we use a color camera and analyze the pictures taking filtering the colors, since the painted numbers are in a color very distinguishable from the boxes. As part of the experiment, several switches and 4 motors were used, no other sensor or actuator, besides the camera, was involved.

2. Methodology and construction

First of all, the robot must be able to distinguish the information that the box shows us. This is the image processing part.

a) Image processing

The robot is provided with a color-camera, which will feed the code with color pictures of 240x320px. From this image we have to extract the information to know what is the robot seeing. We divided this in 3 possibilities

1. A box with a 0 facing us.
2. A box with a 1 facing us.
3. Space between boxes.

To differentiate the possibilities we convert the image from color to a binary image, selecting a color and painting it as white in the binary image, and painting black the rest. The process to do this is to loop through all the pixels in the picture taking the RGB value of every pixel. Having the RGB value we can measure the distance in the RGB color space from the desired color to the one that we have just measured.

The distance between two colors is given by the formula:

$$distance = \sqrt{\frac{1}{3}(red0 - red1)^2 * \frac{1}{3}(green0 - green1)^2 * \frac{1}{3}(blue0 - blue1)^2}$$

Formula1. RGB distance

Being color0 and color1 the rgb values of the color read and the desired color.

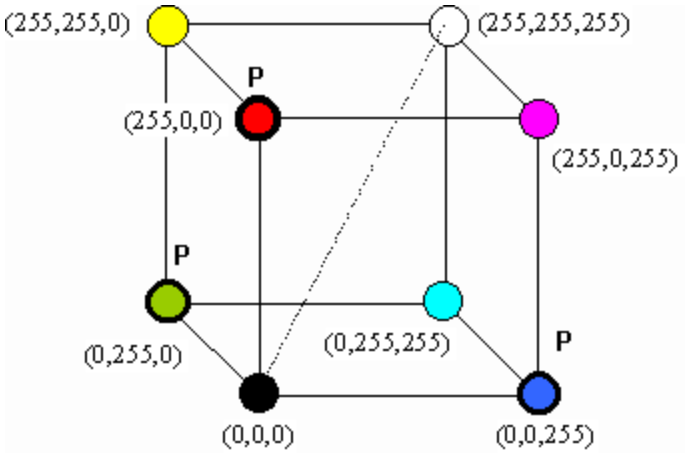

Figure 1. RGB Color Space

For computation purposes we didn't apply last formula, but a simplified one instead:

$$distance = (red0 - red1)^2 * (green0 - green1)^2 * (blue0 - blue1)^2$$

Figure 2. RGB distance simplified

This way we save computational power and there is not practical difference. The only thing we have to do is to increase the distance to compare. In our case we ended, after a lot of empirical tests, with a "simplified distance" of 6000 that gave us the best results.

Color segmentation:

While looping through all the pixels, if the pixel being examined was in the range of the "simplified distance", that pixel becomes a 1 (white), and its added to the histogram of rows and columns, if not, the pixel becomes a 0 (black). This histogram is later used to measure the size of the object detected, and it's boundaries. The camera have some noise, and this is shown on the histogram by some 1s appearing where the noise is. To avoid the noise we have to establish a minimum threshold, where any number under that threshold is not took into account for the boundaries. As the width and height is different on the image, we took different thresholds, 15 pixels for height and 20 for width. These numbers seemed to work quite well for the amount of noise that we were experimenting.

Based on the histogram get the X center of the blue pixels appearing on the image, the width and height of the blue part, and the number of blue

pixels. Only if the center of the blue pixels is in the center of the image we will continue processing, as we will explain later on on the strategy. Assuming the blue number is in the center of the camera, the next move is to know if the number in front of us is a 0 or a 1. To do this, and using the data extracted before, we do the ratio between width and height. This value should be around 1 for the 0, and less than 0.5 for the number 1.

Normalization:

This approach have, as a major inconvenience, that is very sensible to the light. To solve this, is possible to normalize the image after we get it. The normalization process consist on summing all values red(R), green(G) and blue(B) and substitute them by their value divided by the total. So RGB becomes rgb:

$$T = R+G+B;$$

$$r = R/T \quad g = G/T \quad b = G/T$$

This still has a problem with luminosity, since dark pixels can be confused and treated as black color. During our experiments we didn't find this method useful enough regarding the results obtained, so we didn't apply normalization on our robot.

b) Construction

For the different steps of the course, different robots were built. The first constructions consisted on a 4 wheeled robot, with the wheels in the center of the sides of the robot, so it could turn over his own center. With this initial design we passed the first tasks of the course. This tasks consisted on analyzing the image obtained by the camera, constructing a gear box, and follow an object placed in front of the camera using the leds and the robot. The schema of the robot is shown in figure 2.

