République **A**lgérienne **D**émocratique et **P**opulaire
Ministère de L'**E**nseignement **S**upérieur et de la
Recherche **S**cientifique

Université **M**ouloud **A**mmeri De **T**izi-**O**uzou

Faculté de **G**énie **E**lectrique et d'**I**nformatique
DEPARTEMENT D'AUTOMATIQUE

Mémoire de fin d'étude
de MASTER ACADEMIQUE
Spécialité : **Automatique et informatique industrielle**

Présenté par
Aghiles ABED
Hassina KACI MOUSSA

Mémoire dirigé par **Boussad IDJERI**

Thème

Conception et réalisation d'un système de régulation à base d'un microcontrôleur

Mémoire soutenu publiquement le 12 juillet 2018 devant le jury composé de :

M Mouhand-Outahar BENSIDHOUM	MCA, Tizi-Ouzou,	Président
Mme Ouiza BOUKENDORE	MAA, Tizi-Ouzou,	Examinatrice

Remerciements

Nous tenons tout d’abord à remercier Dieu le tout puissant et miséricordieux, qui nous a donné la force et la patience d’accomplir ce modeste travail, comme nous remercions nos très chers parents qui ont toujours été là pour nous.

En second lieu nous voudrons exprimer toute notre reconnaissance à notre promoteur monsieur B.IDJERI, nous le remercions de nous avoir encadrés, orientés, aidé et conseillés.

Nous tenons à remercier les membres de jury d’avoir accepté d’examiner ce modeste travail.

Et pour finir, nous remercions tout le personnel de l’établissement de Mouloud Mammeri et en particulier ceux de la faculté génie électrique, département automatique, comme nous tenons aussi à remercier toutes les personnes qui ont contribuées de près ou de loin à la réalisation de ce mémoire.

TABLE DES MATIERES

CHAPITRE I : Présentation des matériels

CHAPITRE II : Prototypage (Arduino / Matlab Simulink)

CHAPITRE III: Commande et régulation

CHAPITRE IV: Réalisation et simulation du système

Liste des figures

Liste des tableaux

Introduction générale

L'automatique est une science qui traite de la modélisation, de l'analyse,de l'identification et de la commande des systèmes dynamiques. Elle inclut la cybernétique et a pour fondements théoriques les mathématiques, la théorie du signal et l'informatique permettant de commander des systèmes en répondant à différents cahiers des charges (rapidité, précision, stabilité…).

Le domaine de l'automatique revêt un caractère primordial dans le domaine industriel auquel il tente d'apporter des solutions pour une large gamme de problèmes.

Depuis l'avènement des logiciels et matériels de toutes sortes, les recherches n'ont cessé de progresser afin d'établir des nouveaux modèles de commandes et répondre aux besoins de l'industrie.

Le travail décrit dans ce mémoire consiste en la conception d'un système automatique servant à réguler la température ambiante et l'humidité du sol ainsi la luminosité et la qualité de l'air dans une chambre en vue d'assurer un environnement propice à la survie des plantes. Pour cela, le système est composé de trois parties principales dont la première est composée de capteurs servant au recueil des données. Une deuxième partie de traitement de données est composée d'une carte à microcontrôleur Arduino UNO et du logiciel Matlab. La troisième partie est constituée des actionneurs servant à réguler les grandeurs déclarées qui sont commandées à des valeurs de consignes.

Le mémoire résumant le travail réalisé est organisé autour de quatre chapitres :

Dans le premier chapitre nous allons présenter la partie materielle et logicielle de la carte à microcontrôleur arduino, les capteurs et les parties puissances utilisés.

Le deuxième chapitre est consacré à la présentation des outils de communication entre le module Simulink du logiciel Matlab et la carte à microcontrôleur Arduino UNO.

Le troisième chapitre consiste en la description des outils théoriques exploités en commande numérique, avec une présentation sur le principe de la commande en boucle ouverte et fermée à base de régulateur (correcteur) PID (Proportionnel Intégral Dérivé) et flou, que nous allons adopter.

Le quatrième chapitre porte sur la conception et l'étude théorique du système à commander et les différents tests effectués.

La conclusion synthétise le travail réalisé et donne un aperçu des perspectives qui peuvent être développées.

I)- Introduction :

L'automatisme est une science technique, ou science de l'ingénieur, constituant l'une des branches les plus importantes de la physique appliquée, qui étudie et conçoit les structures effectuant des traitements de signaux électriques, porteurs d'informations désignant toute grandeur physique tel que : la température, l'humidité…etc., qui peut évoluer en temps réel selon une loi inconnue. Les applications automatiques peuvent être divisées en deux groupes distincts : le traitement de l'information et la commande. La première englobe les domaines tel que l'informatique, les télécommunications, les mesures, tandis que la seconde concerne la gestion de l'information par le biais d'instructions, générés par des systèmes électroniques tels que les microprocesseurs et les microcontrôleurs.

Ce chapitre est consacré à la présentation du matériels utilisés tel les capteurs exploités pour l'acquisition d'information, la carte à microcontrôleur Arduino servant à la réception et au traitement de données et les différents actionneurs .

II)-La carte Arduino :

La carte à microcontrôleur Arduino (ou son tout récent synonyme Genuino) est une carte électronique qui intègre un microcontrôleur d'architecture Atmel AVR (comme ATmega328 ou ATmega2560 pour les versions récentes ATmega168 ou ATmega8 pour les plus anciennes), et des composants complémentaires qui permettent la programmation et l'interfaçage avec d'autres circuits. Cette carte est une plate-forme à entrée/sortie simple qui est destinée à la programmation interactive qui peut être utilisée pour communiuquer avec des logiciels tel que le matlab lorsque elle est connectée à un ordinateur.

La carte Arduino contient tout ce qui est nécessaire pour le fonctionnement du microcontrôleur.

II.1)- Types de cartes Arduino :

Il existe plusieurs types de cartes Arduino, on trouve les originaux et les derivés (compatibles avec Arduino) . Parmis les plus utilisées dans le monde des systèmes embarquées on a : la UNO, la LEONARDO, la DUE, la MEGA et sa petite amélioration la MEGA 2560, et la Yun……..

