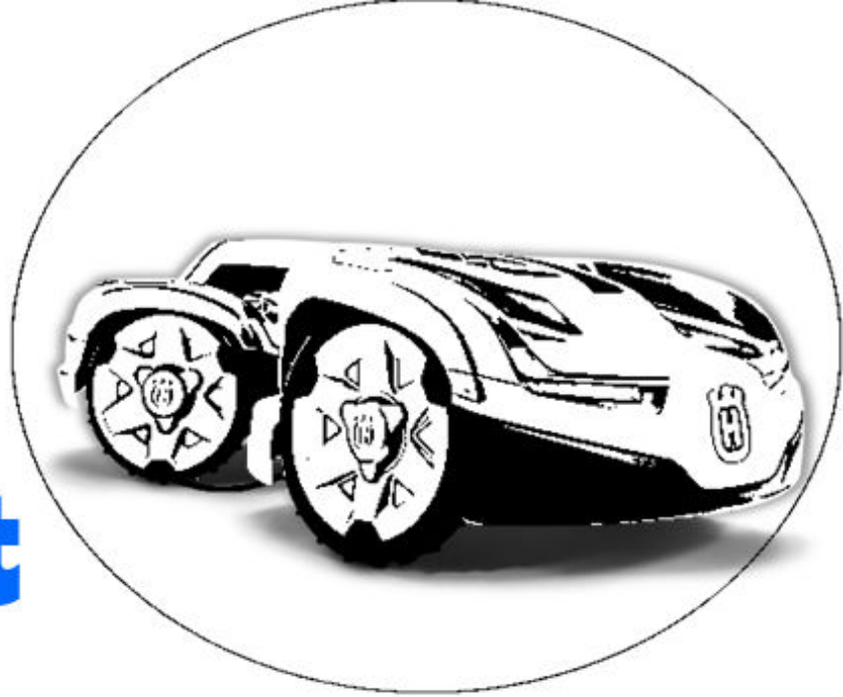
esith
Constructeur de compétences



Grass Cutter robot

Tondeuse robotique pour terrains de golf inclinés

NAJMI YASSIN

Code CNC: SF026T

CENTRE CPGE MOULAY ABDELLAH SAFI

Plan de présentation:

- Introduction:
  - Contextualisation
  - Problématique
  - Cahier des charges
- Les objectifs:
  - Dimensionner le moteur
  - Asservir la vitesse de déplacement du robot
  - Choix et positionnement du capteur d'obstacle
  - Gestion de la trajectoire du robot
  - Dimensionnement du panneau photovoltaïque
- Conclusion

Contextualisation

« Pourquoi « Grass cutter robot »? »

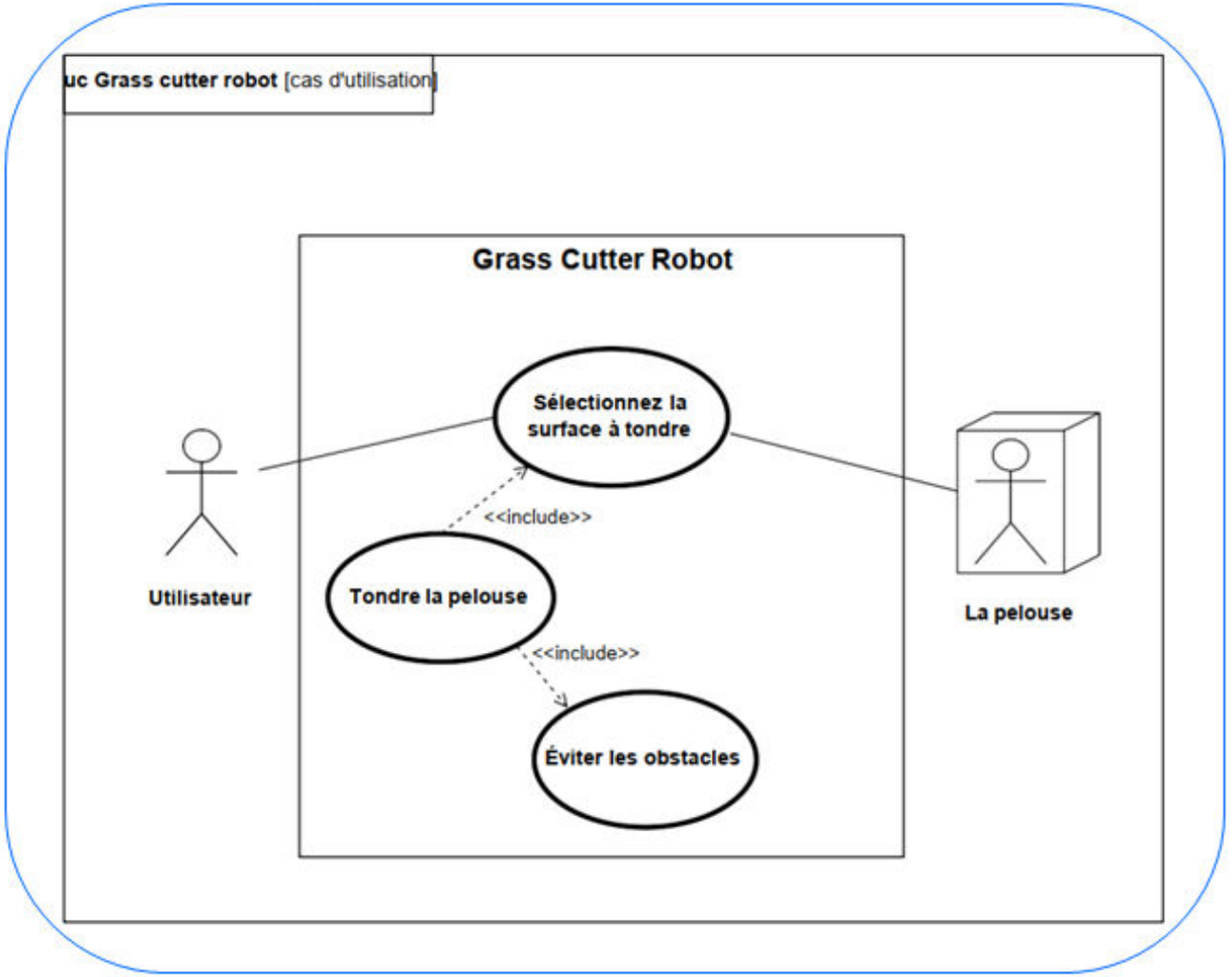
Figure 1: inclinaison de terrain

Problématique

« Comment notre système doit accomplir la tâche de tonte du gazon de terrain sur les pistes sans nécessiter d'intervention humaine, en évitant les obstacles et en tenant compte du respect de l'environnement ? »

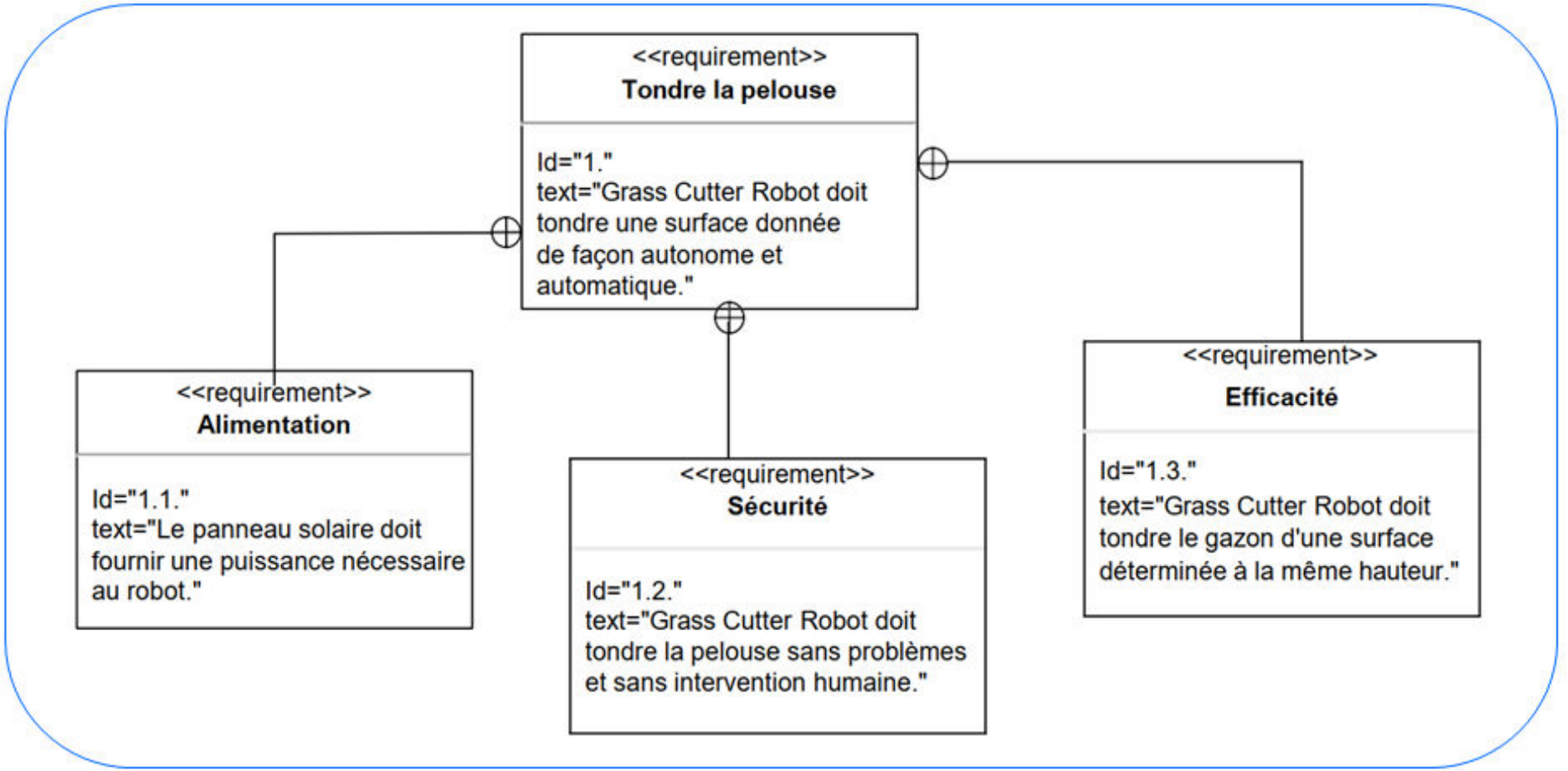
Cahier des charges

Diagramme des cas d'utilisation



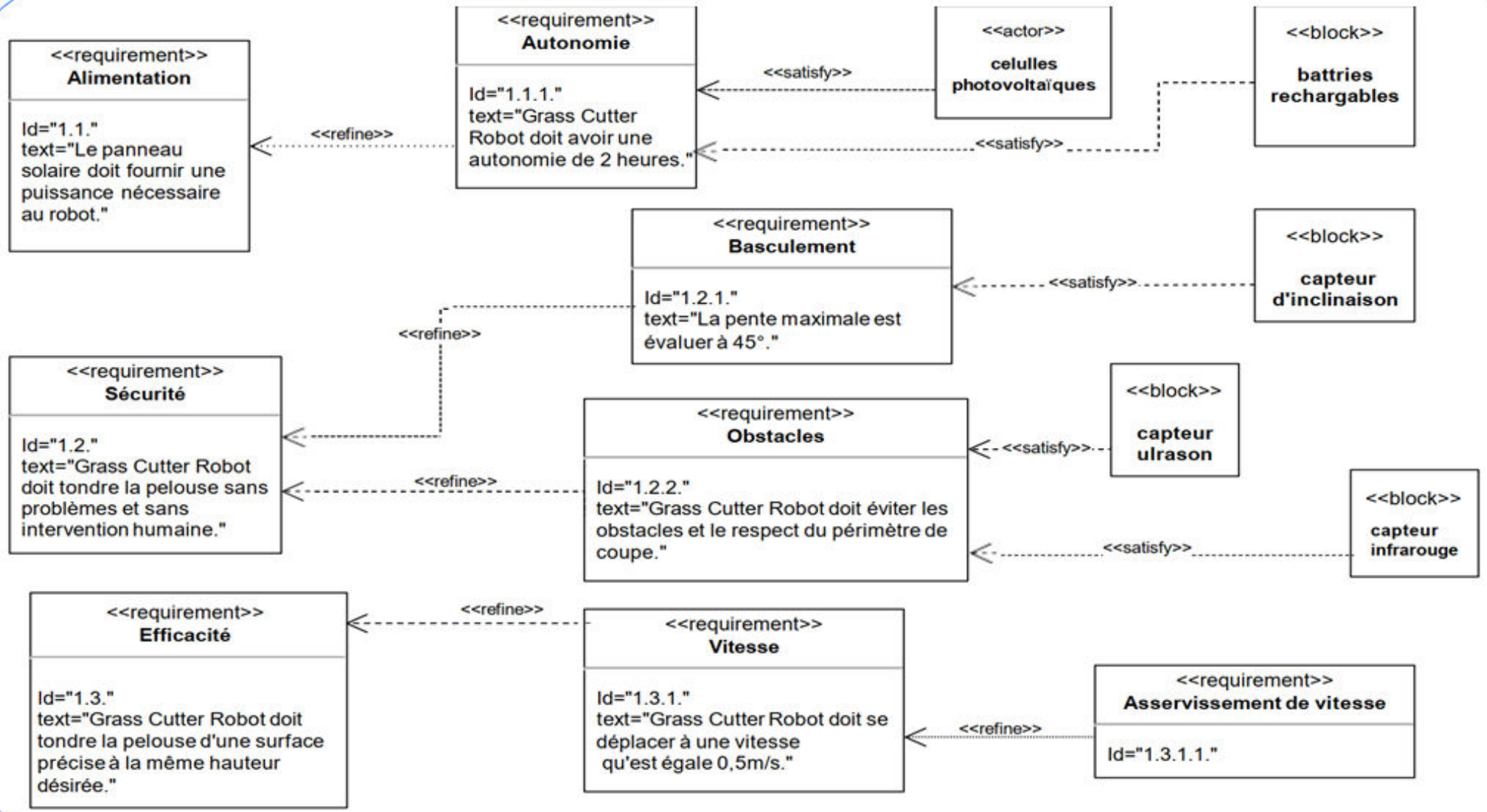
Cahier des charges

Diagramme des exigences



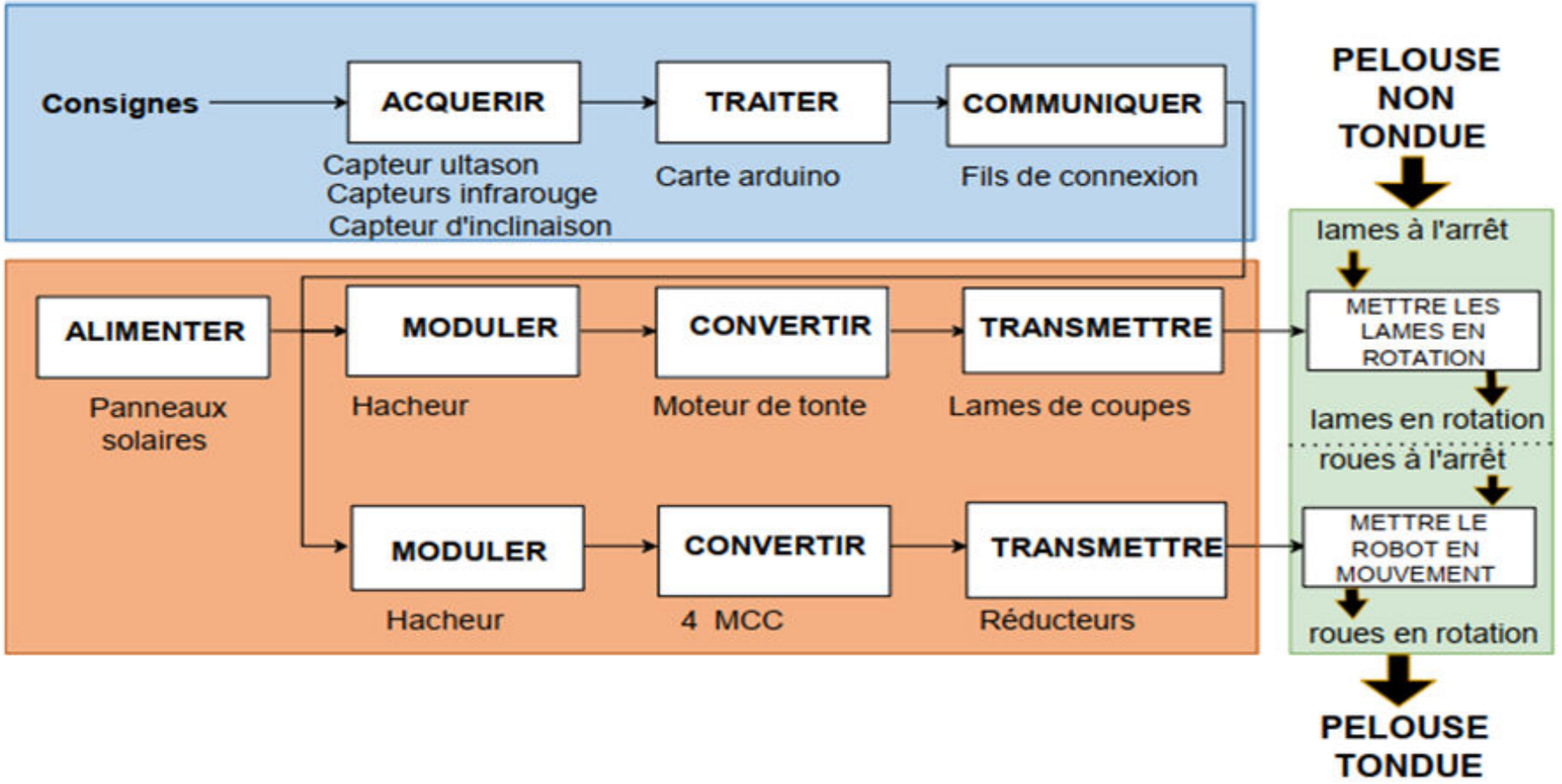
Cahier des charges

Diagramme des exigences



Cahier des charges

Chaine d'information et chaine d'énergie



Objectif 1:

