**Groupe 8 – EV3Meg Robot**



Members: LI Jiaqi, SHI Libin, TAO Kai, ZHANG Peidong

# Member and Task Introduction

## 1.1 Group Members

*SY1624127 TAO Kai (Hugo)*

*SY1624132 LI Jiaqi (Victor)*

*SY1624129 ZHANG Peidong (Paul)*

*ZY1624134 SHI Libin (Olivier)*

## 1.2 Distribution of tasks

Tracker assembly with LEGO: *all the members;*

Test of movement: *TAO Kai, ZHANG Peidong;*

Simulation with MATLAB: *all the members;*

Design of trace control and Programming: *Li Jiaqi, SHI Libin;*

Test of control: *all the members*.

# Planning

Model Selection & Robot Assembly: **17th, April**

Movement Test: **18th ~ 19th, April**

Simulation with MATLAB: **25th ~ 26th, April**

Algorithm Design & Programming: **26th, April ~ 8th, May**

Control Test: **8th ~ 11th, May**

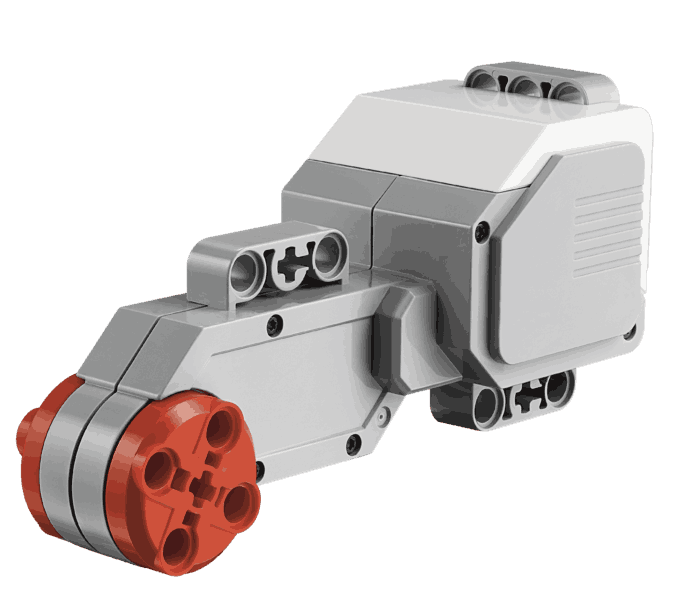
# Robot Structure

## 3.1 Components

Programmable controller x 1



large-size motors (control of wheels) x 2



medium-size motor (control of arms) x 1



ultrasonic sensor x 1



color sensor x 1



Wheels x 4



## 3.2 Assembled Robot





## 3.3 Functions

We build an automatic control tracker that can use a color sensor to follow different colors, detect objects in its way, and react.

**Function 1:** basic movements. The robot can move towards a direction, change its direction, and stop movement.

**Function 2:** use a ultrasonic sensor to avoid obstacles while moving. The robot will turn around and change its direction of movement when it detects an obstacle in its way within a predefined distance.

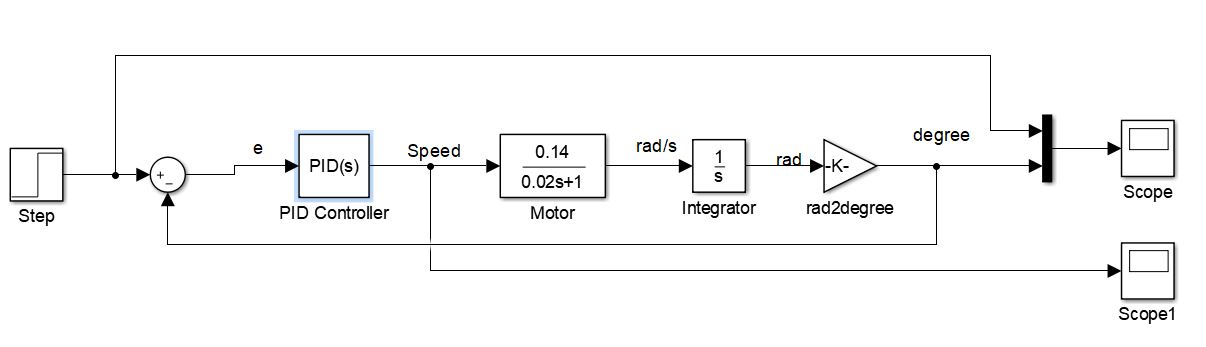
**Function 3:** use a color sensor to track a path of specified color. The robot will detect the color of the path and keep following the path without too much deviation.

# Evaluation of Performances

## 4.1 Simulation with SIMULINK

In order to realize the functions of our robot, the most basic and important step is control the motor to rotate a precious degree (associated with our matlab function TURN). All the functions of basic movement, obstacle avoidance and path tracking is based on the TURN function. Unfortunately, we cannot directly control the motor rotate a certain degree, but can only control the motor’s speed. So we use PID to realize the basic function TURN.

Firstly, we use the Simulink library in Matlab to simulate the system, as the following figure shows:



The input of this system is the degree which we want the motor to rotate. The output is the real ratation degree of motor, which can be read by the Encoder in motor. In fact, it can be read by **readRatation(motor)** in Matlab LEGOEV3 module. The difference **e** between system input(control degree) and system output(real ratation degree) is receieved by the PID module, and the output of PID is motor’s speed. The transfert function of PID module can be expressed as:

where is reperspectively the coefficient of Proportional, Intergral, Derivative, ~~and means the Filter coefficient~~.