Le tableau I.1 décrit brièvement quelques importantes caractéristiques des cartes précitées :

Arduino	Microcontrôleur	Flash ko	EEPROM ko	SRAM ko	Broches d'E/S numériques	Broches d'entrée analogique	Vitesse du Processeur
Uno	ATmega328P	32	1	2	14	6	16 MHZ
Leonardo	ATmega32U4	32	1	2,5	20	12	16 MHZ
Mega	ATmega1280	128	4	8	54	16	16 MHZ
Mega2560	ATmega2560	256	4	8	54	16	16 MHZ
Due	Atmel SAM3X8E	512	0	96	54	12	84 MHZ
Yun	ATmega32u4	32	1	2,5	20	12	16 MHZ

Tableau I.1 : Quelques cartes Arduino et leurs caractéristiques.

Notre choix s'est porté sur la carte UNO, car ses performances sont largement suffisantes pour gérer le fonctionnement de notre système. De plus son prix reste abordable, (environ 3500 dinar) pour la dernière version d'Arduino UNO.

II.2)- Cartes Arduino Uno :

II.2.1)- Partie matérielle:

La carte à microcontrôleur Arduino UNO est la première version stable de carte Arduino. Elle utilise un microcontrôleur AVR d'architecture ATmega328p cadencé à 16Mhz. Elle possède 32ko de mémoire flash destinée à recevoir le programme, 2kode SRAM (mémoire vive) et 1 ko d'EEPROM (mémoire morte destinée aux données). Elle contient 14 broches d'entrée/sortie numériques dont 6 peuvent être utilisées comme sorties PWM (Modulation à largeur d'imputions) et elle permet aussi de mesurer des grandeurs analogiques grâce à ces 6 entrées analogiques

La figure I.1 illustre une carte Arduino UNO **[1].**

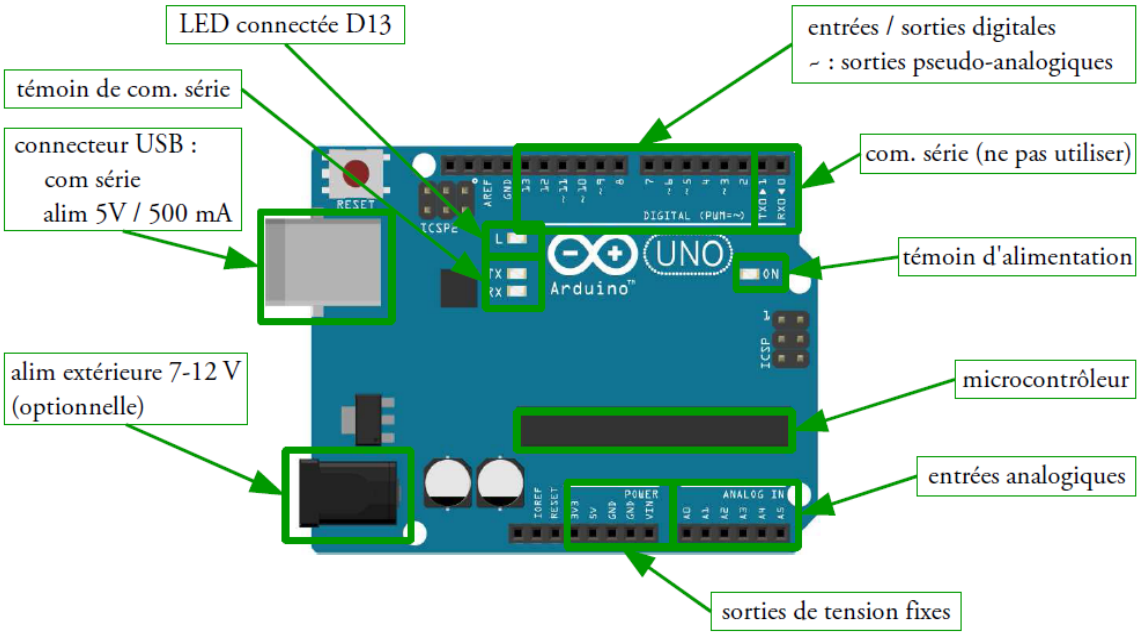


Figure I.1 : Carte Arduino UNO.

a)-Microcontrôleur :

C'est un circuit integré qui rassemble sur une puce plusieurs élèments dans un espace réduit, qui est un système à microprocesseur contenant des périphériques intégrés tel mémoire données, des programmes pouvant être utilisés comme un système embarqué.L'architecture de la carte Arduino a été publiée en open-source, et toute sa philosophie s'appuie sur le monde du libre, au sens large.

Les microcontrôleurs sont de plus en plus utilisés dans les applications embarquées tels les télèphones, les automobiles…….

La figure I.2 illustre un microcontrôleur ATmega328 de la carte Arduino Uno.

Figure I.2 : Microcontrôleur ATmega328.

Le microcontrôleur est composé de quatre parties principales :

-Un microprocesseur dont la fonction est le traitement des informations, composé d'une unité arithmétique et logique(UAL), d'un bus de données, d'adresse et de commande,ayant pour tâche l'exécution du programme embarqué dans le microcontrôleur.

-Une mémoire de données (RAM ou EEPROM) dans laquelle seront stockées les données temporaires nécessaires aux calculs qui est une mémoire de travail volatile.

-Une mémoire de programme (flash), contenant les instructions du programme à exécuter. Il s'agit ici d'une mémoire non volatile.

-La dernière partie correspond aux ressources auxiliaires qui sont :

- Des ports d'entrées / sorties (parallèle ou série).

-Des timers servant à générer ou mesurer des signaux.