Dimensionner le moteur

Objectif 1: Dimensionner le moteur

modélisation du système

Cahier des charges :

- V=0,5m/s : la vitesse de déplacement du robot
- α= 45 ° : l'angle d'inclinaison maximal du terrain
- M=1 kg : la masse du robot
- R=3,3 cm : le rayon des roues motrices
- g=9,81m/s² : accélération de la pesanteur

Hypothèse :

- Action de l'aire négligeable
- Les roues ne glissent jamais

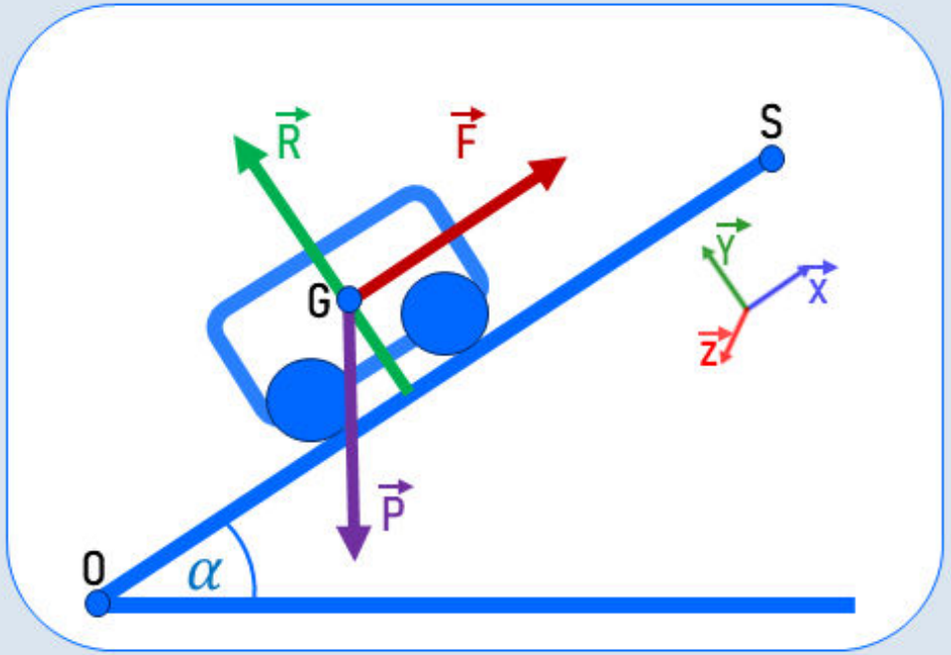


Figure 2: Robot sur un plan inclinée

Le couple à développer par les moteurs

Système étudié:{châssis, roues}

Application de théorème d'énergie cinétique sur le système : $\frac{dEc(S/0)}{dt} = P(\vec{F}) + P(\vec{P}) + P(\vec{R})$

- $P(\vec{F}) = C\omega$ (pour un roue) ➡ $P(\vec{F}) = 4C\omega$
- $P(\vec{P}) = \vec{P}.\vec{V}$ ➡ $P(\vec{P}) = -M.g.R.\sin(\alpha).\omega$ avec $V = \omega.R$
- $P(\vec{R}) = 0$

puisque V=0.5m/s=cst alors : $4C\omega - M.g.R.\sin(\alpha).\omega = 0$ ⇨ $C = 1/4 M.g.R.\sin(\alpha)$

$\alpha max = 45°$

AN: C=0.05722 N.m

Choix des moteurs

On choisit 4 moteurs réducteurs dont le couple : C ≥ 0,0572 Nm

Rated Voltage	Motor Type	Stall Current	No-Load Current	Gear Ratio	No-Load Speed (RPM)	Extrapolated Stall Torque (kg · cm)	Extrapolated Stall Torque (oz · in)	Max Power (W)	Without Encoder	With Encoder
6 V	Low-Power (LP)	2.0 A	100 mA w/o encoder 120 mA with encoder	1:1 (no gearbox)	6200	0.15	2.1	2.1	–	item #4820
				4.4:1	1300	0.63	8.7	2.1	item #1581	item #4821
				9.7:1	630	1.3	18	1.9	item #1582	item #4822
				20.4:1	290	2.5	35	1.9	item #1583	item #4823
				34:1	180	3.9	54	1.7	item #1584	item #4824
				47:1	130	4.8	67	1.5	item #1585	item #4825
				75:1	80	7.5	100	1.5	item #1586	item #4826
				99:1	61	9.1	130	1.4	item #1587	item #4827
				172:1	35	14	190	1.2	item #1588	item #4828
				227:1	27	17	240	1.1	item #1589	item #4829
				378:1	16	25	350	–	item #1590	item #4830
				499:1	12	31	430	–	item #1591	item #4831

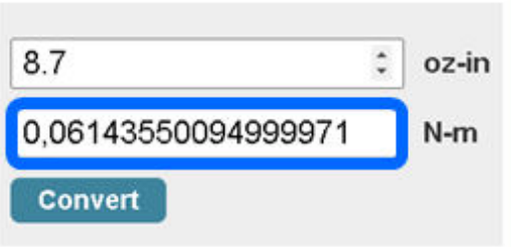
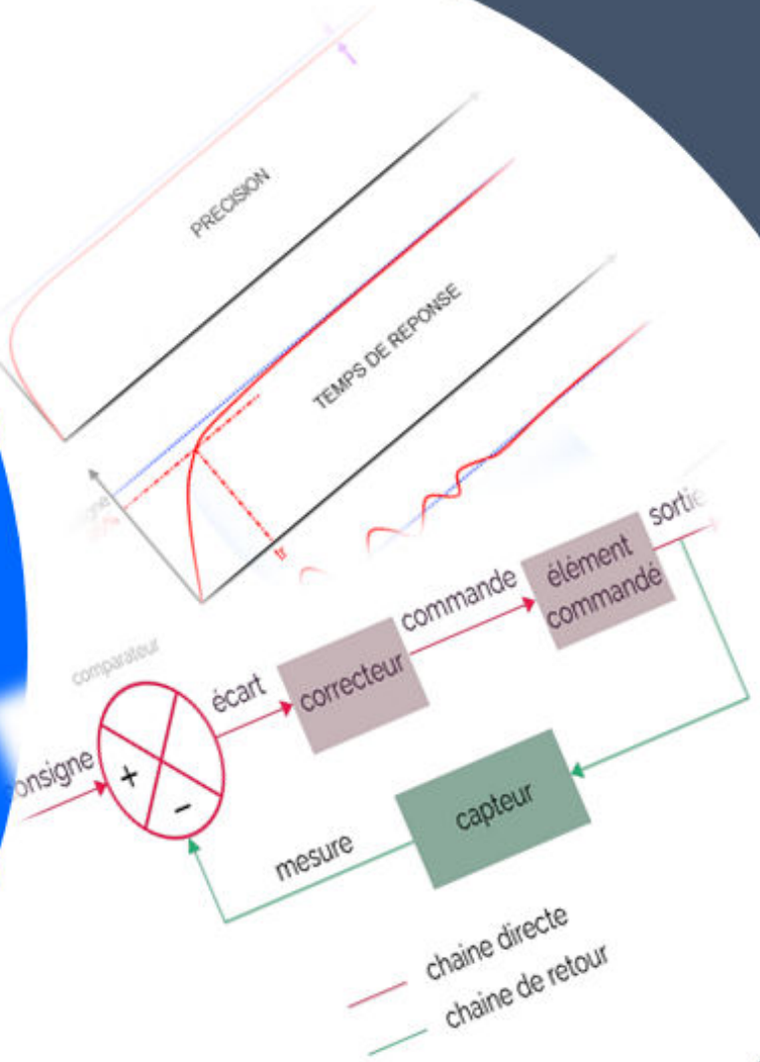


Figure 3: Catalogue extrait du site "Pololu"

Caractéristiques du moteur choisi : Pololu item#4821 , 4:4.1, 120 mA , 6 V, 1300RPM, 48CPR

Objectif 2:

Asservir la vitesse de déplacement du robot



Objectif 2: Asservir la vitesse de déplacement du robot

modélisation d'un MCC

Equation électrique : $u(t)=e(t)+R.i(t)+L.\frac{di(t)}{dt}$

Equation mécanique : $J.\frac{dw(t)}{dt}=Cem(t)-Cr(t)-f.w(t)$

Equation électromagnétique :

$$\begin{cases} e(t)=k.w(tt) \\ Cem=K.i(t) \end{cases}$$

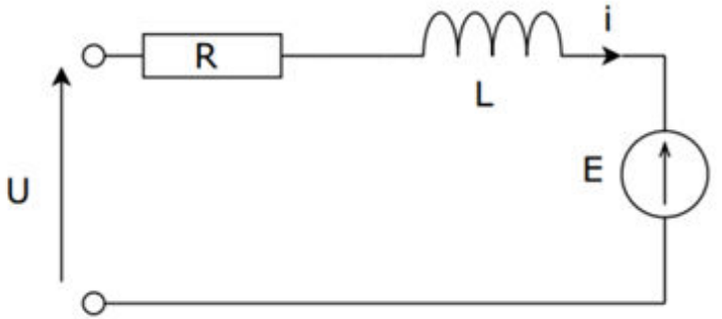


Figure 4 : Schéma électrique équivalent du MCC

Transformée de Laplace:

$$\begin{cases} U(p)=E(p)+R.I(p)+L.p.I(p) \\ E(p)=k.\ \Omega(p) \\ Cem(p)=k.I(t) \\ J.p.\Omega(p)=Cem(p)-Cr(p)-f.\Omega(p) \end{cases}$$

Objectif 2: Asservir la vitesse de déplacement du robot

modélisation d'un MCC

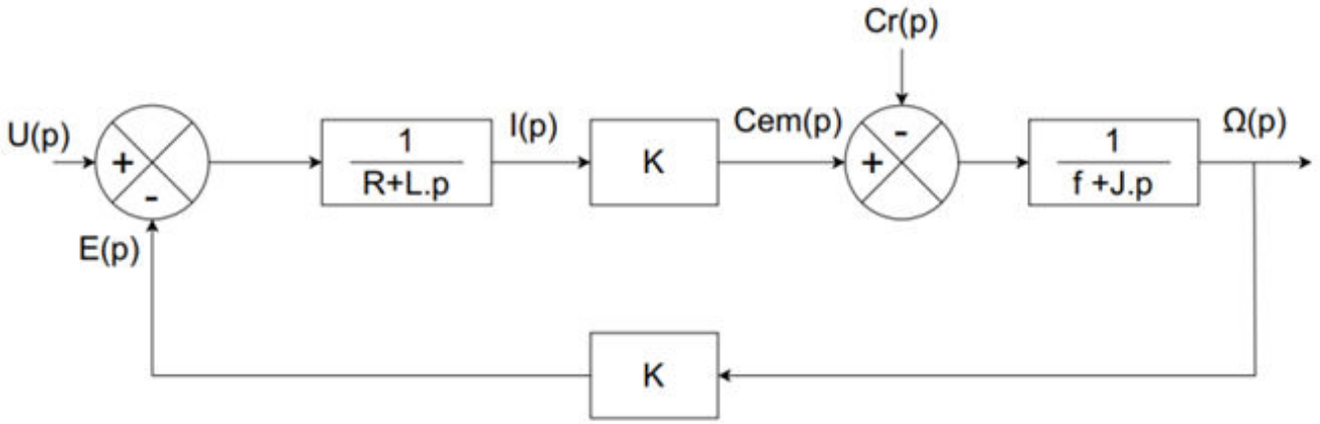


Figure 5 : Schéma bloc du MCC

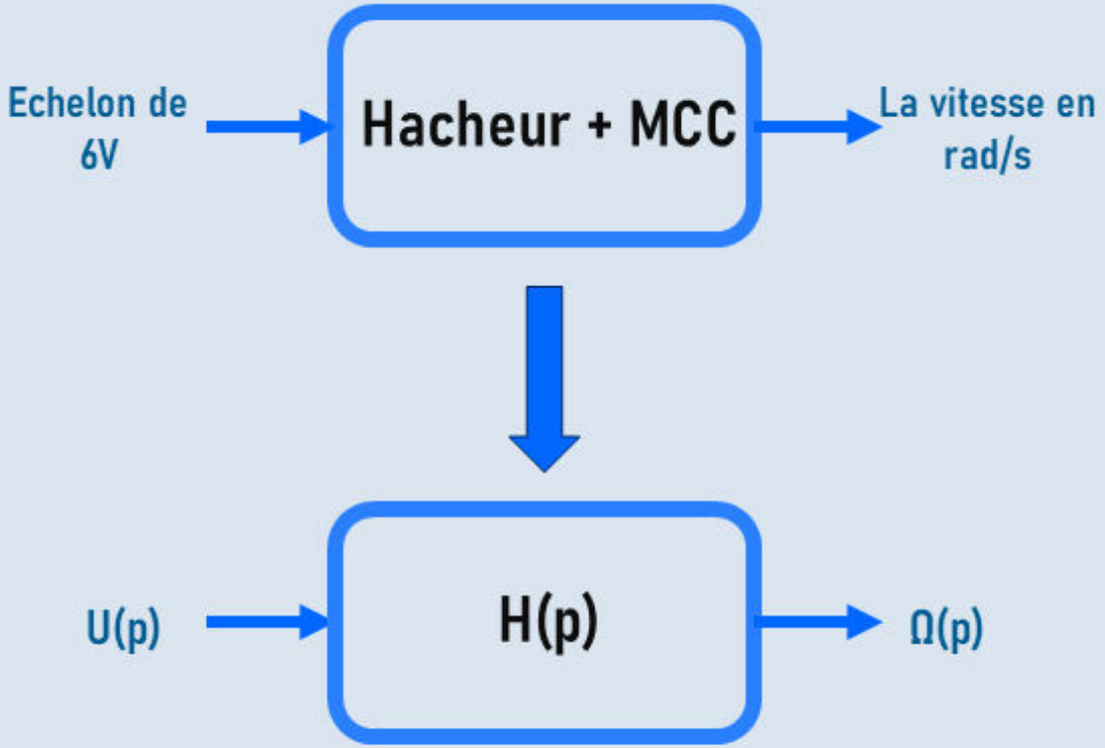
Pour faciliter la tâche, on suppose que :