After the first stages of the course, we destroyed the first design and started building the final robot.

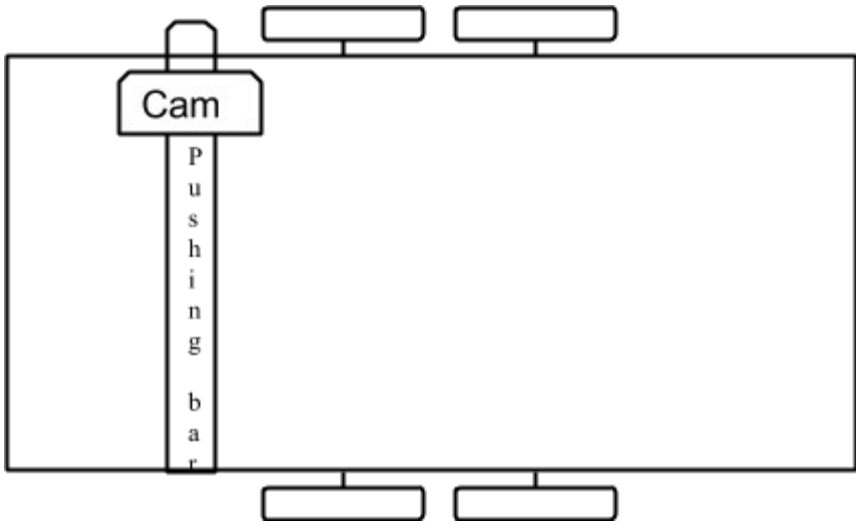

Figure 2. First robot design

On the final design we decided to built a 2 direction robot, so it could move only forward-backwards or right-left. The reason to do this was to find precision on the movement side to side. This was very important for us, since we use a hitting bar to turn the boxes, which has a limited range of motion, and if we are aiming for speed, we shouldn't be recalibrating the distance too often. A picture of the final robot can be seen on Figure 3.

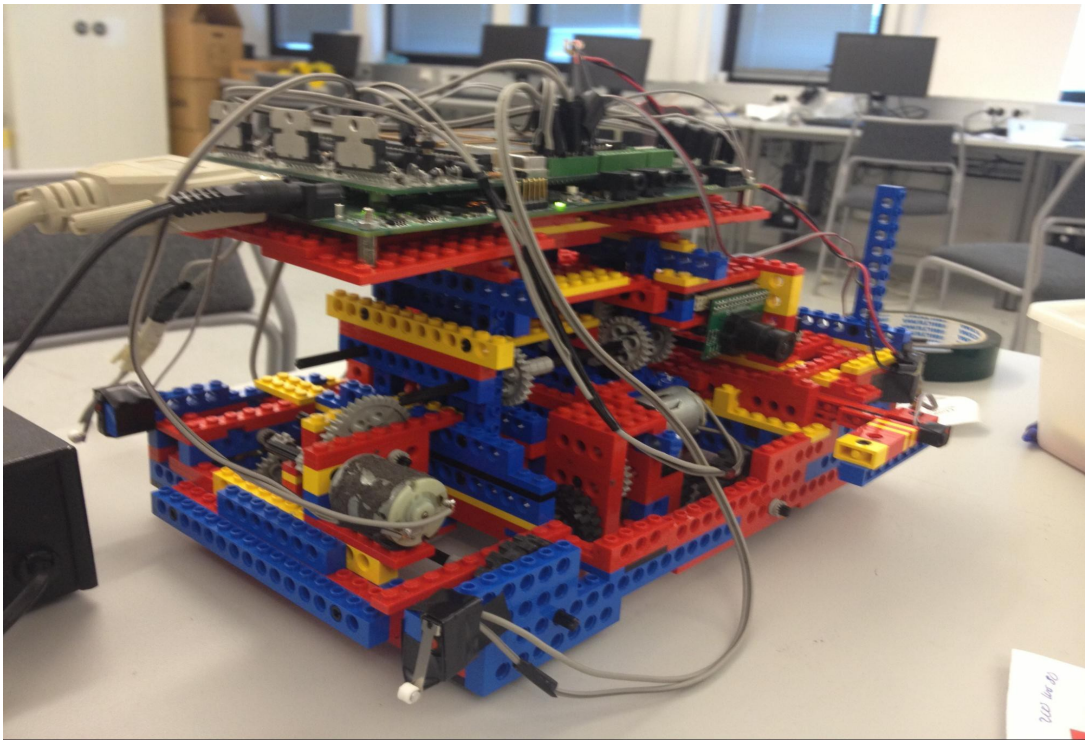
Figure 3. Final robot design

The manipulator:

The way of moving the boxes, as commented before, is a hitting bar. With a length of 12cm and a switch placed in the front of the bar. It is moved by a motor and a gearbox with a 1/48 ratio. At the back part of the bar there is another switch to know when the bar is totally retracted. To guide the bar, a rail was built and a token placed on the bar, since the lego pieces are not totally straight.