- Des convertisseurs A/N pour le traitement des signaux analogiques.

b)-Entrées/Sorties numériques:

Cette carte possède 14 broches numériques (numérotée de 0 à 13) qui sont programmables par le biais d'instructions pinMode (), digitalWrite ( ) et digitalRead ().
Ces broches fonctionnent à 5V, chacune peut fournir ou recevoir un courant maximal de 40mA disposant d'une résistance interne de (rappel au plus) (pull-up) (déconnectée par défaut) de 20-50KOhms. Cette résistance interne s'active sur une broche en entrée à l'aide de l'instruction digitalWrite (broche, HIGH).
En plus, certaines broches ont des fonctions spécialisées par exemple:

-**Transmission et réception série:** les broches 0 (RX) et 1 (TX) permet de recevoir et transmettre les données séries de niveau TTL et ces pins sont connectés aux pins correspondants de l'USB-TTL puce Serial ATmega8U2.
-**Interruptions Externes**: les broches 2 et 3 peuvent être configurées pour déclencher une interruption sur un niveau bas, sur un front montant, descendant et sur un changement de valeur.
-**Modulation à largeur de bande (PWM : Pulse Width Modulation)** : les broches 3, 5, 6, 9, 10,11 peuvent fonctionner en mode PWM pour faire varier la puissance du signal envoyé sur 8 bits à l'aide de l'instruction analogWrite () **[2]**.

c)-Entrées analogiques:

La carte UNO dispose de 6 entrées analogiques (A0 à A5), où chacune peut fournir une mesure avec fonction analogRead ( ) du langage Arduino sur une résolution de 10 bits (de 0 à 1023). Par défaut, ces broches mesurent une tension comprise entre le 0V correspondant au niveau 0 et le 5V correspondant au niveau 1023. Notons qu'il est possible de modifier le niveau supérieure de la plage de mesure en modifiant la tension sur la broche AREF ou en utilisant l'instruction analog Reference ( ) du langage Arduino.

Note :

La carte UNO peut-être alimentée soit via le port USB ou à l'aide d'une alimentation externe qui consiste à brancher une batterie au connecteur qui s'appelle (prise jack).

II. 3)- Partie logicielle :

Le logiciel IDE (**Integrated Développement Environement**) Arduino offre une interface minimale pour rédiger des programmes appelés « Sketch », les compiler et les transférer dans la carte Arduino à travers une liaison USB, intégrant aussi un moniteur de port série pour l'affichage des données.

II.3.1)- Langage de programmation :

Un langage de programmation est un langage permettant d'écrire un ensemble d'instructions qui seront converties en langage machine grâce à un compilateur. L'avantage du langage Arduino est qu'il est basé sur les langages C/C++ qui supporte toutes les syntaxes standards du langage C et quelques-unes des outils du C++. En plus, la disponibilité des librairies permettent de faciliter la communication avec le matériel connecté à la carte (Afficheurs LCD, Afficheurs 7 segments, capteurs, servomoteurs… etc.).Pour écrire un programme avec le langage Arduino, il faut respecter certaines règles. Notons que l'exécution d'un programme Arduino s'effectue d'une façon séquentielle, c'est-à-dire que les instructions sont exécutées les unes à la suite des autres. Avant tout le compilateur doit vérifier l'existence de deux structures obligatoires à tout programme en langage Arduino qui sont :

-la partie initialisation et configuration des entrées/sorties (la fonction setup ()).

-la partie principale qui s'exécute en boucle (la fonction loop ()).

II.3.2)- Structure générale d'un programme en langage Arduino :

Pour utiliser l'IDE standard Arduino (arduino.exe), il suffit de saisir le code dans la fenêtre dédiée, de compiler et de téléverser le programme sur la carte Arduino. La carte doit être reliée à l'ordinateur via un câble USB. Le modèle de la carte Arduino ainsi que le port série sur lequel elle est branchée doivent être déclarés dans le menu de l'IDE Outils/type de carte et Outils/port série.

Une fois le programme compilé téléversé dans le microcontrôleur, son exécution sera lancée. La fonction setup ( ) s'exécute une seule fois après la mise sous tension et après chaque redémarrage. Par contre la fonction loop () s'exécute en boucle.

La figure I.3 illustre la structure générale d'un programme IDE.

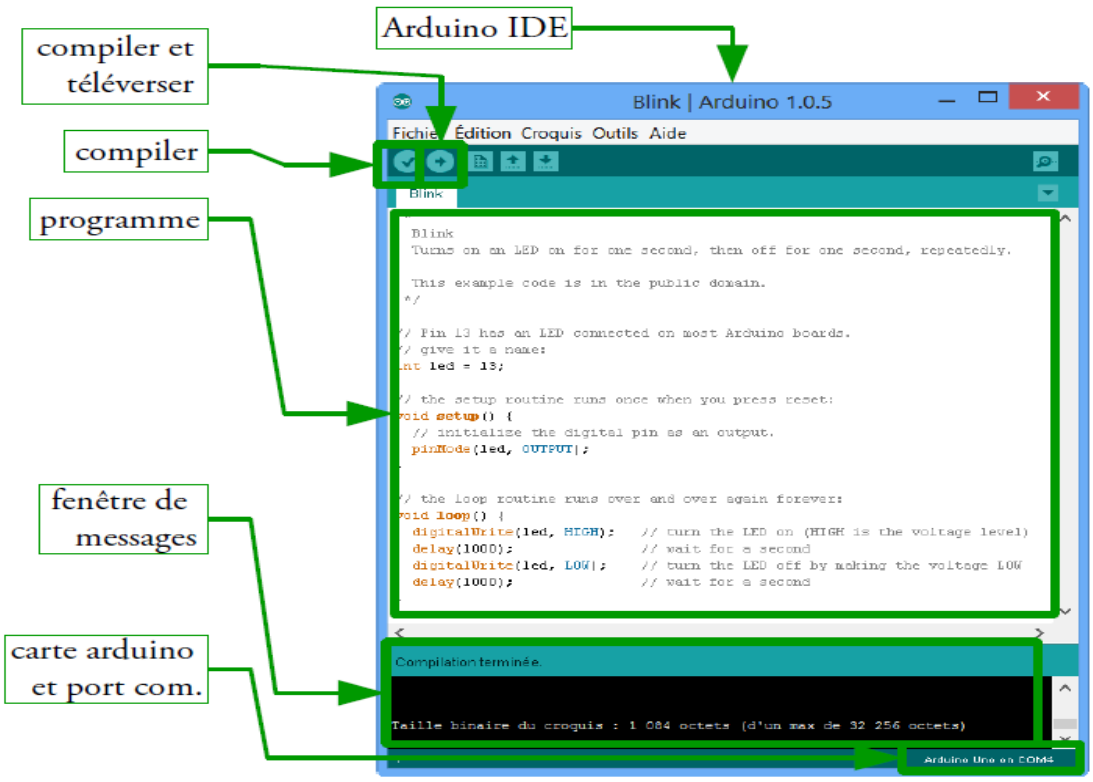


Figure I.3 : Structure d'un programme.