- on néglige le couple résistant (Cr=0)
- on néglige l'influence de l'inductance d'induit

Objectif 2: Asservir la vitesse de déplacement du robot

Acquisition de la vitesse

Modèle :



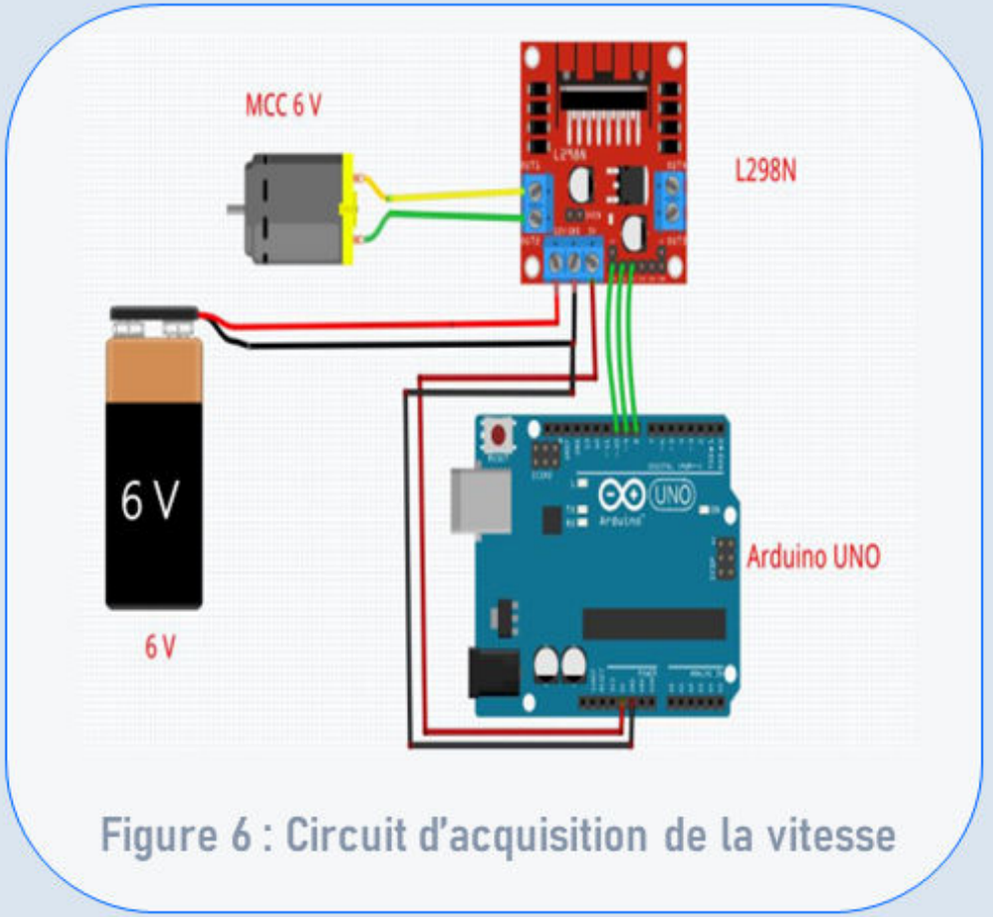


Figure 6 : Circuit d'acquisition de la vitesse

Objectif 2: Asservir la vitesse de déplacement du robot

Identification de système {Hacheur+MCC}

En utilisant la bibliothèque MATPLOTLIB et on trace la courbe figure :

Le système {Hacheur+MCC} est assimilable à

un système de premier ordre : $H(p)=\frac{K}{1+\tau.p}$

- U=6V
- $\Omega(\infty)\approx$17,5 ras/s
- K=2,91 rad/s.V
- τ = 0,3 s

D'ou $H(p)=\frac{2,91}{1+0,3.p}$

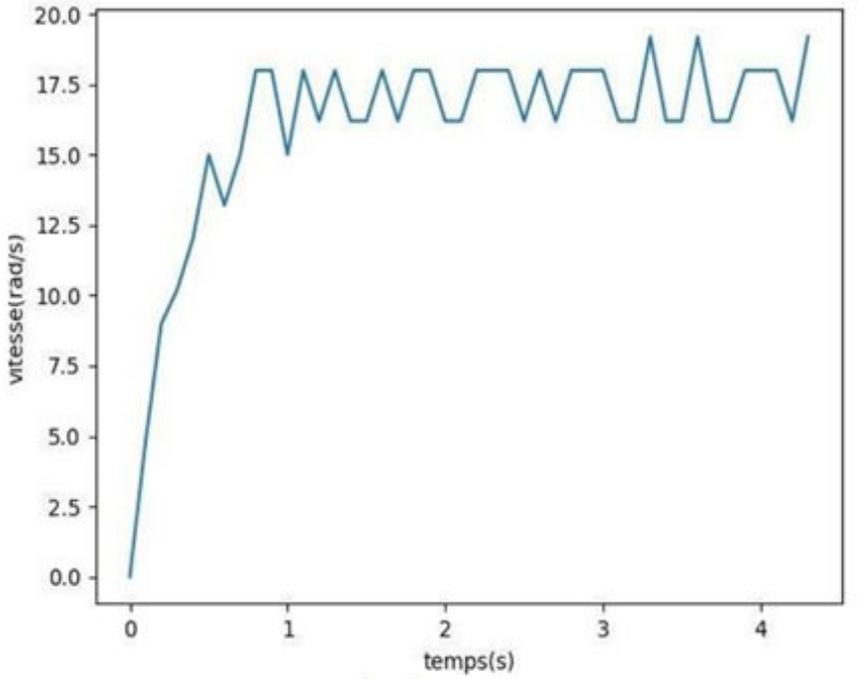


Figure 7 : Le tracé réel de la vitesse rotation en fonction du temps

Objectif 2: Asservir la vitesse de déplacement du robot

Identification de système {Hacheur+MCC}

Simulation du modèle approximé par MATLOPLOTLIB (python)

$$\Omega(t) = 17{,}5.(1 - e^{-t/0{,}3})$$

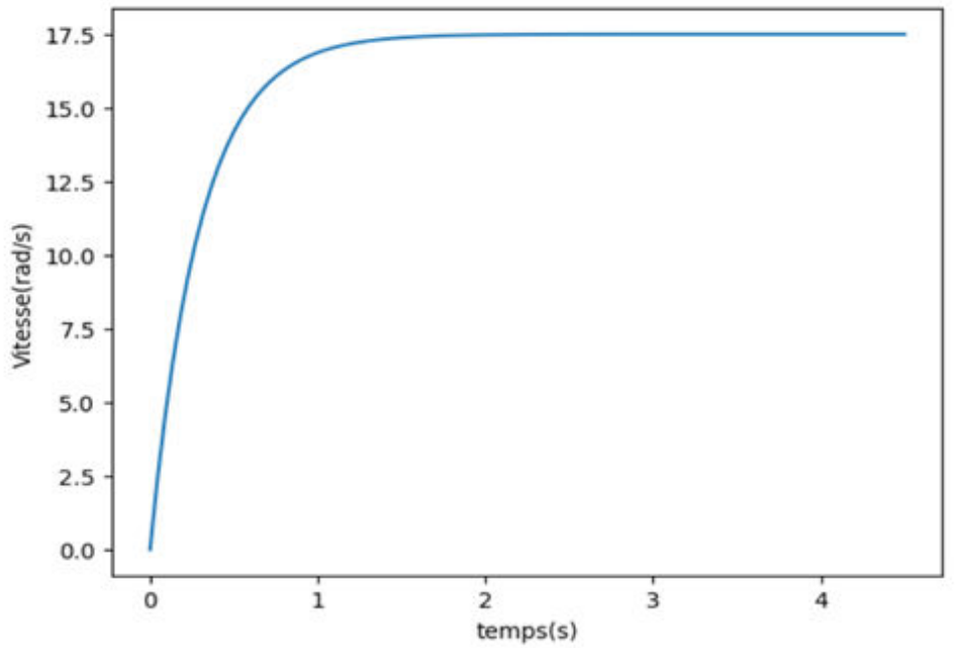


Figure 8 : La courbe approximé de vitesse de rotation en rad/s

Simulation du modèle approximé par Xcos (scilab)

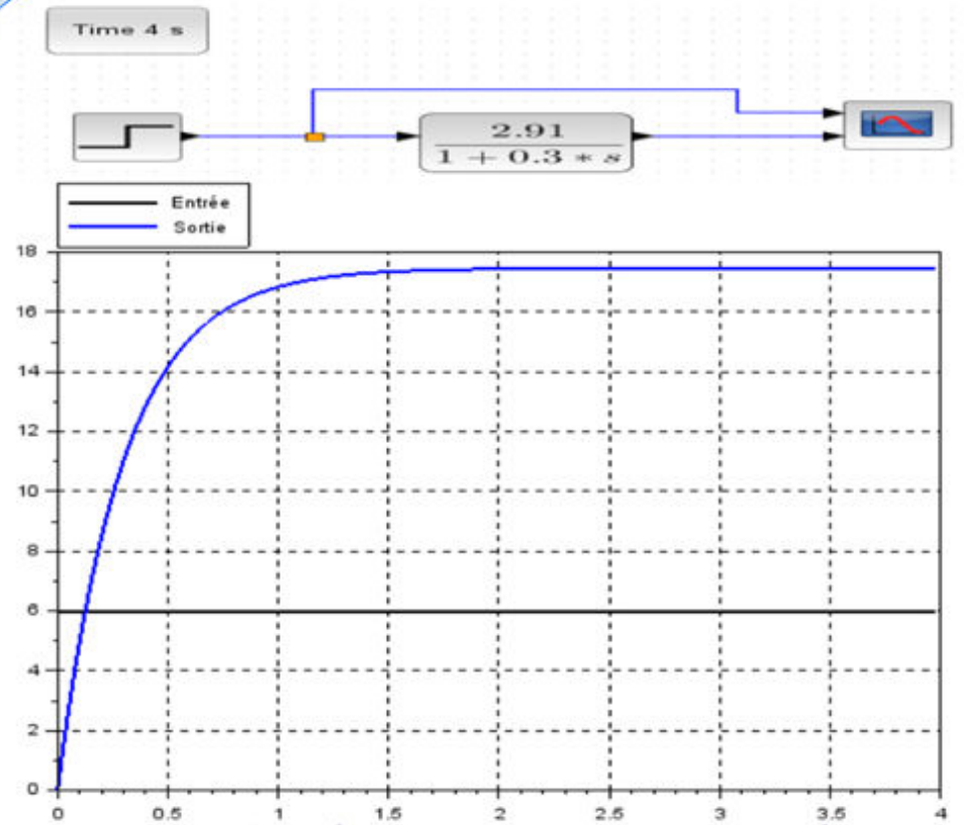


Figure 9 : Réponse temporelle du système

Asservissement de la vitesse

Système asservie sans correction

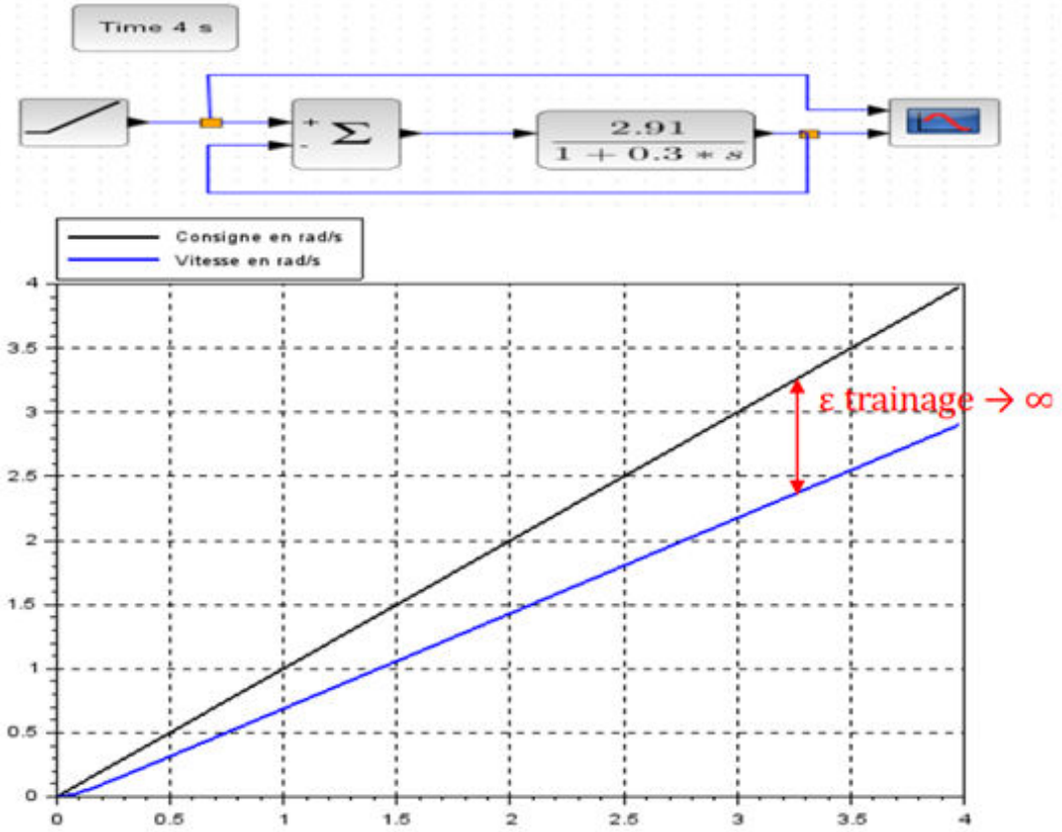


Figure 10 : Réponse à une rampe de 1V

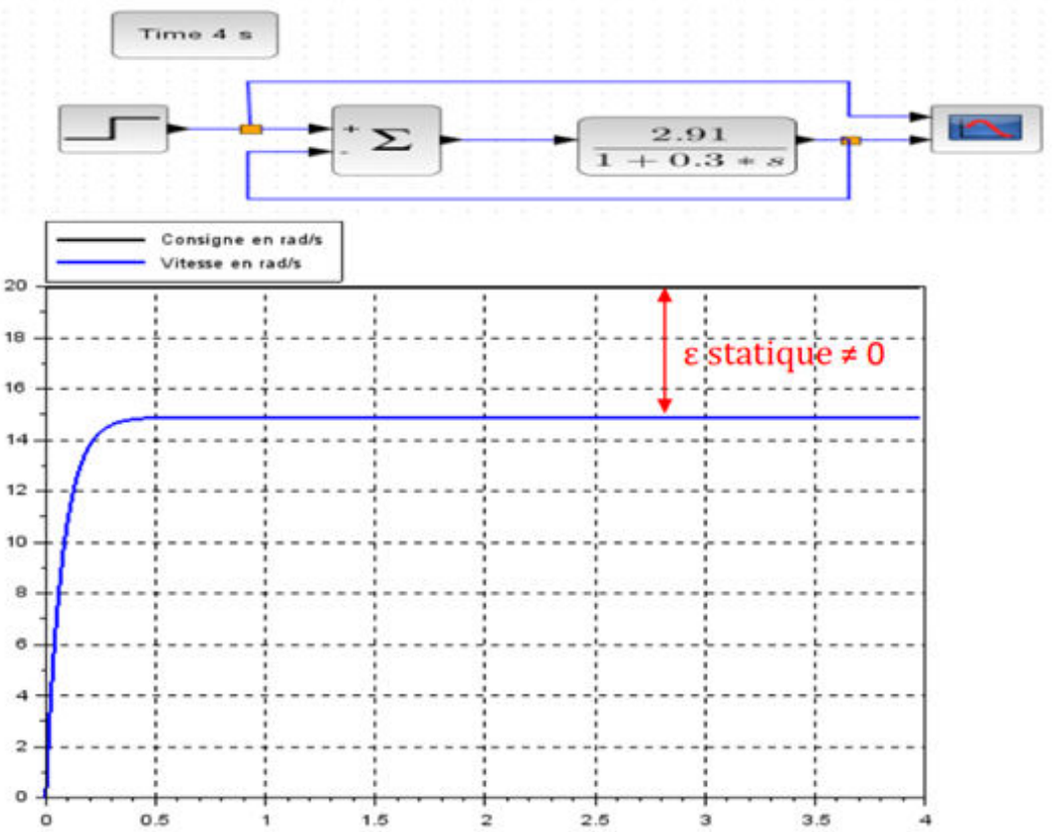


Figure 11 : Réponse à un échelon de 6V

Objectif 2: Asservir la vitesse de déplacement du robot

Asservissement de la vitesse

D'où la nécessite d'une correction($\varepsilon \neq 0$) :

