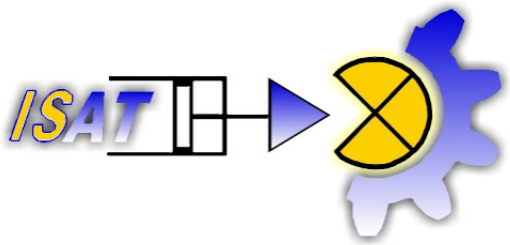

UE A2073 : Robotique PARTIE II

Professeur : COSTA Emile

RAPPORT D'AVANCEMENT

Etudiants : ESSALHI Younes
HEMELEERS Emile
KORKUT Caner
ZEUKENG Ronald

Séance III

1 Introduction

A l'ordre du jour, nous traiterons les points suivants :

- Réalisation d'un *mind map*
- Réception et montage d'un second prototype pour les tests à effectuer ;
- Adaptation du code destiné au mouvement du robot ;
- Choix du pont H approprié à notre circuit ;
- Réalisation du travail demandé sur Fusion 360 ;
- Prochains objectifs

Ce compte-rendu a pour but de faire le bilan sur les objectifs atteints depuis la dernière séance de cours et de fixer de nouveaux objectifs à venir.

2 Réalisation d'un *mind map*

Afin d'avoir une vue globale sur le Projet Robotique, nous avons estimé qu'il était nécessaire de faire une carte mentale des informations que nous avions à disposition. Ceci nous sera utile lorsque nous ferons un bilan d'avancement ultérieurement.

Cette carte a été annexée à ce rapport.

3 Réception et montage d'un second prototype pour les tests à effectuer

Nous avions commandé un second prototype afin d'optimiser les essais. Ainsi, nous pensons avoir un modèle qui sera plus ou moins fixe, où les pièces viendront se rajouter au fur et à mesure tandis que le second prototype servira aux tests. Ce dernier nous servira pour essayer les scripts à la volée, sans que le débranchement d'un composant n'ait d'incidence sur le projet. Il s'agit d'une forme de versionnage hardware auquel nous nous sommes mis d'accord.

Après avoir terminé le projet final, un couvercle viendra se poser sur le robot afin de rendre le visuel plus agréable.

4 Adaptation du code destiné au robot

Le code que nous avons pris sur internet [1] (voir annexesB) nous a permis de faire tourner les roues dans les deux sens et ce à une vitesse variable par l'intermédiaire du $L293D$. Cependant, il ne se mouvoient pas à des vitesses indépendantes et cela ne nous permet pas de tourner à gauche ou à droite.

Pour parer à ce problème, nous avons prévu d'adapter le programme fourni.

5 Choix du pont H approprié à notre circuit

Pour le choix du pont H destiné aux roues du robot, nous avions le choix entre le $L298N$ et $L293D$. Après avoir analysé les données techniques liées aux différents composants (voir annexes A), nous avons décidé d'utiliser le $L293D$ pour sa compacité et son faible coût. Cependant, nous n'excluons pas l'utilisation du $L298N$ dans le cas où nos moteurs demanderaient 2 fois plus de courant, où notre pont H dissiperait trop de chaleur, etc...

6 Réalisation du travail demandé sur Fusion 360

Pour la conception de notre maquette, nous avons décidé d'utiliser deux plaques entre lesquelles se trouveront les différents composants du robot tels que le moteur, le circuit électrique etc.

Ainsi, sur Fusion 360 nous avons déjà construit certains éléments comme les deux plaques, les roues et les tiges métalliques qui maintiendront les deux plaques, le capteur ultrason, les moteurs DC 5V, l'Arduino, la breadboard.

L'étapes suivante est de de dimensionner parfaitement les composants tel que le circuit électrique imprimé qui se trouvera entres les deux plaques pour pouvoir les dessiner ou choisir sur internet et d'ensuite assembler le tout sur Fusion afin de finaliser la maquette.

Les fichiers sont disponibles en annexe à ce rapport.

1. http ://bit.ly/2Ih4hvm

7 Tâches à réaliser pour la prochaine séance

Nous avons constaté que la manière la plus efficace d'atteindre nos objectifs était de travailler par énumération. Par cela, nous entendons qu'en nous fixant des petits objectifs chaque semaine, nous sommes plus efficaces que si nous donnions une échéance finale pour une date ultérieure. Parmi les prochaines tâches à réaliser, nous pouvons citer les points suivants :

— Faire tourner les roues indépendamment ;
— Intégrer la communication Bluethoot ;
— Intégrer le RFID dans le code avec les mouvements ;
— Continuer le travail sur Fusion360
— Décisions à prendre sur le projet final

A ANNEXE - Datasheet *L298N* et *L293D*

L298N

ABSOLUTE MAXIMUM RATINGS

Symbol	Parameter	Value	Unit
V_S	Power Supply	50	V
V_{SS}	Logic Supply Voltage	7	V
V_I,V_{en}	Input and Enable Voltage	−0.3 to 7	V
I_O	Peak Output Current (each Channel) – Non Repetitive (t = 100 µs) –Repetitive (80% on –20% off; t_{on} = 10ms) –DC Operation	 3 2.5 2	 A A A
V_{sens}	Sensing Voltage	−1 to 2.3	V
P_{tot}	Total Power Dissipation (T_{case} =75 °C)	25	W
T_{op}	Junction Operating Temperature	−25 to 130	°C
T_{stg},T_j	Storage and Junction Temperature	−40 to 150	°C

TEXAS INSTRUMENTS

L293, L293D

SLRS008D – SEPTEMBER 1986 – REVISED JANUARY 2016 www.ti.com

6 Specifications

6.1 Absolute Maximum Ratings

over operating free-air temperature range (unless otherwise noted)(1)