Un programme Arduino comporte trois parties principales :

1. la partie déclaration des variables (optionnelle).
2. la partie initialisation et configuration des entrées/sorties : la fonction setup ().
3. la partie principale qui s'exécute en boucle : la fonction loop ().

Dans chaque partie du programme sont utilisées différentes instructions issues de la syntaxe du langage Arduino.

II.3.3)- Etapes de l'implémentation sur le microcontrôleur :

-La création d'un projet.

-L'écriture du programme ensuite enregistrement.

-La vérification de la syntaxe et correction d'éventuelles erreurs.

-Le télé versement vers le microcontrôleur.

III)- Composants électroniques :

III.1) Capteurs :

Un capteur à pour fonction la conversion d'une grandeur physique en une grandeur électrique pouvant être exploitable dans une chaîne de mesure.

III.1 .1)- Capteur d'humidité YL-69 :

a)-Description :

Le capteur d'humidité du sol (YL-69) ou l'hygromètre est généralement utilisé pour détecter l'humidité du sol. Ainsi, il sert à construire un système d'arrosage automatique ou pour surveiller l'humidité du sol des plantes.Le capteur est configuré en deux parties: la carte électronique et la sonde à deux électrodes qui détecte la teneur .ce dernier dispose d'un potentiomètre intégré pour le réglage de la sensibilité de la sortie numérique (D0), d'une LED d'alimentation et d'une LED de sortie numérique.

la figure I.4 decrit la carte electronique du capteur yl-69 :

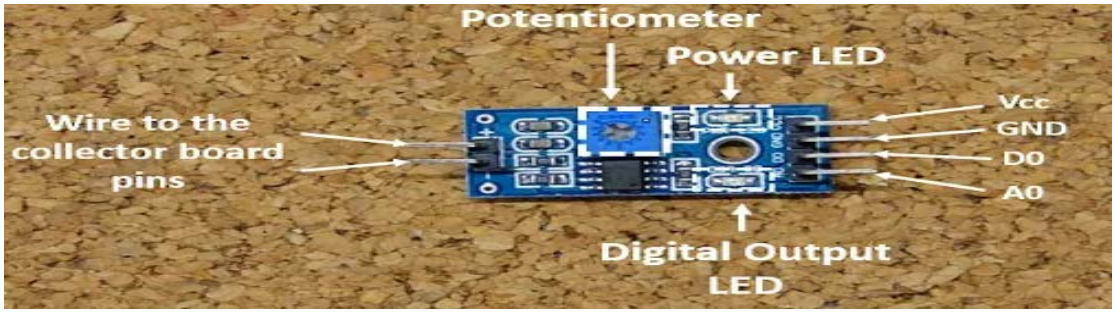


Figure I.4 :Carte electronique du capteur yl-69.

b)- Branchement sur la carte Arduino :

Pour exploiter le capteur yl-69 il suffit :

-De relier la sonde a la carte électronique avec deux files .

-D'allimenter les pattes VCC et GND de la carte .

-De brancher les pattes au milieu de la carte a des entree analogique (A0 ..A5) et des E/S digitales (0..13) de la carte arduino comme le montre le schéma de la figure I.5.

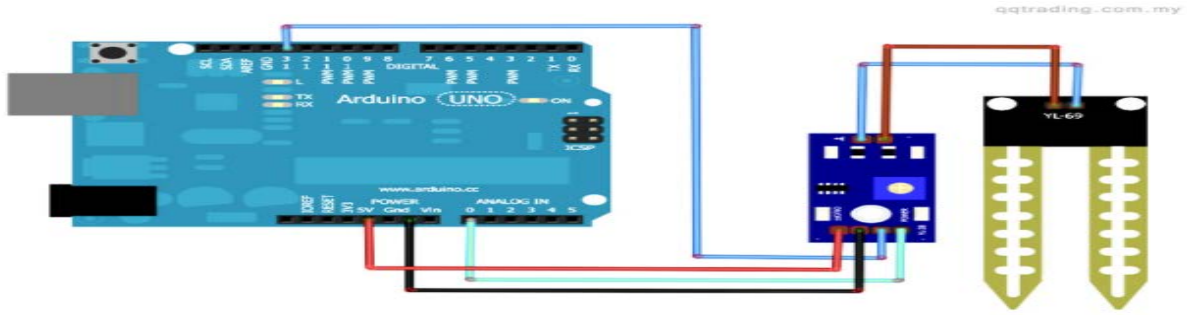


Figure I.5 : Branchement du capteur avec arduino.

c)- Caractéristiques du capteur d'humidité :

Le tableau I.2 résume les caractérisques du capteur d'humidité YL-69 **[3]** :

VCC	3,3V ou 5V
GND	GND(0V)
AO	0 - 200 dans l'eau 200 -400 milieu d'arrosage parfait 400 – 700 milieu d'arrosage moyen 700_ 950 sol sec 1023 dans l'air
DO	0 OU 1
taille	60×20×5MM
Courant	35 MA

Tableau I.2 : Caractéristiques d'humidité YL-69.

III.1.2)- Capteur de température LM-35 :

a)- Description:

Le LM-35 fait parti des capteurs de température electronique de precision en structure integré. Ce capteur convertit la temperature en une valeure de tension analogique proportionelle à la temperature avec une sensibilité de 10mV /C et une gamme de température comprise entre 0°C a 70°C .

La figure I.6 ilustre un capteur LM35.