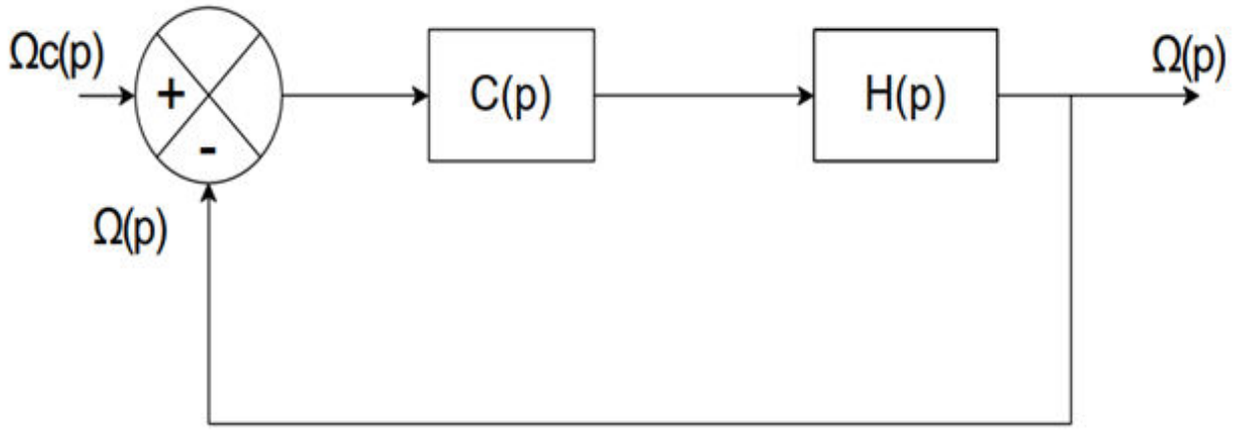


Figure 12 : Le schéma bloc de la boucle de la

Paramètre du correcteur PI

$$C(p)=Kp+\frac{Ki}{p}$$

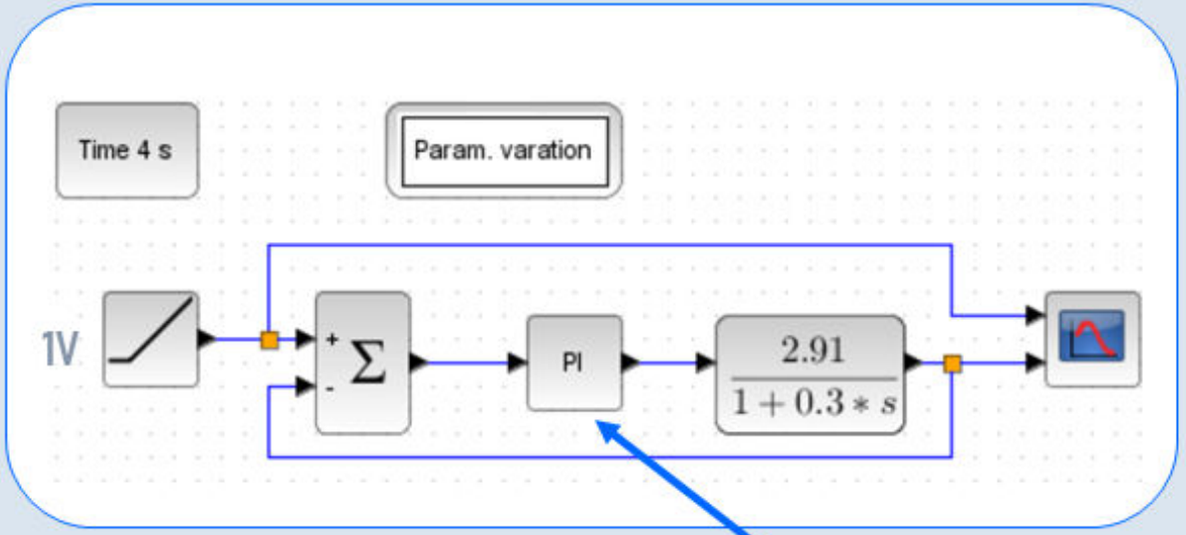
Exigence	Critère	niveau
Id=1.3.1.1 Asservissement de la vitesse	Temps de réponse à 5% prés	< 0,8 s
	Dépassement	0 %
	Erreur statique vis a vis de la consigne	Nulles
	Erreur trainage vis a vis de la consigne	< 8 %

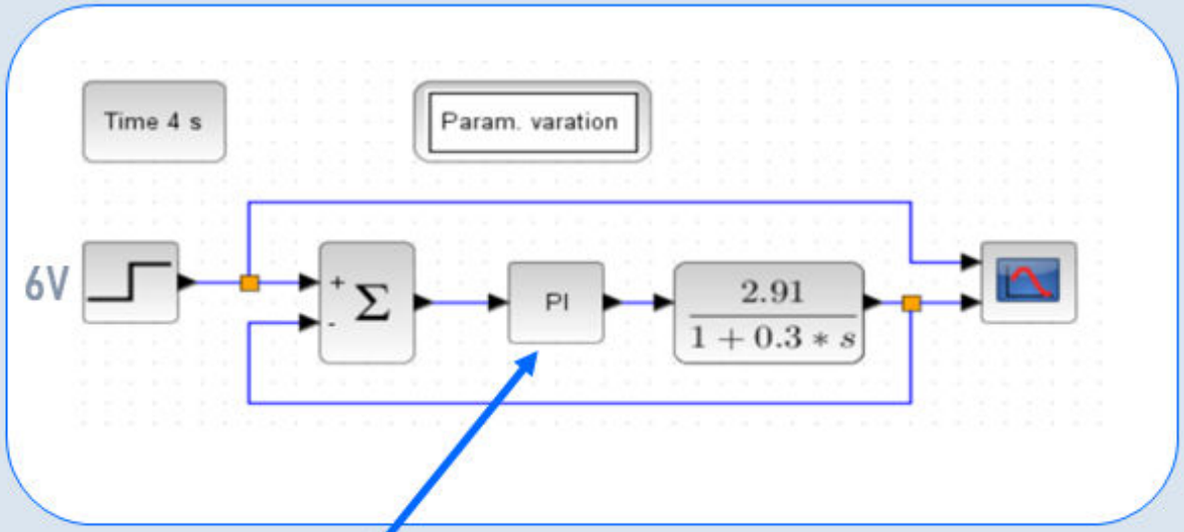
Figure 13:Extrait du cahier de charges

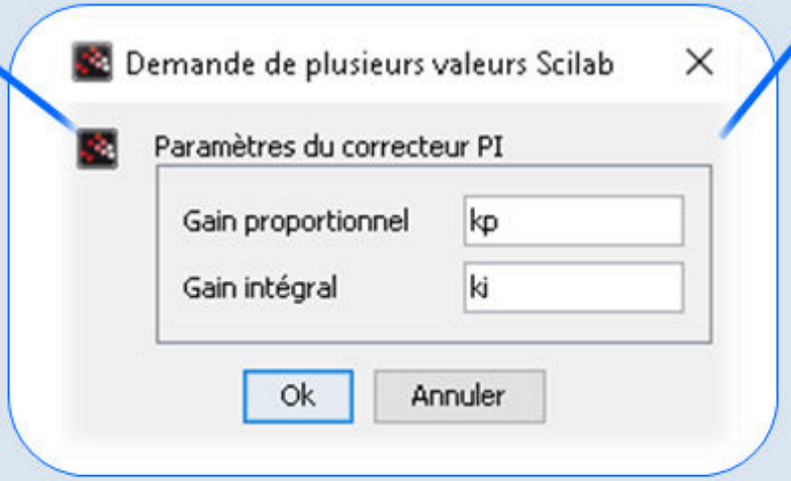
Objectif 2: Asservir la vitesse de déplacement du robot

Asservissement de la vitesse

Boucle de vitesse pour des entrées échelon et rampe sur Scilab :



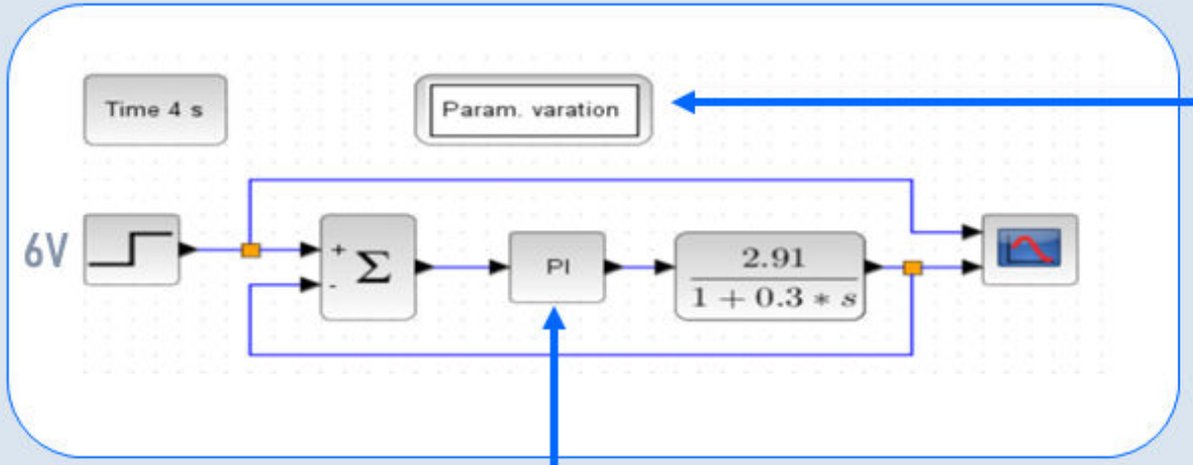


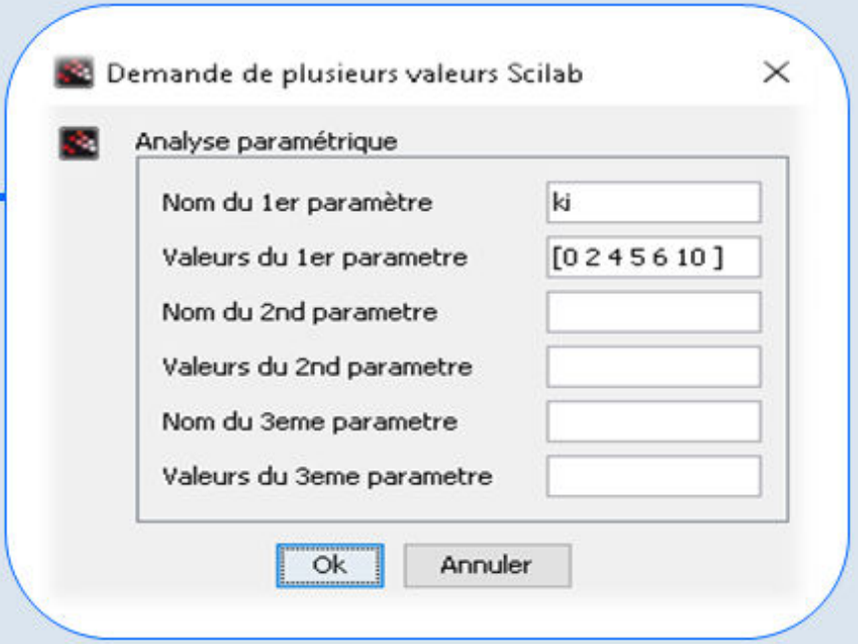


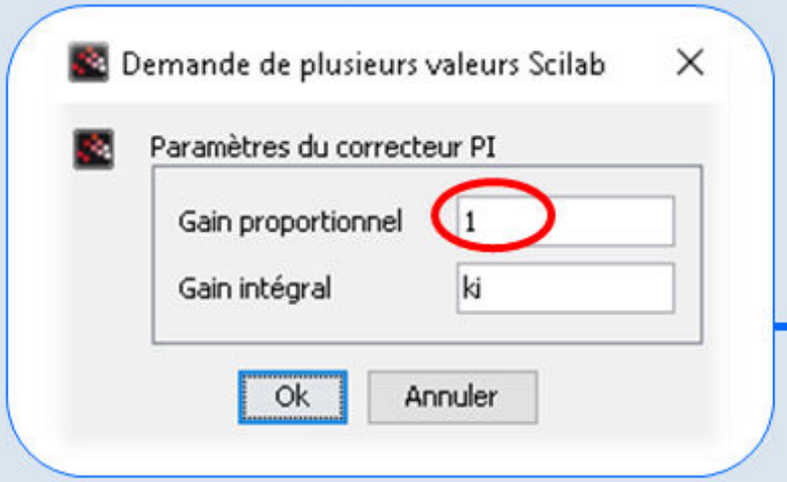
Objectif 2: Asservir la vitesse de déplacement du robot

Asservissement de la vitesse

L'entrée est un échelon de 6V:







On prend Kp=1 , et on cherche la valeur du Ki convenable

Objectif 2: Asservir la vitesse de déplacement du robot

Asservissement de la vitesse

Détermination du Ki convenable

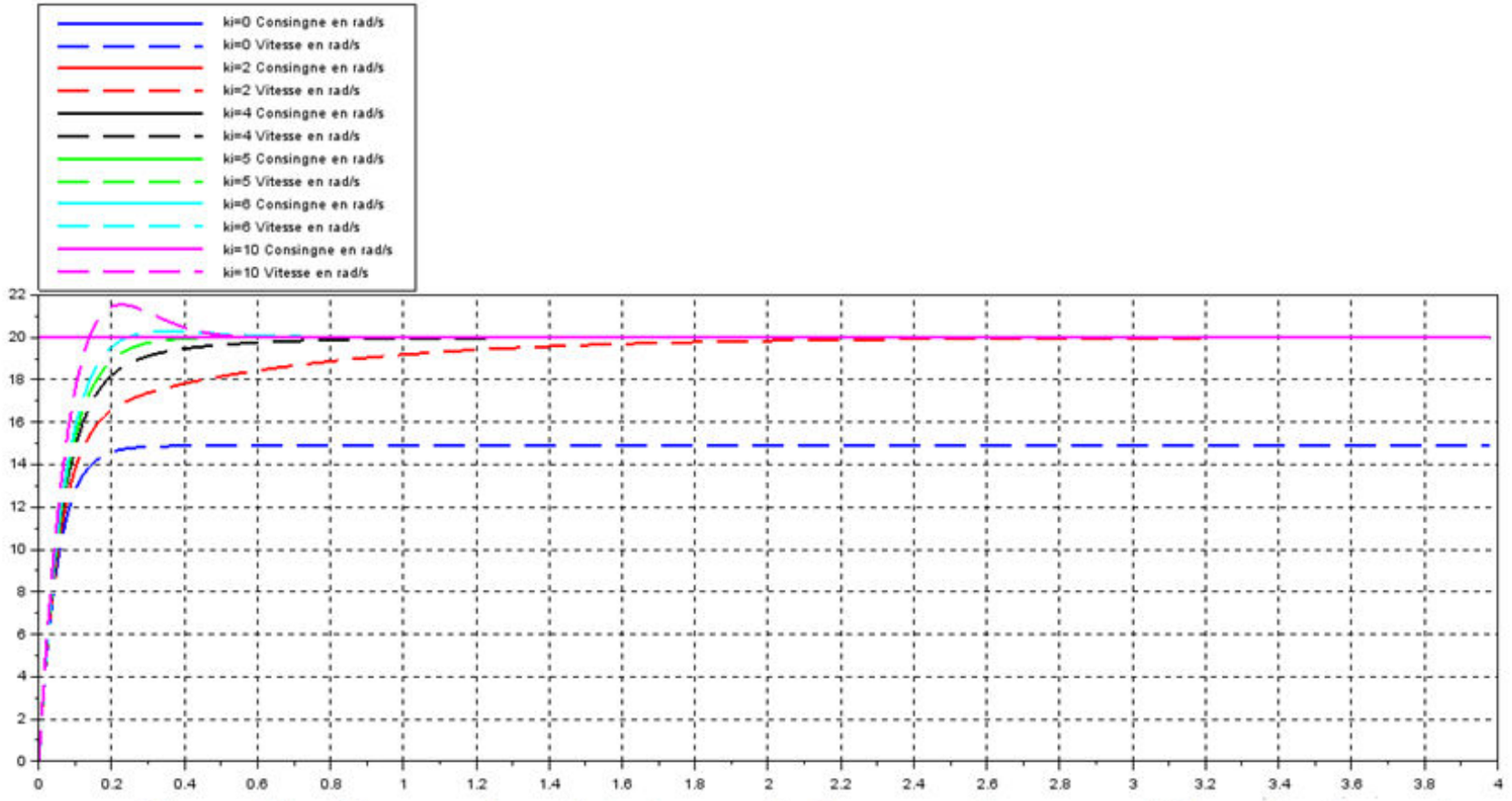


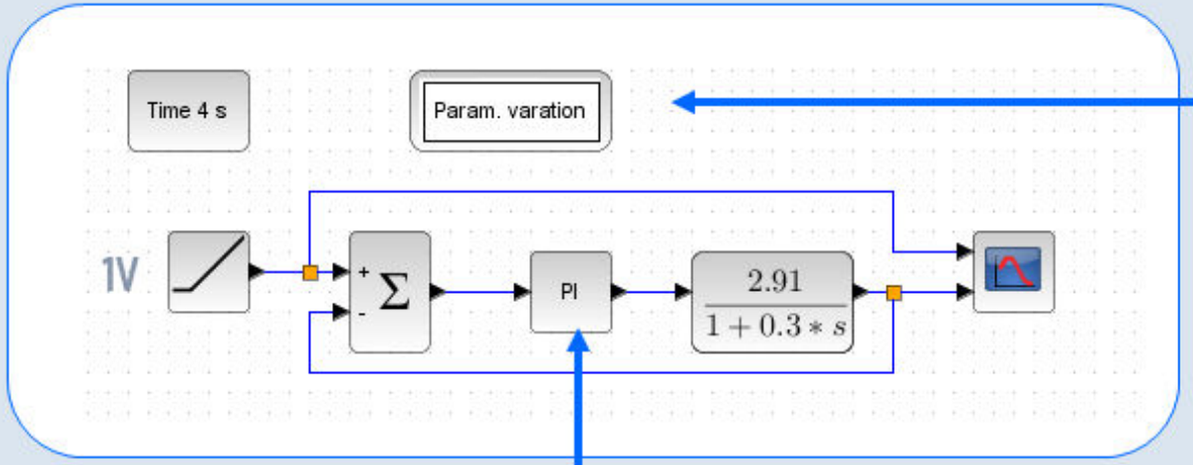
Figure 14 : Réponse à un échelon après la correction pour différent valeurs de Ki

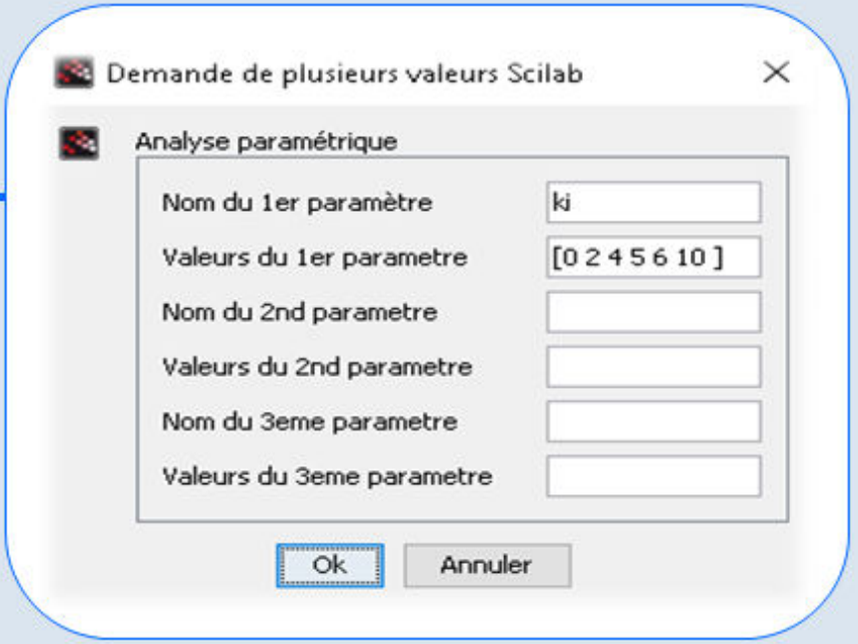
Ki=5

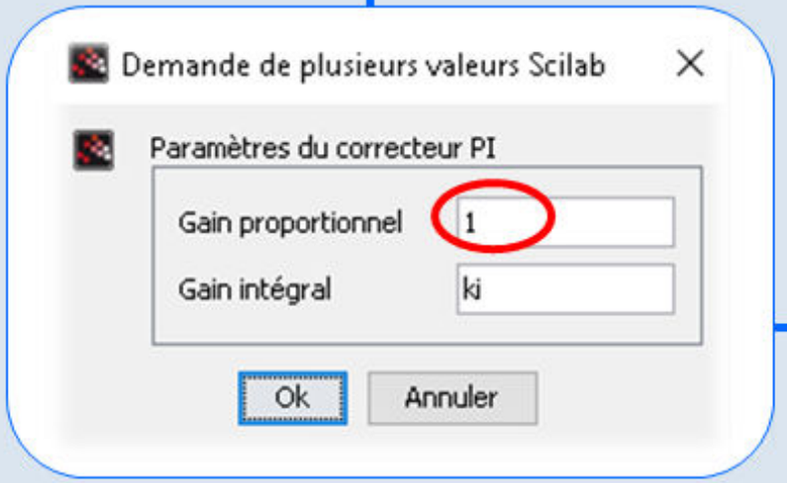
Objectif 2: Asservir la vitesse de déplacement du robot

Asservissement de la vitesse

L'entrée est une rampe de 1V:







On prend Kp=1 , et on cherche la valeur du Ki convenable

Objectif 2: Asservir la vitesse de déplacement du robot

Asservissement de la vitesse

Détermination du Ki convenable

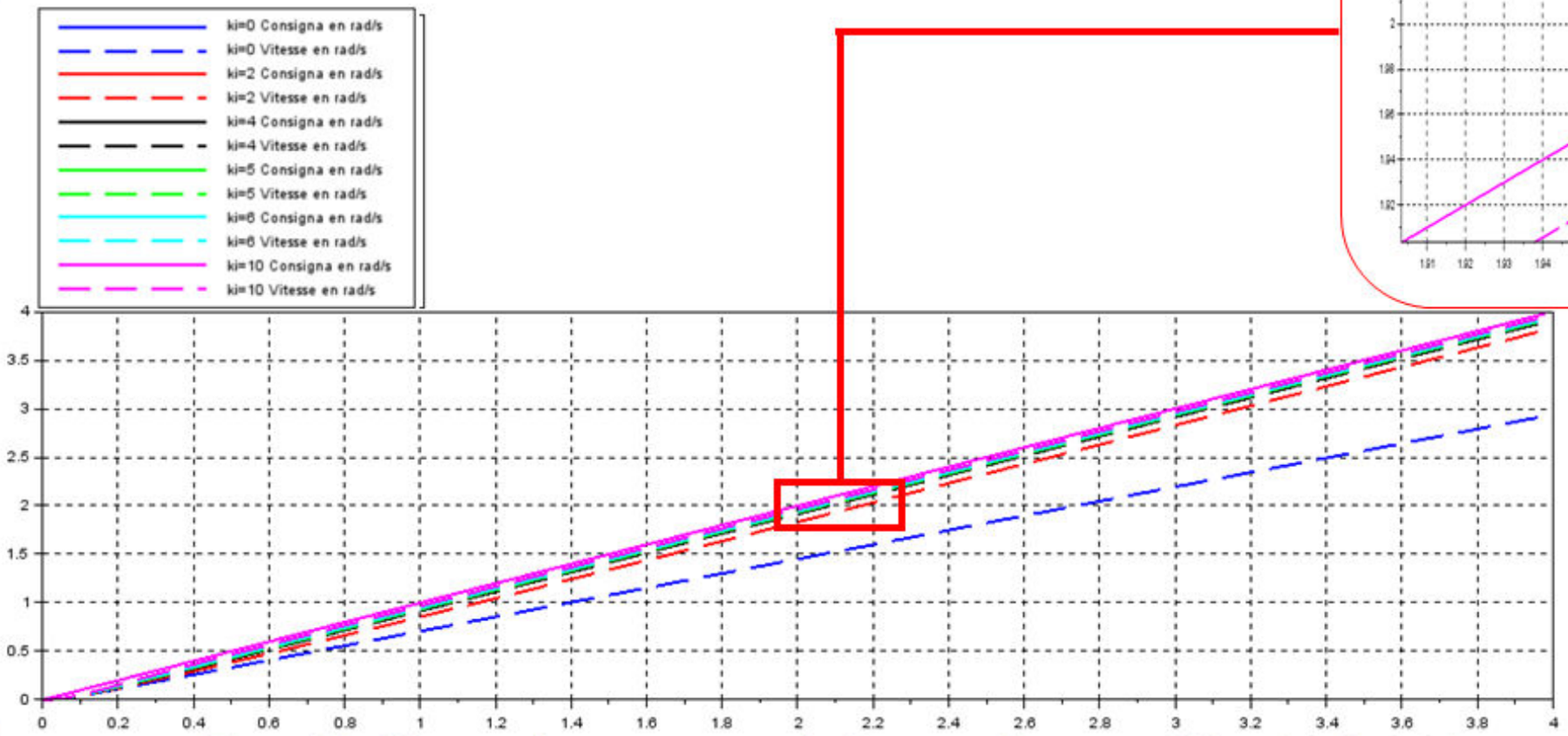


Figure 15 : Réponse à une rampe après la correction pour différent valeurs de Ki

Ki=5

Objectif 2: Asservir la vitesse de déplacement du robot

Asservissement de la vitesse

Conclusion

Le correcteur choisi

$$C(p)=1+\frac{5}{p}$$

EXIGENCE SATISFAITE :

Critère	Valeur	*Cahier des charges*
t5%	0,2 s	✓ respecté
Dn%	0%	✓ respecté
ε statique	0%	✓ respecté
ε trainage	7%	✓ respecté

Objectif 3:

Comparaison, Choix et positionnement du capteur d'obstacle

Objectif 3: Choix et positionnement du capteur d'obstacle

Comparaison et choix du capteur d'obstacle

Capteur Ultrason HC-SR04

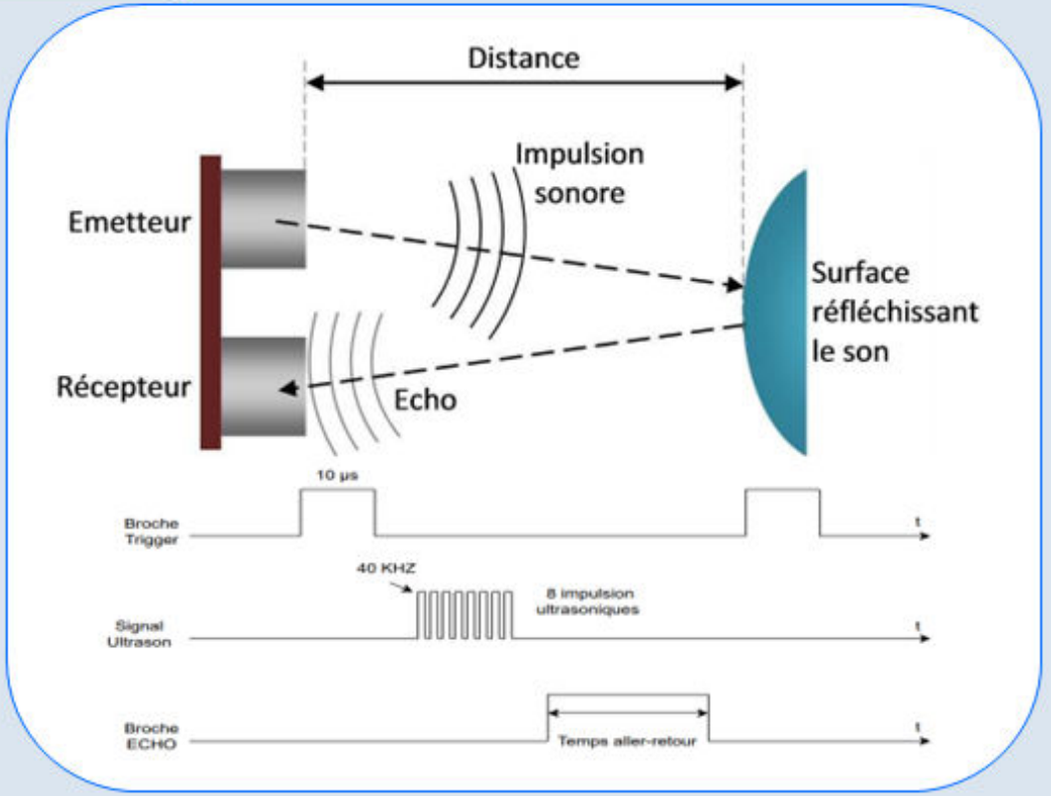


Figure 16 : principe de fonctionnement de capteur Ultrason

Pour capteur Ultrason

$$\text{Distance} = \frac{\text{Vitesse de son}}{2} \Delta t$$

Capteur infrarouge IR

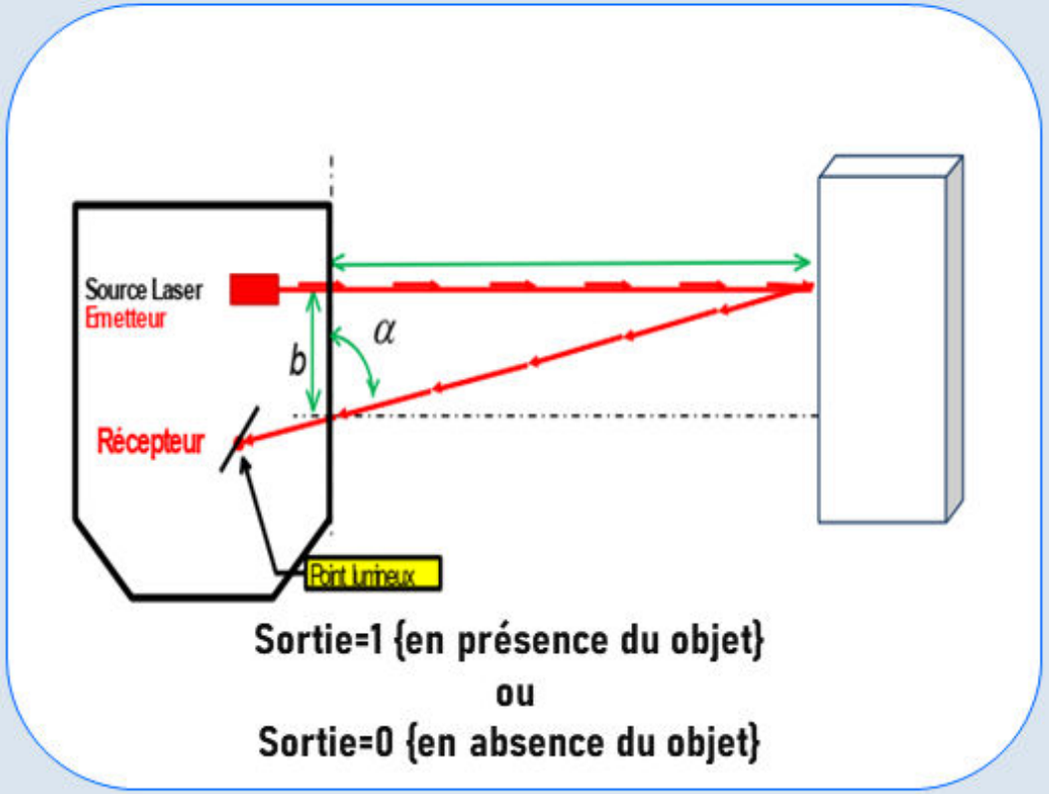


Figure 17: principe de fonctionnement de capteur Infrarouge

Pour capteur Infrarouge

$$\text{Distance} = b * \tan(\alpha)$$

Comparaison et choix du capteur d'obstacle

Capteur Ultrason HC-SR04

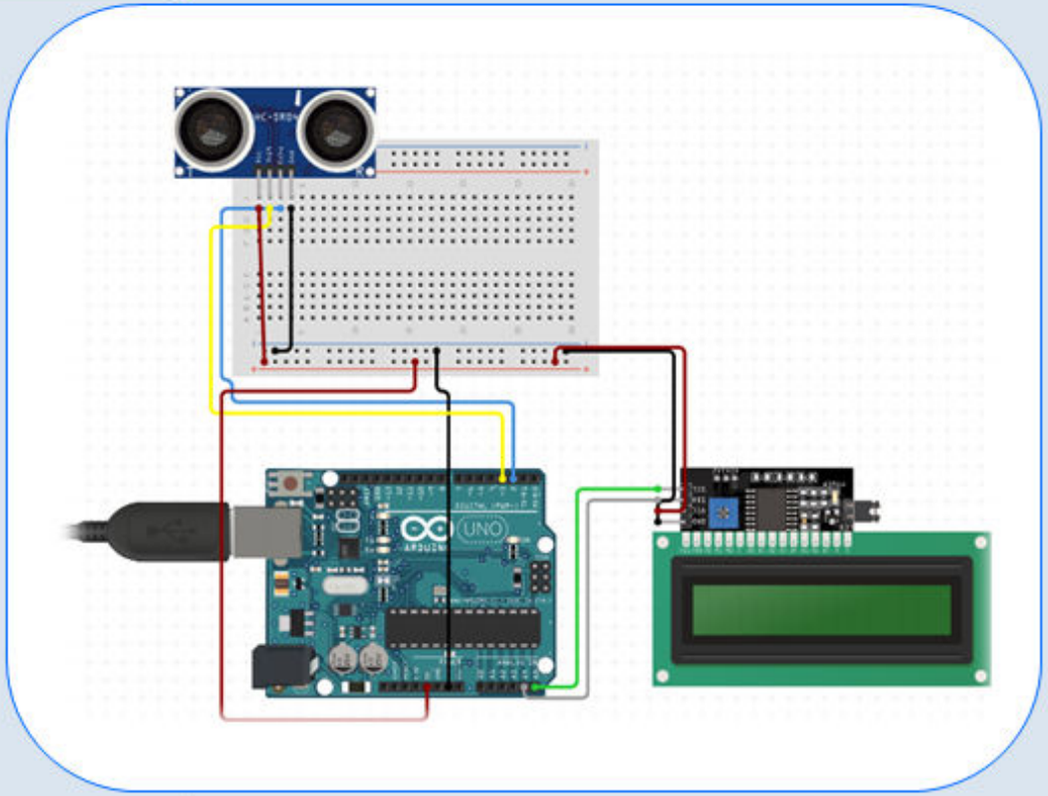


Figure 18 : câblage du capteur Ultrason sur « Circuito.io »

Capteur infrarouge IR

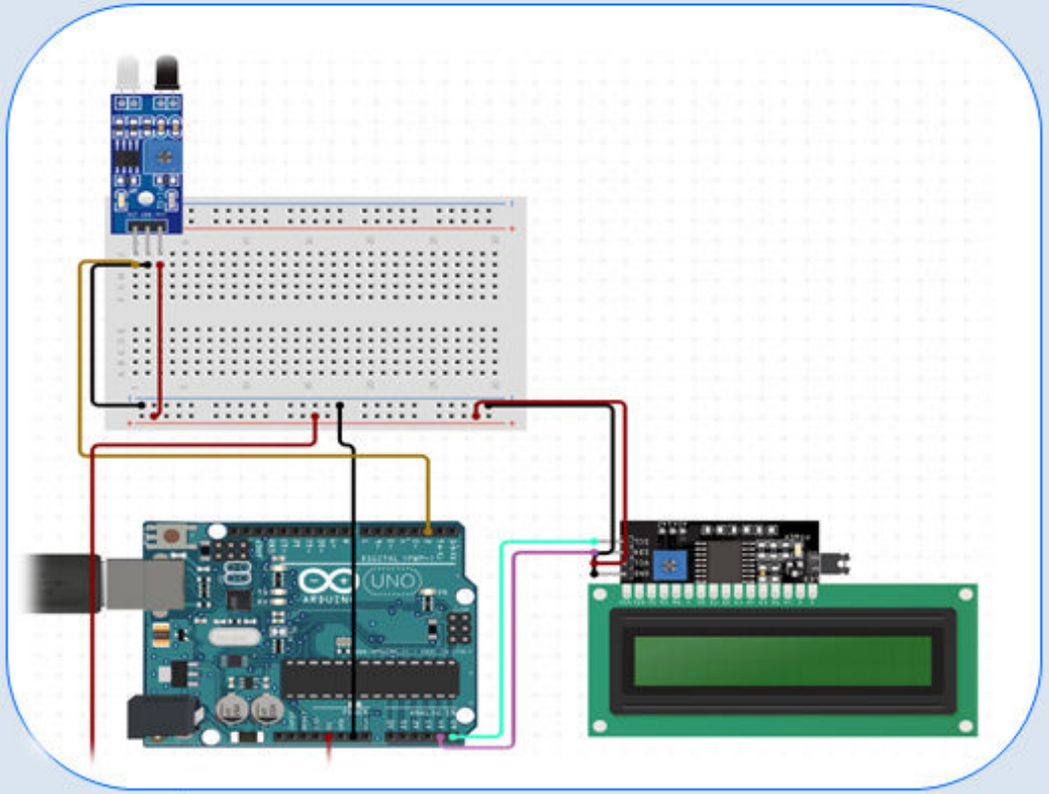


Figure 19 : câblage du capteur Infrarouge sur « Circuito.io »

Comparaison et choix du capteur d'obstacle

Capteur Ultrason HC-SR04

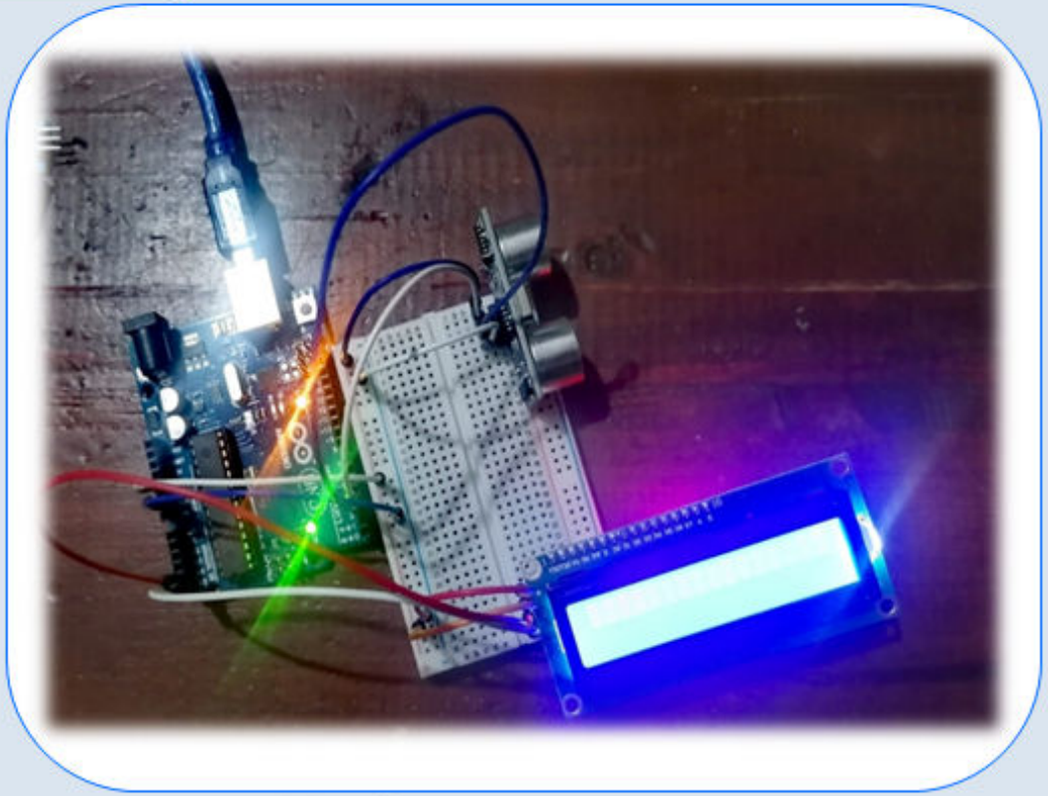


Figure 20 : câblage du capteur Ultrason

Capteur infrarouge IR

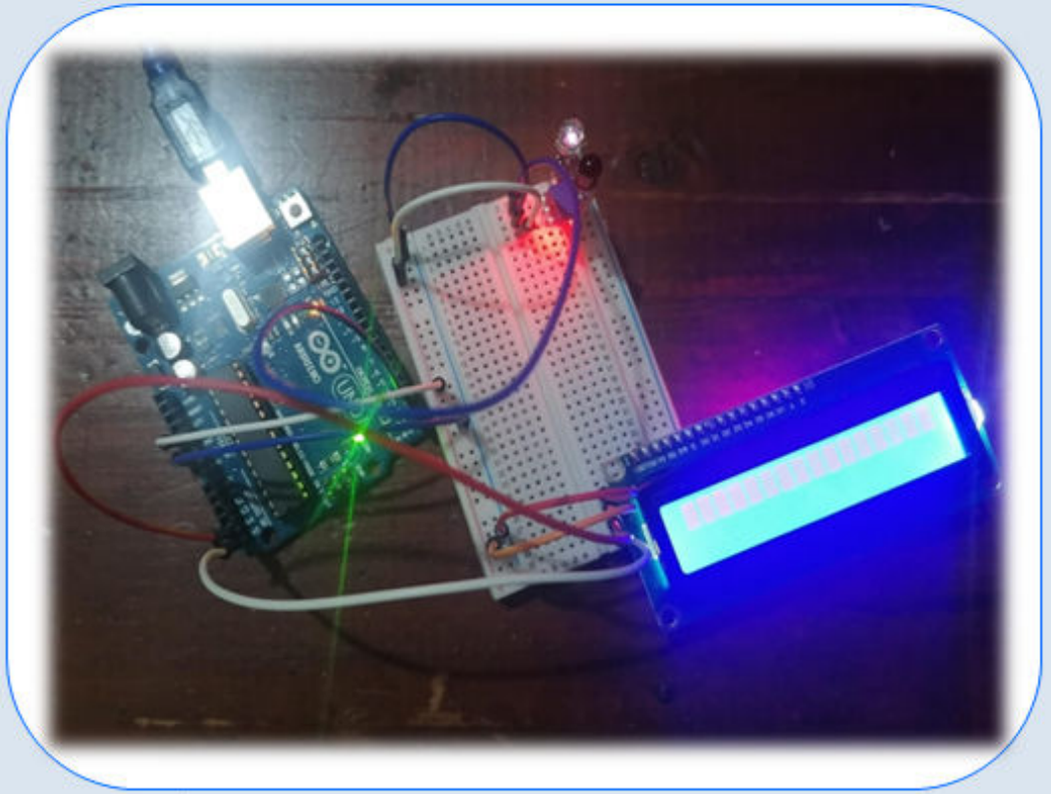
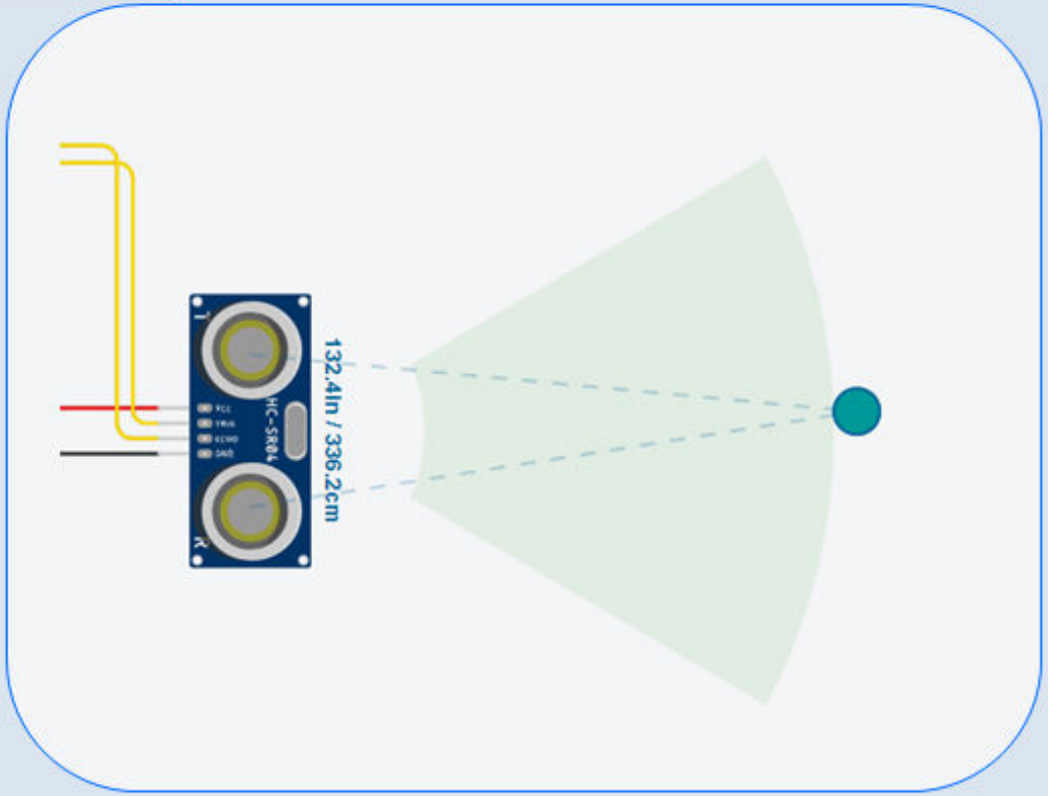


Figure 21 : câblage du capteur Infrarouge sur « tinkercart »

Objectif 3: Choix et positionnement du capteur d'obstacle

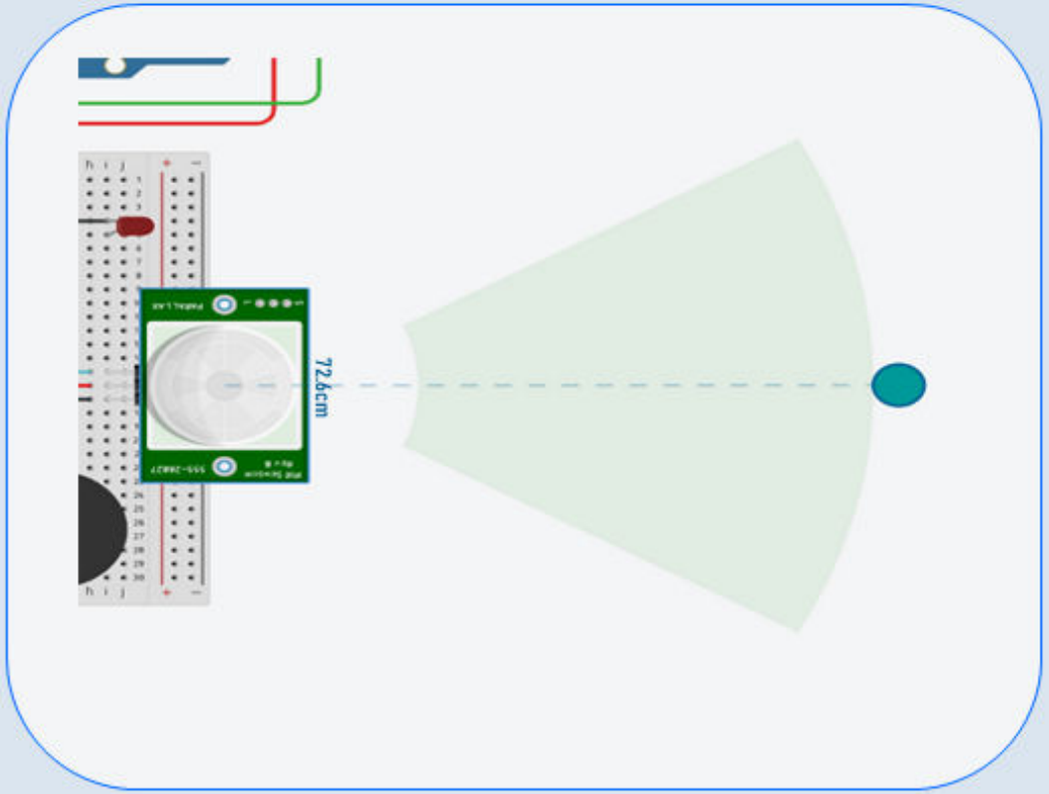
Comparaison et choix du capteur d'obstacle

Capteur Ultrason HC-SR04



$Dmax_{Ultason}$=336.6cm

Capteur infrarouge IR



$Dmax_{Infrarouge}$=72.6cm

Comparaison et choix du capteur d'obstacle

	Capteur Ultrason HC-SR04	Capteur infrarouge IR
Directivité	Cône d'environ 15°	Cône d'environ 5°
Précision	Relativement précis mais la précision diminue avec la distance, l'angle de mesure et les conditions de température et de pression.	Relativement précis mais la précision diminue avec la distance.
Coût	Peu chers	Peu chers
Sensibilité aux interférences	Sensible à la température et à la pression. Egalement sensible aux autres robots utilisant la même fréquence ce qui peut poser problème dans une compétition.	Sont sensibles aux fortes sources de lumière qui contiennent un fort rayonnement infrarouge.Sont également sensibles à la couleur et à la nature des obstacles.