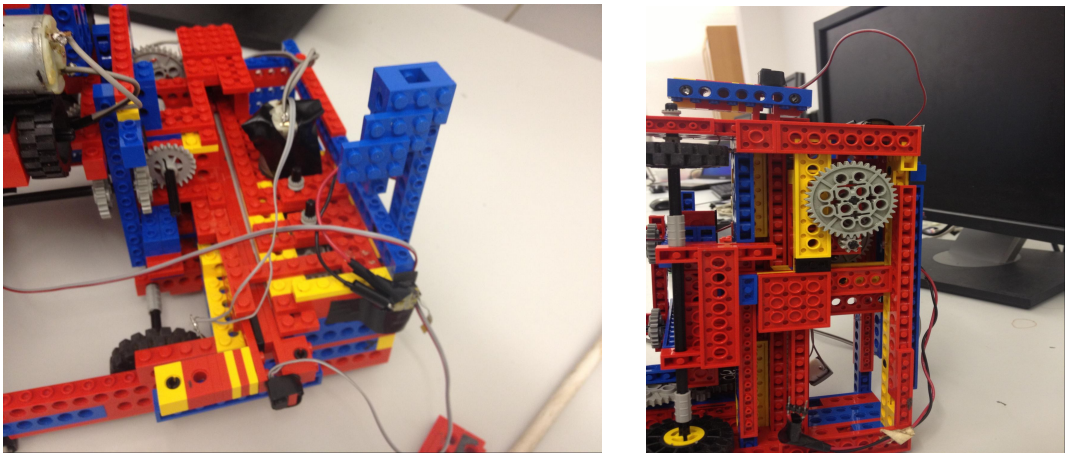

Figure 4. Pushing bar details. Upper (left) and down (right) point of views.

Movement:

The movement is separated into left-right and forward-backwards. The left-right movement is placed on the right side of the robot (left side of the figure 3) - all parts of the robot are different modules, so they can be fixed or changed easily. It consist on a gearbox 1/64 ratio connected to a long axis that join the 2 left wheels. The power is given by a single motor as shown in figure 5.

The forward-backwards movement is handle by a separated small vehicle, attached to the main robot by a small cage with the particularity that can go up and down on the cage. This mechanism allows it to lift the robot or lift the small car, so by lifting one or the other, the movement can be in one direction or the other, but only one at a time. The limits upwards

and downwards of the small vehicle is limited by physical constraints, but also by 2 switches that controls when to stop the motor.

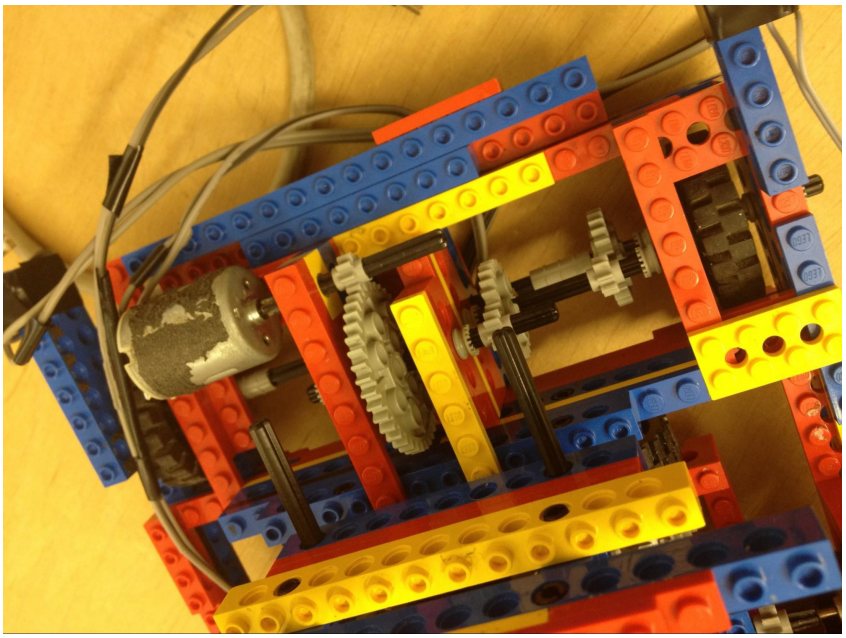

Figure 5. Detail of the left-right movement system.