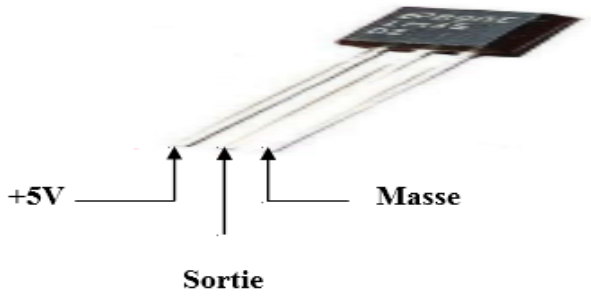


Figure I.6 : Capteur LM35.

b)-Branchement du capteur avec la carte arduino :

Pour exploiter le capteur LM-35 il suffit :

-D'allimenter les pattes VCC et GND .

-De brancher la patte au milieu a une entrée analogique (A0 ..A5) de la carte arduino comme le montre le schema de la figure I.7 .

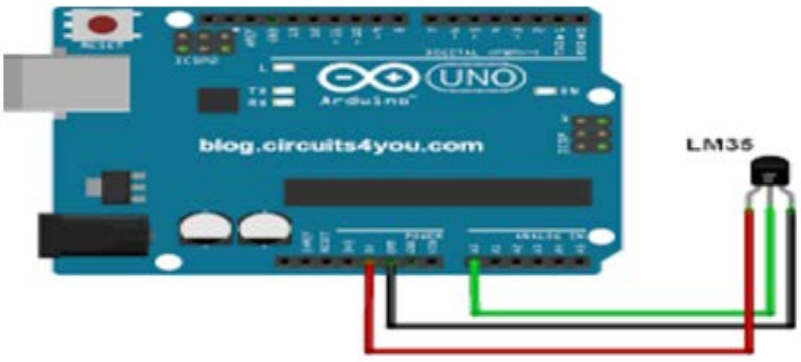


Figure I.7 : Branchement du capteur avec arduino.

c)-Caractéristiques LM35 :

Le tableau I.3 résume les caractérisques de capteur LM35 **[1].**

Calibration	Degrés cilsius (C°)
Sensibilité	Facteur d'échelle de 10 mV / ° C
Gamme de fonctionnement (grand public)	0 ° C à 70 ° C
Alimentation Vcc	4 V à 30 V
Non-linéarité	seulement ± ¼ ° C typique
Taille	4.699 mm × 4.699 mm
GND	GND (0 V)

Tableau I.3 : Caractéristiques LM35.

III.1.3)-Capteur de lumière (photorésistance):

a)-Description:

La photorésistanceLDR (Light Dependent Resistor) est un dipôle dont la résistance dépend de la lumière qu'il reçoit. La partie sensible du capteur est une piste de sulfure de cadmium en forme de serpent : l'énergie lumineuse déclenche une augmentation de porteurs libres dans ce matériau, de sorte que sa résistance électrique diminue a priori.
La figure I.8 représente la photorésistance.

FigureI.8 : Photorésistance.

b)-Branchement du capteur avec la carte Arduino :

pour exploite le capteur photorésistance il suffit d'alimenté les deux pattes comme suivant : Une a la branche 5V de l'arduino ,l'autre a l'entrée analog read de la carte uno par l'intermidiaire d' une résistance de 10KOm comme le montre le schéma de la figure I.9.

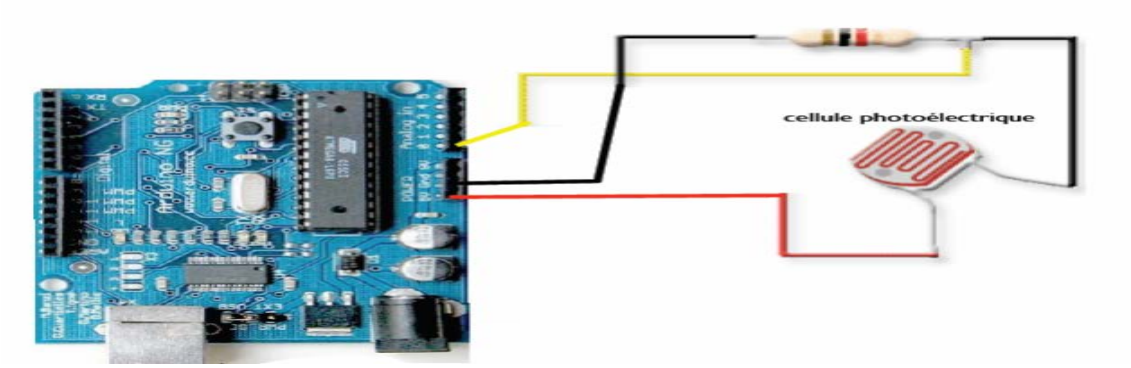


Figure I.9 : Branchement du capteur avec Arduino.

c)-Caractéristiques de la photorésistance :

Le tableau I.4 résume les caractérisques de la photorésistance **[2]:**

Résistance à la lumiére	1 KΩ
Résistance dans l'obscurité	10 KΩ
Tension max	150 V
Puissance max	100 MW

Tableau I.4 : Caractéristiques de photorésistance.

III.1.4)- Capteur de pollution MQ135 :

a)- Description :

Le MQ135 est un capteur qui permet de mesurer la qualité de l'air. Le MQ135 est sensibleaux principaux polluants présents dans l'atmosphère de la maison. Ce capteur est

sensible au CO2, à l'alcool, au Benzène, à l'oxyde d'azote (NOx) et à l'ammoniac (NH3). Ce capteur est plus économique pour mesurer la présence de CO2 dans une pièce.

b)- Caractéristiques :

-Tension nominale: 5V.

-Sorties: Analogiques et numérique tout ou rien selon un seuil réglable.

-Détecte NH3, NOx, alcool, benzène, fumée et CO2.

-Stable et Haute sensibilité: 10 - 300 ppm NH3, 10 - 1000 ppm Benzène, 10 - 300 Alcool.

-Plus la concentration est importante plus la sortie analogique est élevée.

-Dimensions: 18mm diamètre, 17mm hauteur sans pin, Pins - 6mm de haut **[2]**.

La figure I.10 represene un capteur MQ135.