	MIN	MAX	UNIT
Supply voltage, V_{CC1} (2)		36	V
Output supply voltage, V_{CC2}		36	V
Input voltage, V_I		7	V
Output voltage, V_O	–3	V_{CC2} + 3	V
Peak output current, I_O (nonrepetitive, t ≤ 5 ms): L293	–2	2	A
Peak output current, I_O (nonrepetitive, t ≤ 100 µs): L293D	–1.2	1.2	A
Continuous output current, I_O: L293	–1	1	A
Continuous output current, I_O: L293D	–600	600	mA
Maximum junction temperature, T_J		150	°C
Storage temperature, T_{stg}	–65	150	°C

(1) Stresses beyond those listed under *Absolute Maximum Ratings* may cause permanent damage to the device. These are stress ratings only, which do not imply functional operation of the device at these or any other conditions beyond those indicated under *Recommended*

FIGURE 1 – Datasheet des deux composants comparés

B ANNEXE - Code à modifier pris sur Internet

```
//Joystik
const int JOYvertical = 1; // analogique donc A1
const int JOYhorizontal = 2; // analogique donc A2
const int JOYselect = 2; // digital 2
int joyValue = 0;
int joyValueMax = 1023;
int joyValueMin = 0;
int joyValueMid = 512;
int joyValueMidUpper = joyValueMid + 20;
int joyValueMidLower = joyValueMid - 20;

//MOTEURS
int motor1_enablePin = 11; //pwm
int motor1_in1Pin = 13;
int motor1_in2Pin = 12;
int motor1Speed = 0;
int motor1SpeedMin = 100;
int motor1SpeedMax = 255;

int motor2_enablePin = 10; //pwm
int motor2_in1Pin = 8;
int motor2_in2Pin = 7;
int motor2Speed = 0;
int motor2SpeedMin = 100;
int motor2SpeedMax = 255;

void setup()
{
  //Joystik lecture
  pinMode(JOYvertical,INPUT);
  pinMode(JOYhorizontal,INPUT);
  pinMode(JOYselect,INPUT);
  digitalWrite(JOYselect,HIGH); // pull-up
  Serial.begin(9600);

  //MOTEURS
  //on initialise les pins du moteur 1
  pinMode(motor1_in1Pin, OUTPUT);
  pinMode(motor1_in2Pin, OUTPUT);
  pinMode(motor1_enablePin, OUTPUT);
  //on initialise les pins du moteur 2
```

```
  pinMode(motor2_in1Pin, OUTPUT);
  pinMode(motor2_in2Pin, OUTPUT);
  pinMode(motor2_enablePin, OUTPUT);
}
void loop()
{
  //Joystik
  int AvAr, GaDr, select;
  // lecture des valeurs du joystick
  AvAr = analogRead(JOYvertical);     // entre 0 et 1023
  GaDr = analogRead(JOYhorizontal); // entre 0 et 1023
  select = digitalRead(JOYselect);       // HIGH (1)si non appuye, LOW
     (0)sinon

//MOTEURS 1
    joyValue = analogRead(JOYvertical);

    if(joyValue > joyValueMidUpper) //Forward
    {
        motor1Speed = map(joyValue, joyValueMidUpper, joyValueMax,
   motor1SpeedMin, motor1SpeedMax);
        Motor1Forward(motor1Speed);
    }
    else if(joyValue < joyValueMidLower) //Backward
    {
        motor1Speed = map(joyValue, joyValueMidLower, joyValueMin,
   motor1SpeedMin, motor1SpeedMax);
        Motor1Backward(motor1Speed);
    }
    //joyValue Between joyValueMidLower - joyValueMidUpper.
    //Need some range here, because joystick sometime not in  exact
   center.
    else
    {
       Motor1Stop();
    }

     //MOTEURS 2
    joyValue = analogRead(JOYhorizontal);

    if(joyValue > joyValueMidUpper) //Forward
    {
        motor2Speed = map(joyValue, joyValueMidUpper, joyValueMax,
```

```
    motor2SpeedMin, motor2SpeedMax);
81          Motor2Forward(motor2Speed);
82      }
83      else if(joyValue < joyValueMidLower) //Backward
84      {
85          motor2Speed = map(joyValue, joyValueMidLower, joyValueMin,
    motor2SpeedMin, motor2SpeedMax);
86          Motor2Backward(motor2Speed);
87      }
88      //joyValue Between joyValueMidLower - joyValueMidUpper.
89      //Need some range here, because joystick sometime not in  exact
    center.
90      else
91      {
92         Motor2Stop();
93      }
94  }
95
96  void Motor1Forward( byte Spd)
97  {
98      digitalWrite(motor1_in1Pin, HIGH);
99      digitalWrite(motor1_in2Pin, LOW);
100     analogWrite(motor1_enablePin, Spd);
101 }
102 void Motor1Backward( byte Spd)
103 {
104     digitalWrite(motor1_in1Pin, LOW);
105     digitalWrite(motor1_in2Pin, HIGH);
106     analogWrite(motor1_enablePin, Spd);
107 }
108 void Motor1Stop()
109 {
110     analogWrite(motor1_enablePin, 0);
111 }
112
113 void Motor2Forward( byte Spd)
114 {
115     digitalWrite(motor2_in1Pin, HIGH);
116     digitalWrite(motor2_in2Pin, LOW);
117     analogWrite(motor2_enablePin, Spd);
118 }
119 void Motor2Backward( byte Spd)
120 {
```

```
121     digitalWrite(motor2_in1Pin, LOW);
122     digitalWrite(motor2_in2Pin, HIGH);
123     analogWrite(motor2_enablePin, Spd);
124 }
125 void Motor2Stop()
126 {
127     analogWrite(motor2_enablePin, 0);
128 }
```