In the figure 6 we can see the the details of the small car, the mechanism to raise it, and some of the elements used to control the process. The motor to lift the whole robot had to deal with a lot of friction and weight in the process, so for this task we decreased the ratio to 1/96, what gave us the power to lift smoothly the robot, and also the friction to keep the robot up while moving forward, without having to apply any power on the motor.

In the case of the small vehicle, the ratio was more moderate, since the construction was clean enough and, besides holding the whole robot up, the power was enough to move the wheels, that were also smaller than the main ones.

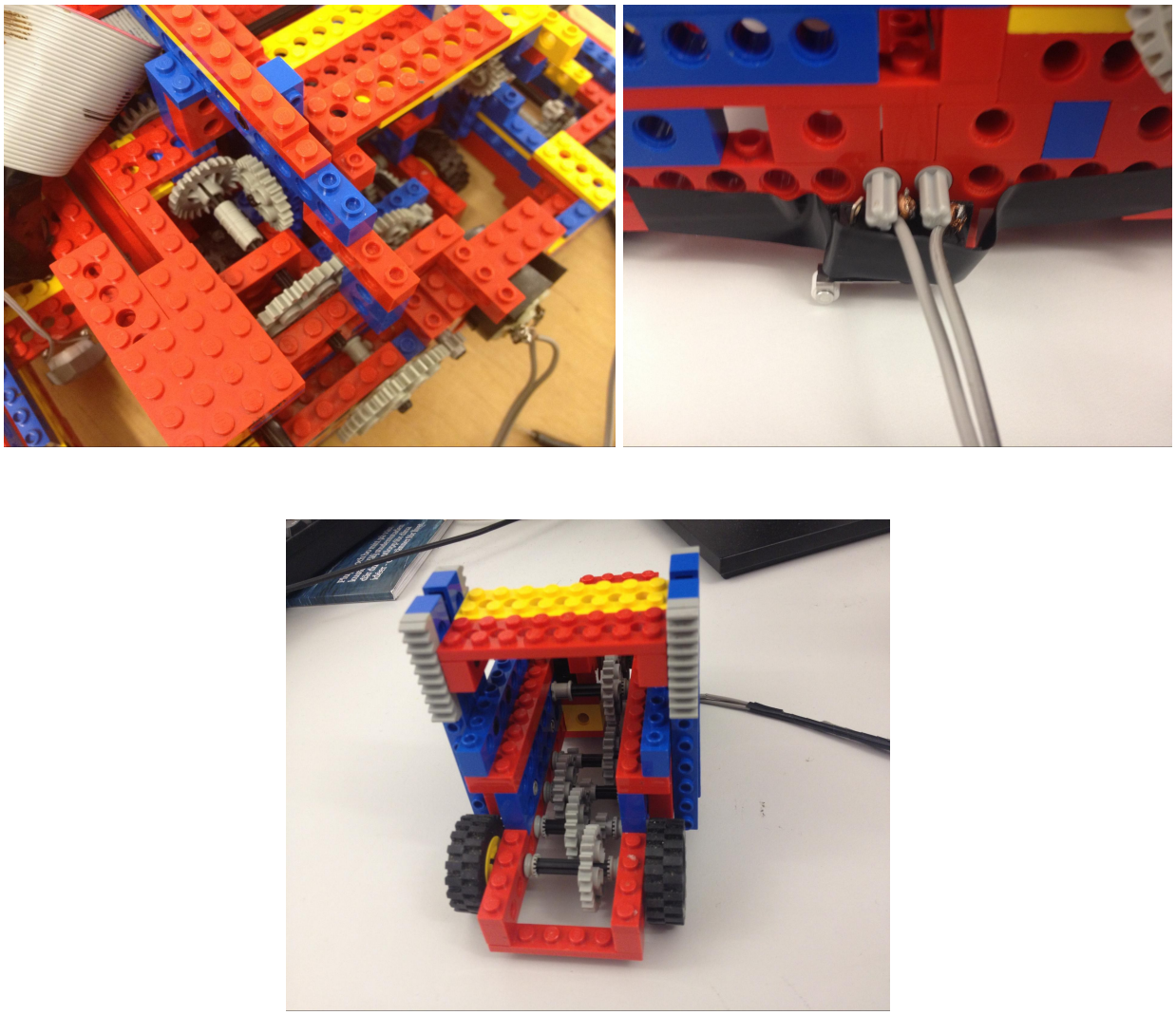

Figure 6. Mechanism to lift the small robot (Left-top). Switch to stop lifting the robot (Top-right). Small robot. (Bottom)

c) Strategy and coding

From the beginning our strategy was clear, we move fast but securing the movements.