The ouput of PID, motor’s speed, is used to control the motor. The motor can be simulated as an one-order system and the parameters are given by our teacher:

Given a certain input of speed, the motor rotates in a specific angular velocity. We can get the rotation degree(in rad) by an intergrator. We convert it to degree with a gain module:

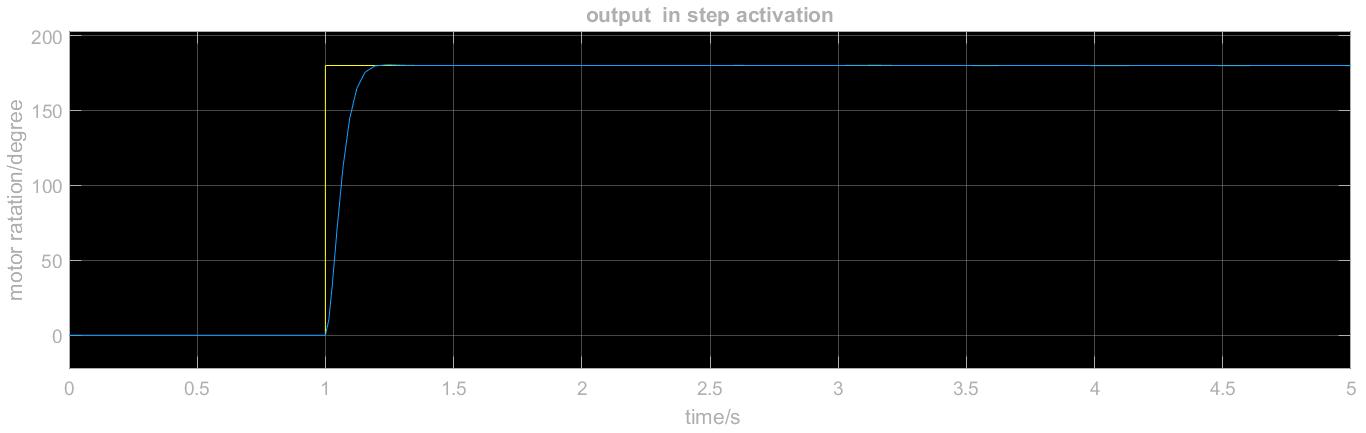
So we build the closed-circle system.

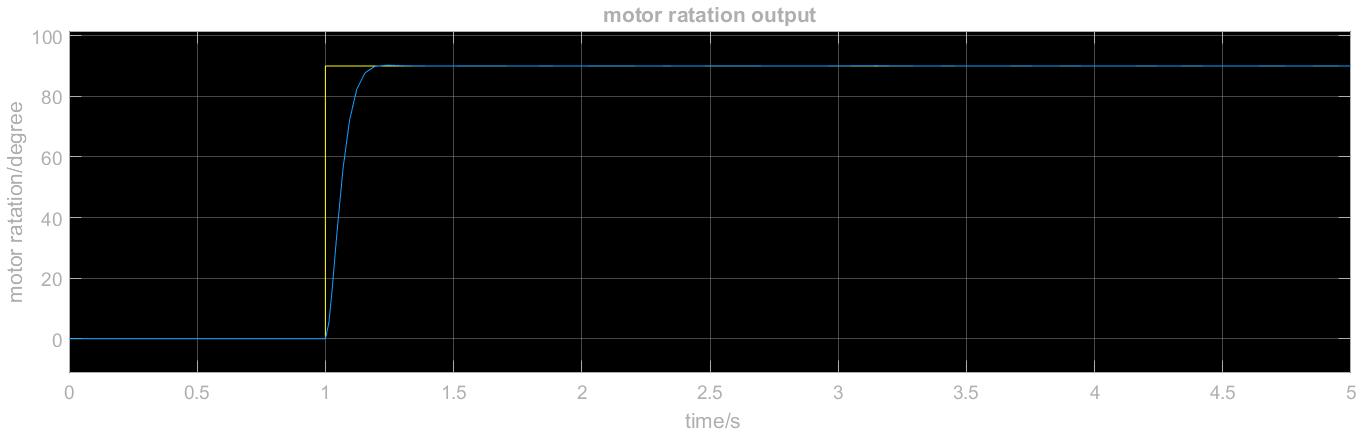
We give the whole simulation a step function as input, representing the degree we want the motor to rotates.

During the simulation, we adjust the parameters of PID controller() to satisfy the system performance (precision, rapidity). We use step functions as the input of system:

After the adjustment of parametes during simulation, we got the suitable values of PID parameters:

The system output of a step activation shows as:





## 4.2 Real output of system

Using the results of simulations

# Problem and Analysis

Although we have realized the three functions to a certain extent, there are still several problems that we want to discuss here. These problems occurred during the process of simulation, programming and test. Some have been solved successfully, others may yet still be improved in the future. Below are presented several problems that we think are important along with our analysis.

## 5.1 Deviation of movement along a straight line

This is probably caused by the slight inconsistence of speeds between the left wheel and the right wheel. However, this phenomenem does not appear obvious during the test process of obstacle avoiding and path tracking, and is only observable when our robot moves straight for a quite long distance (let’s say more than about 2 meters).

## 5.2 Deviation of rotation while changing direction

This problem occurs in the test process of obstacle avoiding. At first trying, the deviation of turning angle can be like when a command of is given. After perceiving this problem, we have improved our algorithm to minimize this deviation. At the end of our test process, this deviation is reduced to less than given a command of .

## 5.3 Lack of robustness control

A simulation using PID control is conducted in our work. However, the robustness control is left out considering the complexity of simulation. Thus, our robot may not be robust if any disturbance is involved during its movement.

# Conclusion