Figure I.10 : Capteur MQ135.

c)-Branchement du capteur avec Arduino :

Pour exploite le capteur il faut le calibré et realiséle schéma illustré dans la figure I.11 :

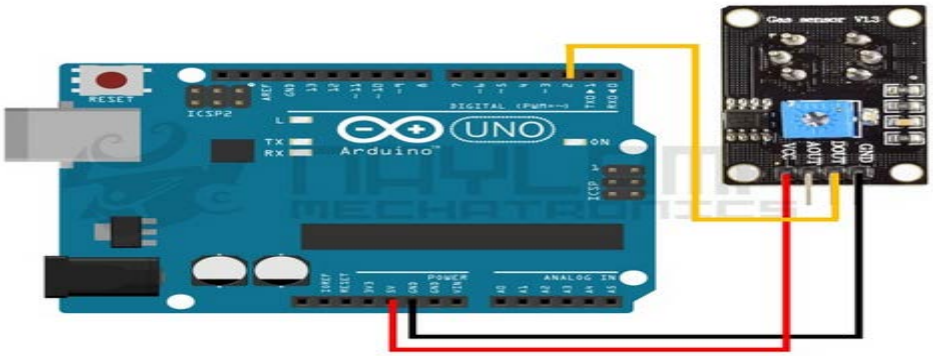


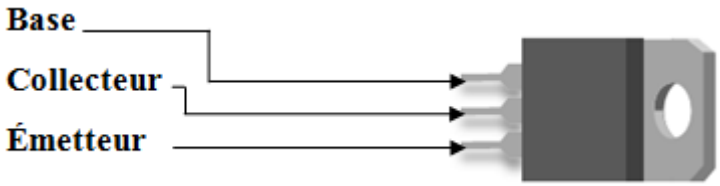
Figure I.11 :Branchementdu capteur MQ135 avec arduino.

III.2)-Transistor TIP121 :

a)-Description:

C’est un transistor Darlington NPN qui permet d’amplifier le courant jusqu’à 5A avec son gain d’amplification au minimum β= 1000 d’après la fiche technique, il possède trois pattes : Base, Collecteur, Émetteur

La figure I.12 ilustre un Transistor TIP121.



FigureI.12 : Transistor TIP121.

b)-Caractéristiques du TIP121 :

Le tableau I.5 résume les caractérisques de TIP121 **[2]** :

Voltage Collecteur-émetteur	80V
Voltage Collecteur-base	80V
Voltage Emetteur –base	5V
Courant de Collecteur	5A
Collecteur dissipation	65W

Tableau I.5 : Caractéristiques du TIP121.

c)-Branchement du transistor avec la carte arduino :

Pour le branchement de transistor TIP121 la borne numéro (1) est le donneur d’ordre (base) qui est connecté à une résistance 1kΩ, qui est elle-même connecté à l’Arduino,celle du milieu connecté à la borne masse de la distination (émetteur) et la borne (3) à la masse de la source (collecteur).

La figure I.13 illustre un branchement standart du transistor TIP 121.

Figure I.13 : Branchement du TIP121 avec Arduino.

III.3)-Servomoteur 9G:

Le 9G est un micro servomoteur dont les caractéristiques sont les suivantes **[2]** :

-Dimensions : 22 x 11.5 x 27 mm.

-Tension d'alimentation : 4.8v à 6v.

- Vitesse : 0.12 s / 60° sous 4.8v.

- Couple : 1.2 Kg / cm sous 4.8v,amplitude : de 0 à 180°.

La figure I.14 représente un servomoteur 9G.

Figure I.14 : Servomoteur 9G.

III.4)-Pompe d'arrosage :

Une pompe est un dispositif permettant d'aspirer et de refouler un fluide. Le choix d'une pompe à eau dépend de prime abord de l'usage que l'on veut en faire. Même si toutes

les pompes servent à déplacer des liquides, elles ne se servent pas pour autant à assurer la distribution de l'eau. Pour un client averti, certains critères indispensables sont à retenir : le débit, la hauteur manométrique totale, le type de pompe et le modèle. Pour notre prototype, nous avons utilisé une pompe d'essuie-glace pour voitures alimentée par une source de tension 12V.

La figure I.15 représente un Moteur pompe 12V pour esuiglasse .

Figure I.15 : Moteur pompe 12V.

III.5)-Alimentation :

Nous avons besoin d'une alimentation avec les caractéristiques suivantes :

-Entrée : une tension alternative de 220V /50Hz.

-Sorties : deux tensions continues de 12V pour les actionneurs.

Note:

La sortie 12V doit fournir un courant assez élevé.

Pour notre réalisation, nous avons utilisé une alimentation présentant les caractéristiques présentées sur le tableau I.6.

Dimension	18×14×7cm
Puissance nominale	400 W
Tensions de sortie	- +3.3V à 28A ; - +5V à 40A ; - +12V à 17A ; - -12V à 0.8A.
allumage	Bouton on-off

Tableau I.6: Caractéristiques de l'alimentation.

La figure I.16 représente une alimentation d'un Alimentation.

Figure I.16 : Alimentation 0_30V**.**

III.6)-LED 12V :

La diode électroluminescente LED (light-emitting diode) est un dispositif capable d'émettre de la lumière lorsqu'il est parcouru par un courant électrique.Cette diode ne laisse passer le courant que dans un seul sens.

La figure I.17 represente une LED.

Figure I.17 :LED.

III.7)-Résistance chaufante :

Les résistances blindées tubulaires sont des résistances chauffantes offrant une large gamme d'applications,composé d'un fils chauffant nickel-chrome .

La figure I.18 ilusre des resistance chaufante .

Figure I.18 :Résistances chaufantes.