Comparaison et choix du capteur d'obstacle

Conclusion

Puis que

$Dmax_{Ultason} > Dmax_{Infrarouge}$

Capteur ultrason est le plus performant .

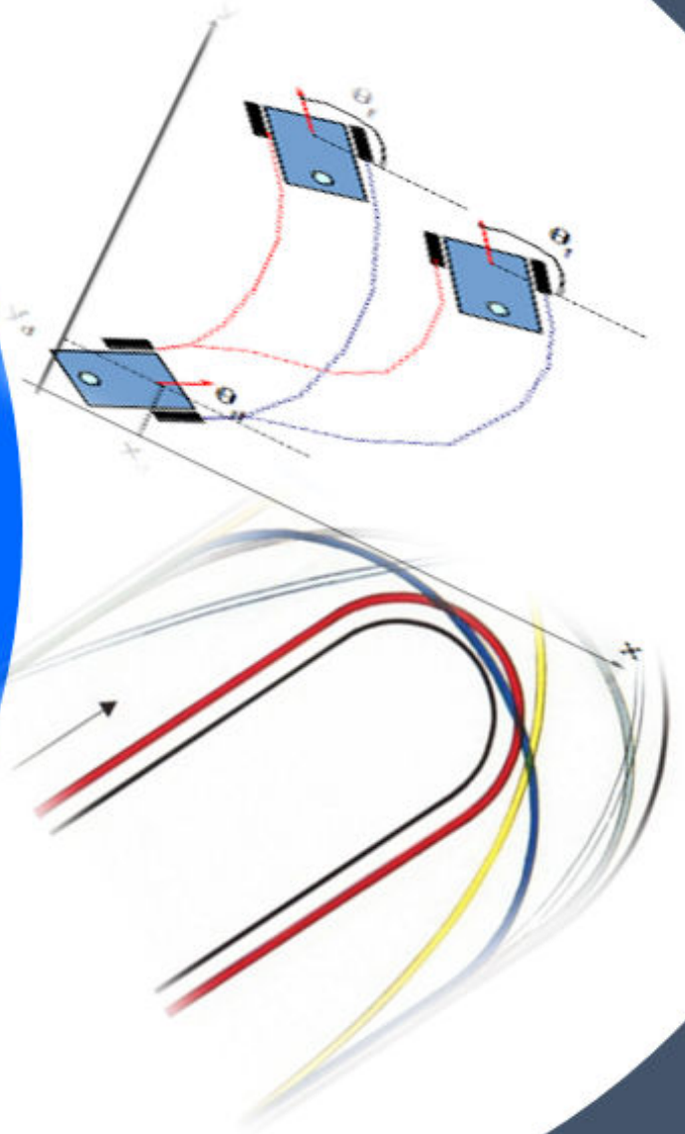
Figure 22 : la position de capteur Ultrason

Caractéristiques de capteur choisi :

Tension d'alimentation	5V DC
Courant d'alimentation	15mA
Fréquence de travail	40Hz
Distance maximale de détection	4m
Distance minimale de détection	2cm
Angle de détection	15 degrés
Signal d'entrée de l'émetteur	Impulsion à l'état haut de 10µs
Signal de sortie du récepteur	Signal numérique à l'état haut et la distance proportionnellement
Dimension	45*20*15mm

Objectif 4:

Gestion de la trajectoire du robot



Gestion de la trajectoire du robot

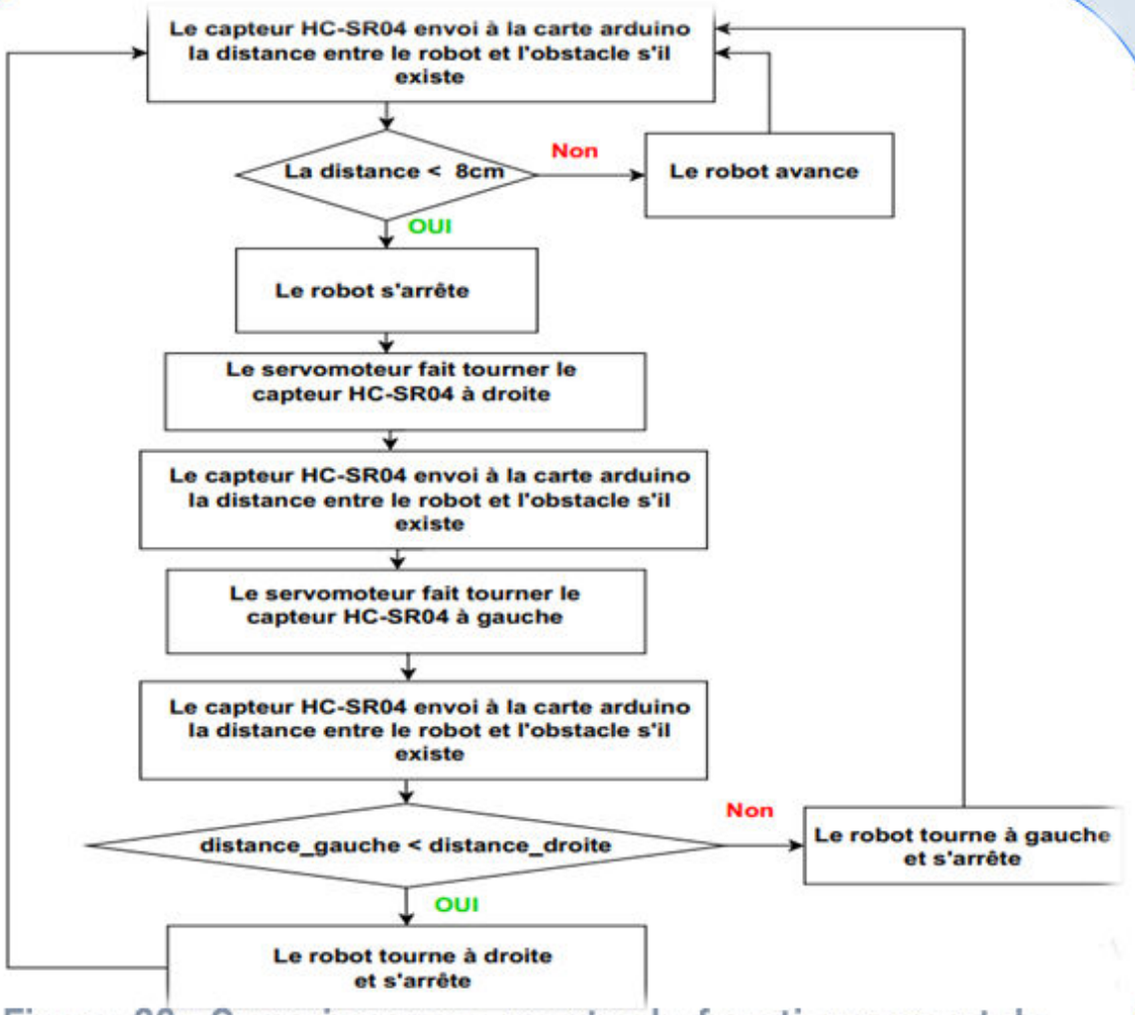


Figure 23 : Organigramme montre le fonctionnement du robot

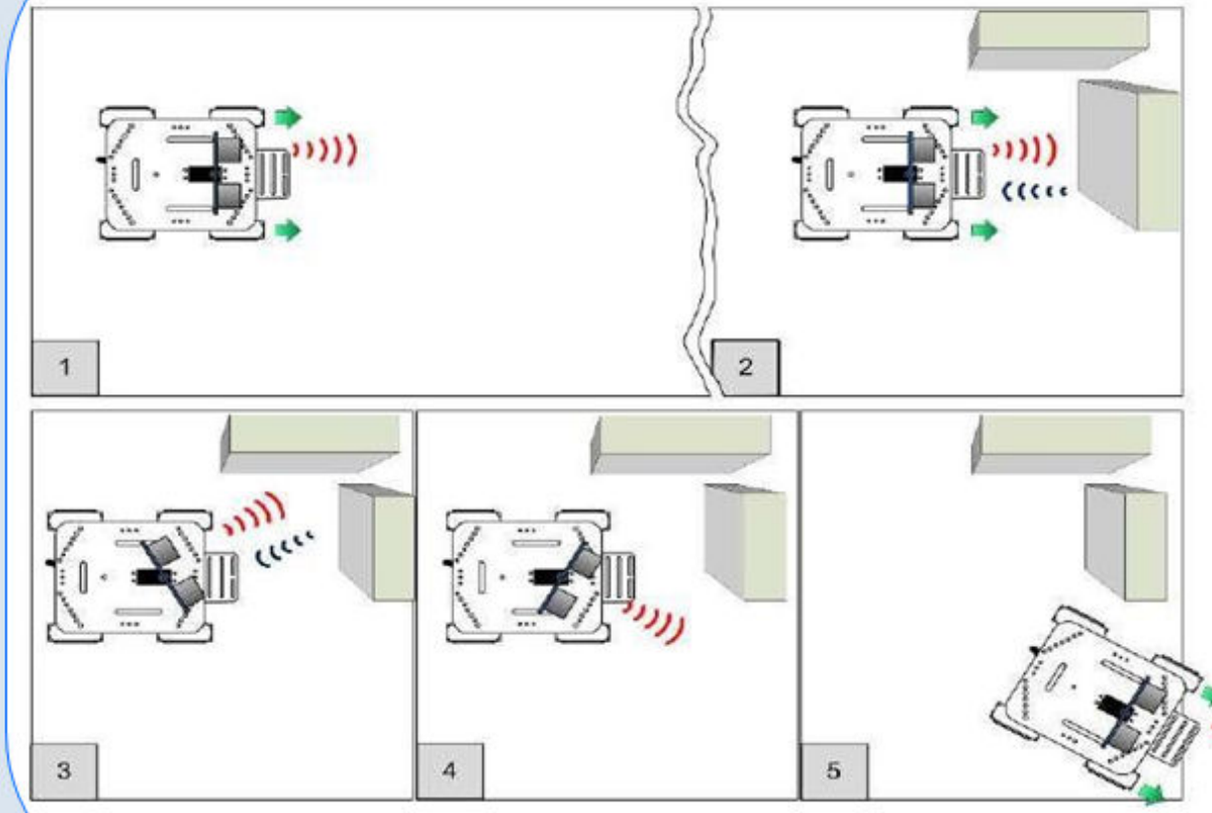


Figure 24 :images montre le fonctionnement du robot

Gestion de la trajectoire du robot

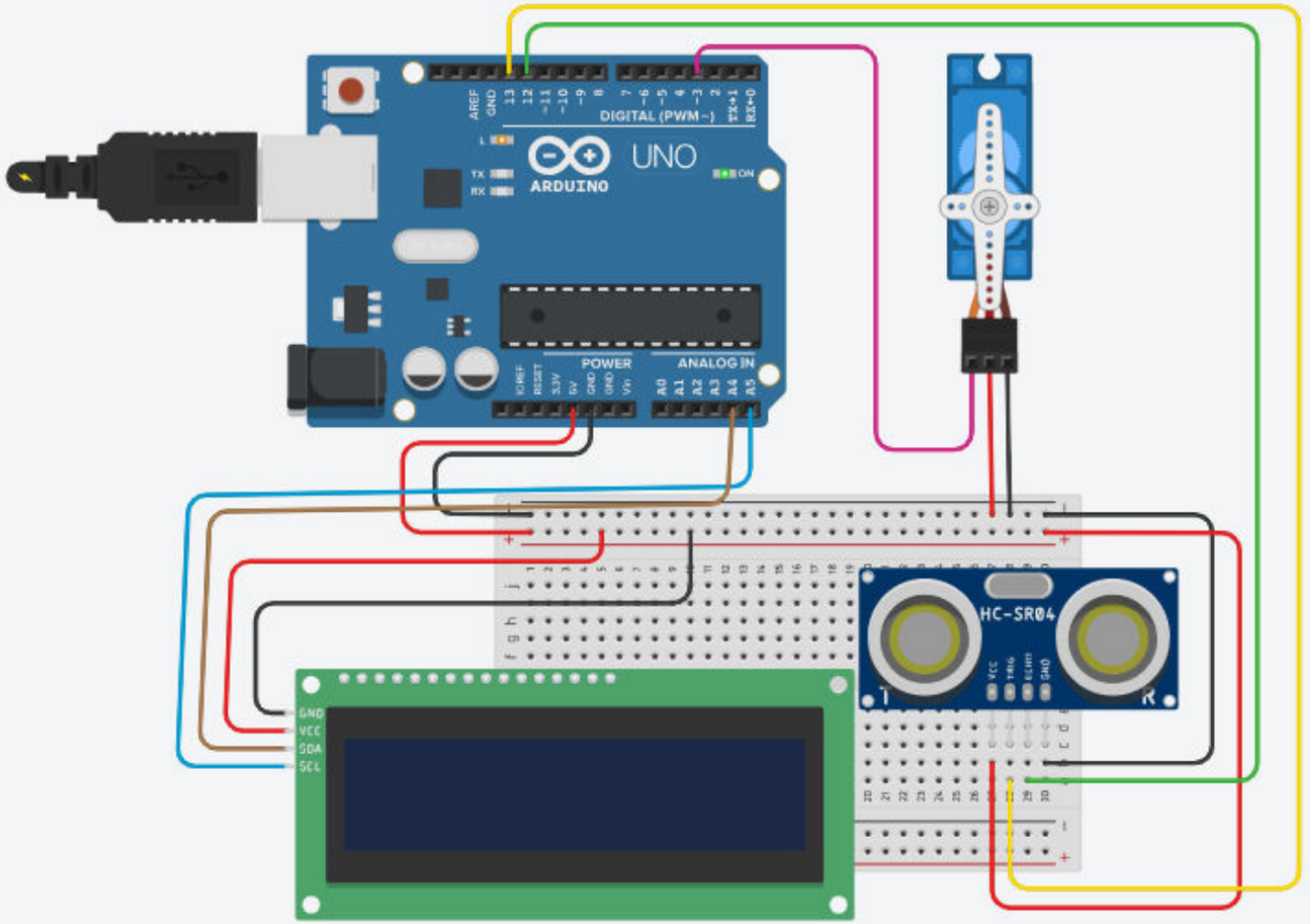


Figure 25 :Branchement du capteur ultrason avec un servomoteur sur "tinkercad"