Regarding the image strategy, the code implemented for the exam was based on the blue color recognition. If the camera detected enough blue pixels on the image, that means that we are in front of a box, if not, should mean that we are in between the boxes.

So the flow of the whole process was as follow:

1. As the robot starts in the middle of the left wall, we first take pictures and find the blue of the left box.
2. Once found the center of the blue, we move the robot sidewise to center the camera (over the pushing bar - Figure 7) with the box.
3. Being centered, we lift the robot over the small car and take the pushing bar out.
4. Start moving forward until the pushing bar touch the box (using the switch at the front of the bar).
5. Put the bar in again and the robot on the ground, lifting the small car inside the robot.
6. Once here, we start taking images, and, if everything went well, we should be in front of a number. We analyze the image 4 times and, if the majority of the analysis says that what we have in front of the robot is a 0, the robot hit the box to turn it, using the pushing bar.
7. The robot continues step 6 until it finds that it has a 1 on the box. At this moment the robot moves in the direction (left or right) established (changes when it reach the sides).
8. If the robot finds blue on the image after moving, it performs small movements left or right to center the blue on the image. If not, it keeps moving towards the direction established.
9. After having the blue center on the image we repeat step 6.
10. If the robot reach the sides, the established direction changes.

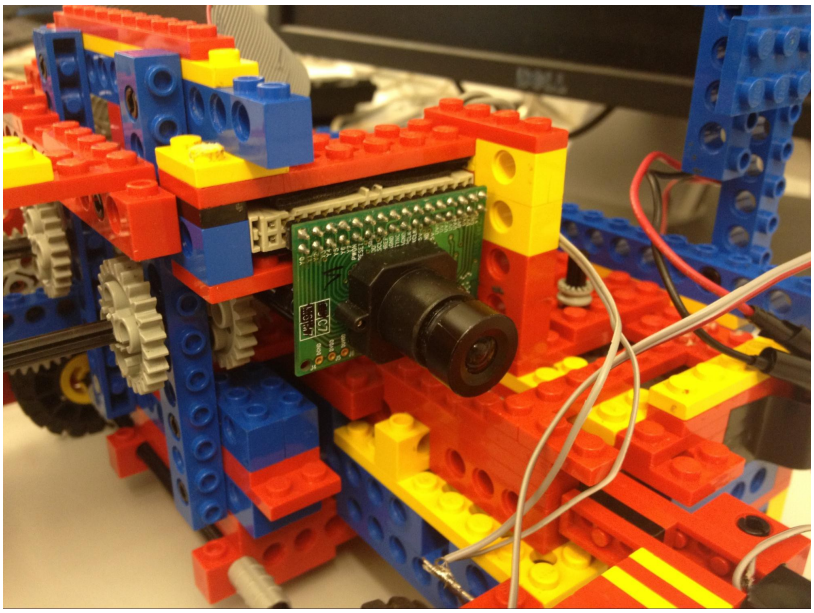

Figure 7. Camera place (over the pushing bar)

Controlling the limits:

All the elements were double checked and didn't rely only on the switches to control them, since we experience some infrequent problems with the setting of the switches, probably because of the poor way of attaching them to the robot. So to solve this, we used also timers to control the power on the motor that push the bar in and out. And the small robot up and down. Physical limits were also used when possible, but usually is not a good idea to rely on them, since the power of the motors can, eventually, break the gearboxes.

3. Results and testing

While debugging the robot we experienced some problems with the camera, but after solving them (hardware problems), the robot worked smoothly. On the tests before the exam the robot managed to turn the six boxes in under a minute. The problems came after flashing the board and removing the cables. It seemed that the weight of the cable kept it straight, and when removed it, the robot slightly turned right while moving. For this

reason the robot only turned 3 boxes out of 6 on the exam, on the first attempt. For the examination, only 3 boxes were needed to pass the exam, so no more official attempts were taken.

After the exam some parts of the algorithm were improved, and the movement of the robot fixed to be straight, in order to participate on the competition.

4. Conclusion

In this project we have learned how to build a simple autonomous robot, and experience the unexpected problems that we have to face while building it. We also experienced how to work with limited resources.

We used and integrated in the same construction, sensors and interactors, camera, switches, motors and mobile elements, connecting them and making them work together. We learned the importance of the calibration of these elements and how to take into account the external elements at the time of reading the sensors.