III.8)- Circuit L293D :

a)-Présentation :

Le L293D est un double pont-H , ce qui signifie qu'il est possible de l'utiliser pour commander la vitesse ainsi que le sens de rotation en raccordant les sorties de façon appropriées,il estpossible de constituer deux pont-H. (Le l293d un circuit intégré monolitique, à haut voltage ,grand courant et 4 canaux).Cela veut dire que ce circuit intégré peut être utilisé pour des moteurs DC alimentés jusqu'à36 volts. Le circuit peut fournir un maximum de 600 mA par canal.**[2]**

En utilisant différentes combinaisons de Input 1 et Input 2 il devient possible de démarrer,stopper ou inverser le courant ainsi commander la vitesse (les branches responsables enable 1 enabel 2, les pattes1 et 9 du l293d…).[**2]**

Figure I.19 : Circuit L293D

b)-Brochage :

Ci-dessous la configuration des broches du L293D et la table de la logique de commande sont ilustrés a travers la figure I.20 et le ableau I.7 **[4].**

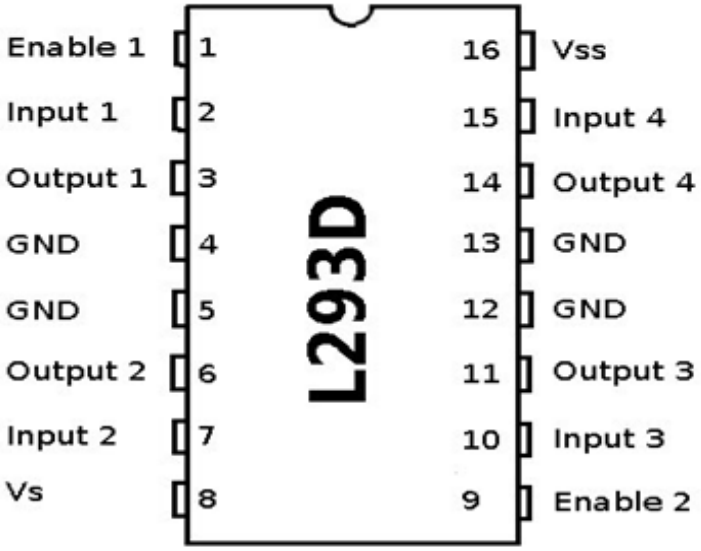


Figure I.20: Bronches du circuit l293D.

Broche	**Nom**	**Description**
1et9	Enable 1 et 2	-permet d'envoyer (ou pas) la tension sur les sorties du moteur via OUTPUT1 & OUTPUT2. -ENABLE commande l'activation/désactivation du Pont-H. Exmp : -Si ENABLE1 = GND, le pont-H est déconnecté et le moteur ne fonctionne pas. -Si ENABLE1 = VSS, le pont-H est connecté aux sorties et le moteur fonctionne dans un sens ou l'autre ou pas en fonction des tensions appliquée sur INPUT1 & INPUT2.
2et7	Input 1 et 2	sont les broches de commande du Pont-H Output1/Output2. Ou Output 3 et Output 4 . Se raccorde a Arduino, permet de commander le sens du courant entre Output 1 et Output 2. Ou Output 3 et Output 4 .
3 - 6	Output 1 e 2	seront les broches à raccorder à la charge (le moteur).
4-5-12-13	GND	Doit être raccorder à la masse (GND) de la source d'alimentation de puissance VS (ex: la borne négative de l'accumulateur +9.2v) et à la masse de la source d'alimentation de la logique "VSS" (donc GND Arduino).

		Si vous n'avez qu'une source d'alimentation pour le tout, c'est forcement plus simple.
8	VS	Alimentation de puissance des moteurs.
10-15	Input 3 et 4	A utiliser conjointement avec Input 4 pour commander le pont-H Output3/Output4.
11-14	Output 3 et 4	deux sorties du second pont-H (Output3/Output4)
16	VSS	Alimentation de la logique de commande (5V). A raccorder à la borne +5V d'Arduino (donc sur le régulateur d'Arduino).

TableauI.7 : Broches et signifiquations du l293d.

c)-Caractéristiques :

Le tableauI.8 résume les caractérisques du L293D**[4]** :

Nbre de pont-H	2
Courant Max Régime continu	600mA (x2)
Courant de pointeMax< 2ms	1200mA
V_S Max Alim moteur	36v
V_S Max Alim logique	7V
Nombre de Broche	16DIP
Perte de tension	1.3 à 1.4v (typical)

TableauI.8 : Caractéristiques du L293D.

V)-Conclusion :

A travers ce chapitre nous avons présenté les différentes parties constituant le système à concevoir avec leurs caractéristiques. Ce système est composé d'une carte à microcontrôleurs Arduino (Uno), les capteurs : de température, d'humidité, luminosité et qualité de l'air. Des actionneurs : pompe à eau, résistance chauffante, LED et un servomoteur. Pour cela dans les prochains chapitres nous allons aborder l'interfaçage Arduino-Simulink et l'étude des méthodes de commande et de régulation que nous allons adopter pour la conception de notre propre système de régulation.

I)- Introduction :

Le prototypage informatique est une méthode commandée par ordinateur, généralement par superposition, cette dernière regroupe un ensemble d'outils qui sont agencés entre eux et permettent de réaliser des interfaces (logiciels et matériels). Son avènement est récent, mais son avancée majeure n'a cessé de progresser donnant lieu au développement de nouvelles techniques de conceptions et de commandes pour répondre aux besoins de l'industrie manufacturière dans plusieurs domaines y compris l'automatisme. L'avantage apporté par cette technique est l'exploitation interactive entre les différents modules.

Servant a la conception, régulation, commande ainsi que la réalisation des systèmes notons que (Arduino et Simulink / Matlab) est un outil innovant à coût modique pour le prototypage.

Dans ce chapitre nous allons développer une solution qui consiste à assurer une communication entre deux logiciels (Arduino et Matlab/Simulink) y compris la carte (Arduino Uno) toute en exploitant le package (Arduino IO) avec les commandes accessibles.

II)- Logiciel Matlab :

C'est un logiciel de calcul mathématique pour les ingénieurs et les scientifiques créé par Mathworks. Ce logiciel est un environnement de programmation pour le développement d'algorithme, d'analyse de données, de visualisation et de calcul numérique.

Avec le MATLAB, la résolution des problèmes de calcul complexes se fait plus rapidement qu'avec des langages de programmation évolués, tels que le C, C++ et le Fortran …etc. **[5]**

II.1)- Module Simulink :

Il s'agit d'un environnement de diagramme fonctionnel pour la modélisation des systèmes dynamiques et le développement d'algorithmes.