Gestion de la trajectoire du robot

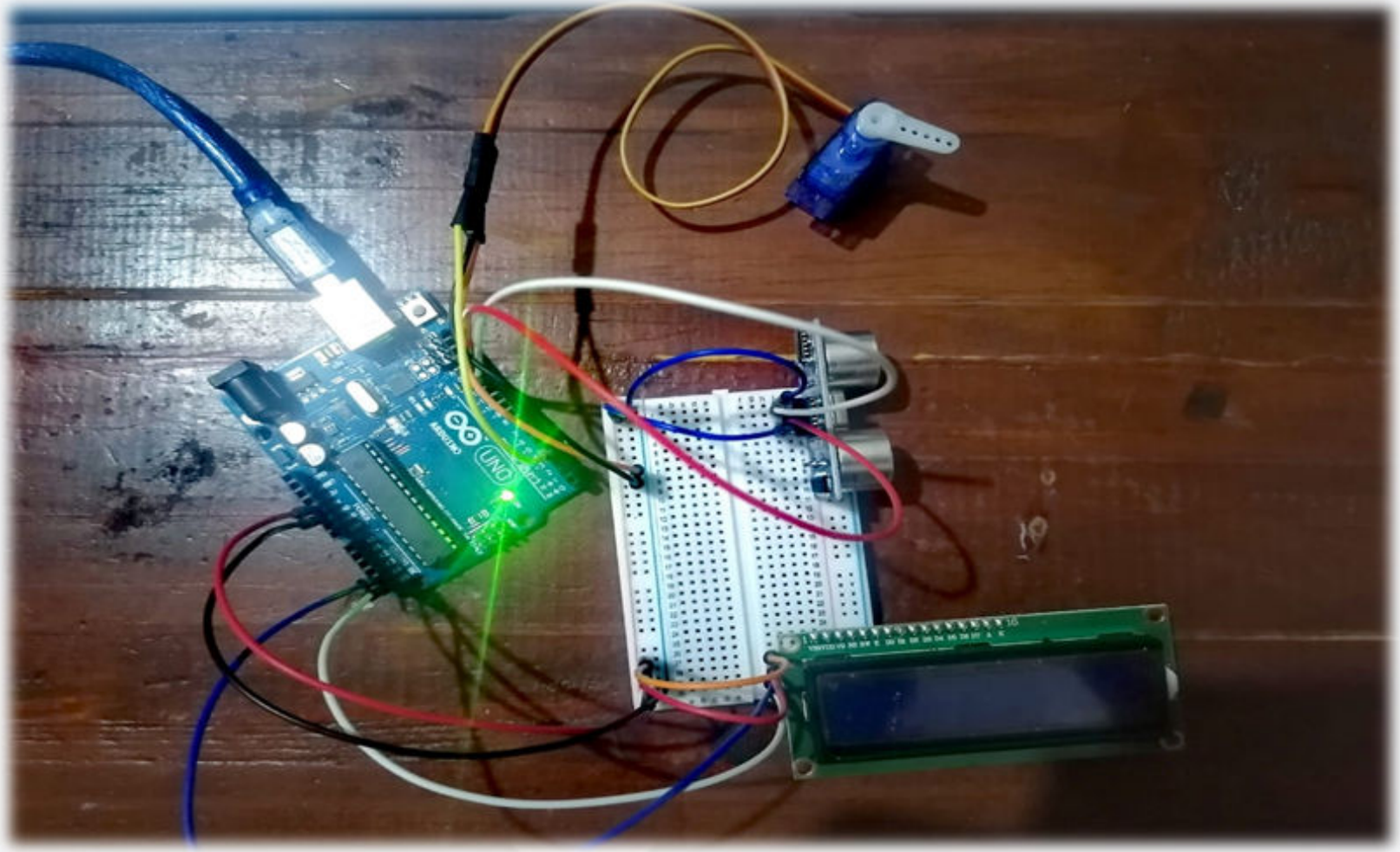


Figure 26 :Branchement du capteur ultrason avec un servomoteur

Gestion de la trajectoire du robot

Conclusion

✓ les exigences liées à la sécurité sont satisfaites .

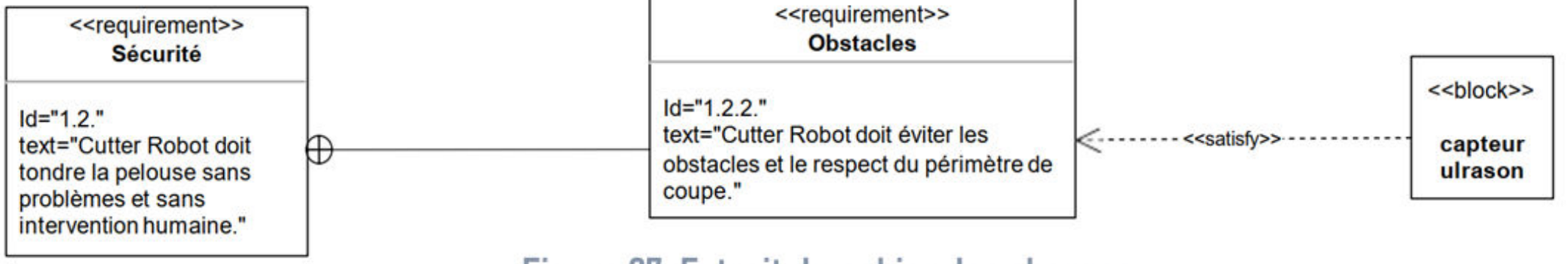


Figure 27 :Extrait du cahier des charges

Objectif 5:

Dimensionnement du panneau photovoltaïque

Consommation maximale

Composant	Quantité	Courant maximale (mA)	Tension(V)	Puissance absorbé (W)
Hc-SR04	1	15	5	0,075
Servomoteur	1	250	5	1,25
Arduino UNO	1	50	5	0,25
Hacheur	2	36	5	0,36
MCC	5	400	6	12
			Ubatt = 12V	Pt = 13,935 W

Figure 28 : Composant du robot

Objectif 5: Dimensionnement du panneau photovoltaïque

Choix d'un panneau solaire

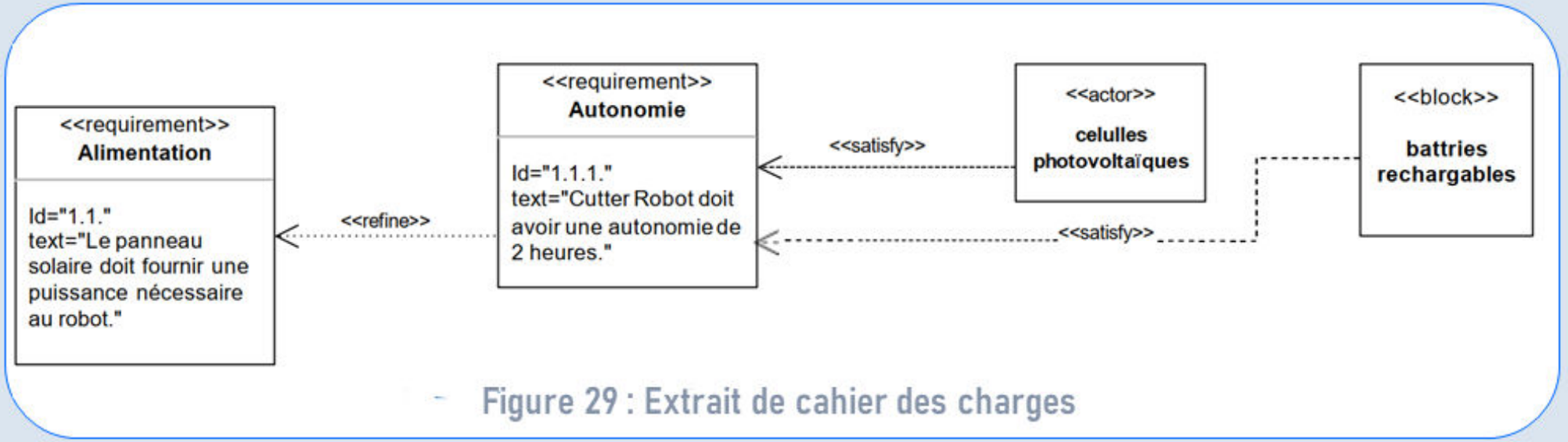


Figure 29 : Extrait de cahier des charges

Calcule La puissance fournie par le panneau solaire :

Énergie consommé par jour : Ec = Pt * durée d'utilisation par jour

AN: **Ec=27.87 Wh**

Rayonnement solaire moyen par jour : **Δt = 5 h**

La puissance fournie par le panneau solaire:

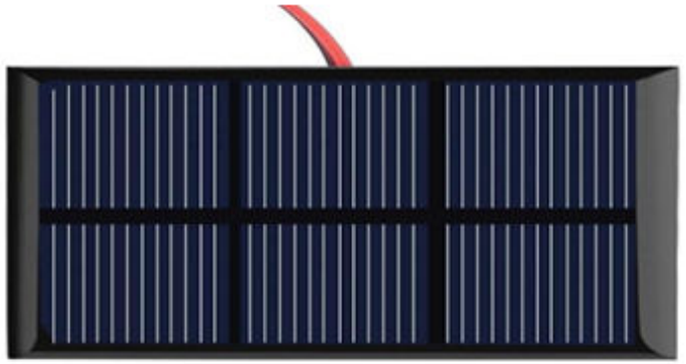
$P \geq 5{,}574$ W

Objectif 5: Dimensionnement du panneau photovoltaïque

Choix d'un panneau et de la batterie

Caractéristiques du panneau :

TensionMAX	CourantMAX	Puissance de sortie
12 V	0,5A	6 W



Choix de la batterie :

- C : capacité d'une batterie (Ah)
- Ec : Énergie consommé par jour (Wh/j)
- N : Le nombre de jours nuageux
- D : Pourcentage de décharge de la batterie
- Ubatt :la tension de la batterie

La formule :

$$C = \frac{Ec * N}{Ubatt * D}$$

On prend : N = 1 et D =0,8 (une batterie de plomb)

AN: $C = 2,91$

Choix de la batterie

Caractéristiques de la batterie :

On choisi quatre batteries de caractéristique suivante :

La tension et **Ubatt = 6 V**

Capacité **C=1,5 Ah**

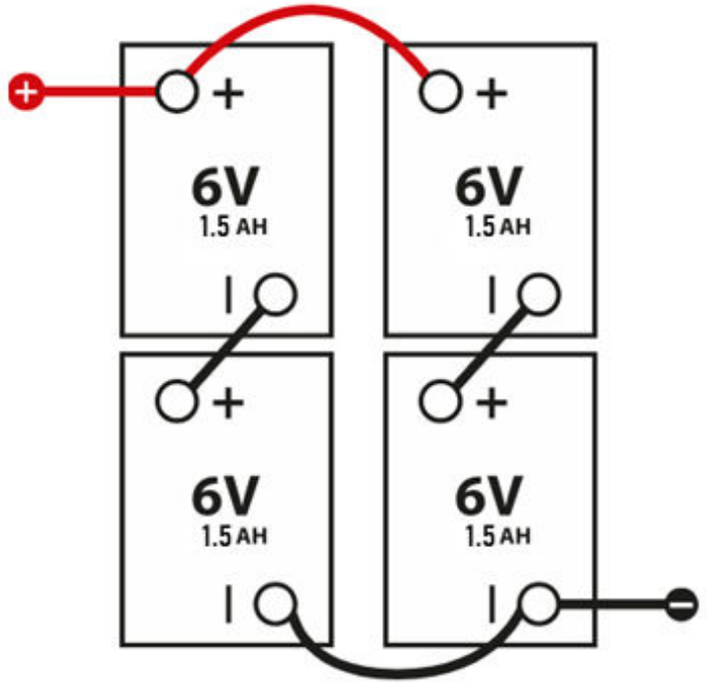


Figure 30: Le branchement des batteries

Conclusion :

✓ Les exigences liées à la sécurité sont satisfaites . Il faut ajouter un mécanisme (suiveur solaire) pour avoir une puissance maximale du panneau.

Merci pour votre attention

ANNEXES

```
#define in1 9
#define in2 8
#define enA 10

void setup() {
 pinMode(in1,OUTPUT);
 pinMode(in2,OUTPUT);
 pinMode(enA,OUTPUT);

 digitalWrite(in1,HIGH);
 digitalWrite(in2,LOW);
 analogWrite(enA,255); // Le moteur tourne à une vitesse maximale en rad/s
 delay(2000);

//Arrêt du moteur
 digitalWrite(in1,LOW);
 digitalWrite(in2,LOW);
}
```

Code de variation la vitesse du moteur par L298N

```
import numpy as np
def f(x) :
    return 17.5*(1-np.exp(-(1/0.3)*x))
```

```
import matplotlib.pyplot as plt
x=np.linspace(0,4.5,100)
plt.plot(x,f(x))
plt.xlabel("temps(s)")
plt.ylabel("Vitesse(rad/s)")
pltshow()
```

Tracer La courbe approximé de vitesse de rotation en (rad/s)

ANNEXES

```
#include <HCSR04.h>
#include <LiquidCrystal_I2C.h>

byte triggerPin = 13;
byte echoPin = 12;

LiquidCrystal_I2C lcd(0x27,20,4);

int led_ultrason = 2;
int led_infrarouge = 3;

int infrarouge_state = 4;

double max_distance = 0;

void detect(){

  double* distance = HCSR04.measureDistanceCm();

  if(distance[0] > max_distance){
    max_distance = distance[0];
  }
  if(distance[0] != -1){
    digitalWrite(led_ultrason, 1);
  }
  else{
    digitalWrite(led_ultrason, 0);
  }

    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Max distance :");
    lcd.setCursor(0, 1);
    lcd.print(max_distance);
    delay(100);

  if(digitalRead(infrarouge_state)){
    digitalWrite(led_infrarouge, 0);
  }
  else{
    digitalWrite(led_infrarouge, 1);
  }

}
```

Commande des capteurs : ultrason et infrarouge

ANNEXES

```
void setup () {
  Serial.begin(9600);
  HCSR04.begin(triggerPin, echoPin);

  lcd.init();
  lcd.backlight();

  pinMode(led_ultrason, OUTPUT);
  pinMode(led_infrarouge, OUTPUT);
}

void loop () {
  detect();

}
```

Commande des capteurs : ultrason et infrarouge

ANNEXES

```
#include <HCSR04.h>
#include <LiquidCrystal_I2C.h>
#include <Servo.h>

byte triggerPin = 13;
byte echoPin = 12;

LiquidCrystal_I2C lcd(0x27,20,4);

Servo servo;
int position = 0;

void detect(){

  double* distance = HCSR04.measureDistanceCm();
  if(distance[0] < 8){
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Obstacle detecte");
    servo.write(0); //135
    delay(2000);
    double* distance_1 = HCSR04.measureDistanceCm();
```

```
servo.write(180); //135
delay(2000);
Serial.print("Comparaison des deux distances : ");
Serial.print(distance_1[0]);
Serial.print("        ");
Serial.println(HCSR04.measureDistanceCm()[0]);
if(distance_1[0] < HCSR04.measureDistanceCm()[0]){
  Serial.println("a droite");
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Contournement");
  lcd.setCursor(0, 1);
  lcd.print("a droite...");
}
else {
  Serial.println("a gauche");
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Contournement");
  lcd.setCursor(0, 1);
  lcd.print("a gauche...");
}
```

Code de l'organigramme

ANNEXES

```
    Serial.println("\n");
    servo.write(90);
    delay(2000);
     lcd.clear();
  }
}

void setup () {
  Serial.begin(9600);
  HCSR04.begin(triggerPin, echoPin);

  lcd.init();
  lcd.backlight();

  servo.attach(3);
  servo.write(90);
}

void loop () {
  detect();

}
```

Code de l'organigramme