Il fournit un environnement graphique interactif et un ensemble de bibliothèques de blocs qui permettent de concevoir, simuler, mettre en application, et examiner une variété de systèmes, tel que les systèmes de communications, de commandes, de traitement des signaux, de traitement visuel, et de traitement d'image **[5]**.

Il s'agit d'un environnement de diagramme fonctionnel pour la modélisation des systèmes dynamiques et le développement d'algorithmes.

II.2)- Interfaçage (Arduino /Simulink) :

L'interfaçage d'Arduino avec Simulink permet de construire des projets Arduino en utilisant une programmation de haut niveau et des diagrammes de blocs.
Le module de prise en charge MATLAB pour Arduino nous permet de communiquer via USB avec notre carte Uno et les périphériques connectés .Etant donné que MATLAB est un langage interprété de haut niveau, on peut voir immédiatement les résultats des instructions d'entrées sorties sans les compiler et aussi il inclut des fonctions mathématiques, d'ingénierie et de traçage intégrées que l'on va utiliser pour analyser et visualiser les données de notre système.

Il existe plusieurs possibilités d'interfacer la carte Arduino avec Matlab/Simulink,
tels que :

1. Programmation de la carte Arduino Uno comme une carte d'interface.
2. Utilisation du package ArduinoIO.
3. Utilisation du package Arduino Target.

Le langage amélioré et la capacité à tracer facilement les données des capteurs nous attirent tant que automaticiens à l'utilisation du package Arduino IO sur Simulink.
Le module de support Simulink pour Arduino nous permet de développer des algorithmes autonomes sur notre carte.

Après la création d'un modèle sous Simulink on pourra bien le simuler, ainsi régler les paramètres de l'algorithme et le télécharger pour une exécution indépendante sur le périphérique. Un des avantages majeurs à tirer de l'utilisation du module Simulink est la possibilité de régler les paramètres en direct de notre ordinateur pendant que l'algorithme fonctionne sur le matériel.

II.3) -Package « Arduino IO » :

Cette solution (serial communication) consiste à utiliser la carte Arduino comme une interface d'entrées (analogiques/numériques) et de sorties (numériques) en permettant de

communiquer Matlab ou Simulink avec la carte Arduino via un câble USB.

Cette interface consiste à pré-charger le programme (adio) dans la carte Arduino afin que celle-ci fonctionne en serveur.

Ce programme consiste à recevoir les requêtes envoyées via la liaison série (USB) et de répondre à ces requêtes en renvoyant l'état d'une entrée ou en modifiant l'état d'une sortie.

Ces mêmes (entrées/sortie) sont vues dans Matlab comme des entrées logiques ou analogiques (utilisation du CAN) ou des sorties analogiques (mode PWM).

Avant d'accéder à cette solution il faut :

1. Télécharger le package ArduinoIO
2. Décompresser à la racine de votre disque dur, exemple E : \arduinoio
3. Ouvrir le dossier décompressé.
4. Aller vers : "ArduinoIO\pde\adiosrv"
5. Charger le fichier (**adio pde)** vers le logiciel Arduino comme le montre la figure II.1 :

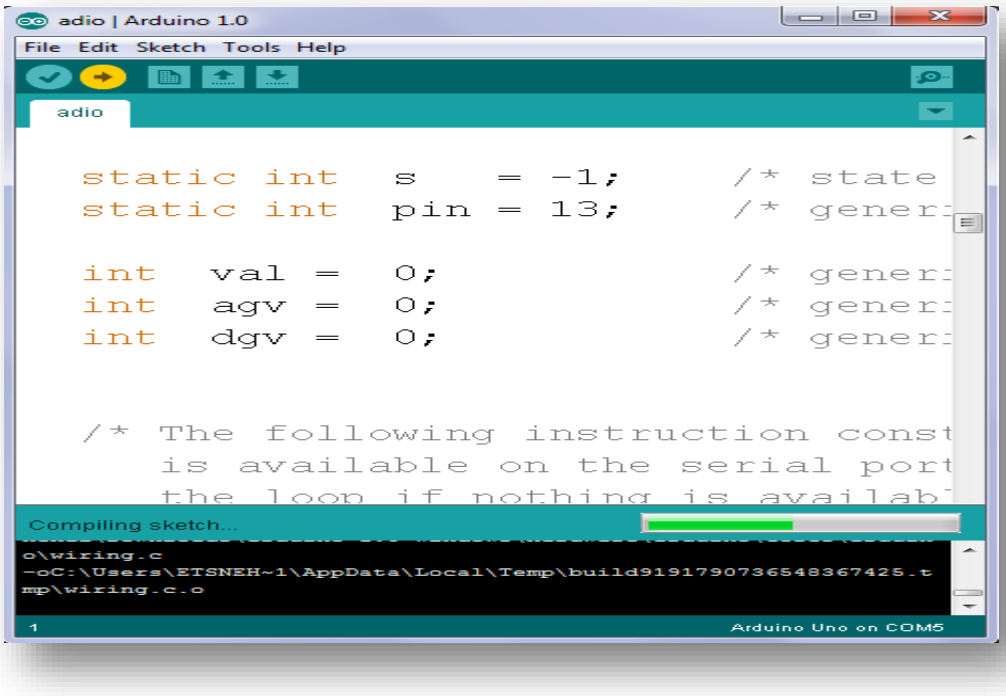


Figure II.1 : Programme adio sous Arduino.

Une fois que le module de support Arduino installé et le programme adio chargé, nous constaterons que dans les bibliothèques MATLAB Simulink, l'une des bibliothèques sera la bibliothèque Arduino IO.

Cette bibliothèque contient 12 blocs opérations qu'nous pourrons faire connecter avec

Simulink et Arduino comme elle permet d'envoyer les informations du MATLAB a la carte Arduino connecté ainsi que de recevoir de la carte, c'est se que on appel (serial communication).

Les blocs opérations sont illustrés dans la figure II.2 suivante:

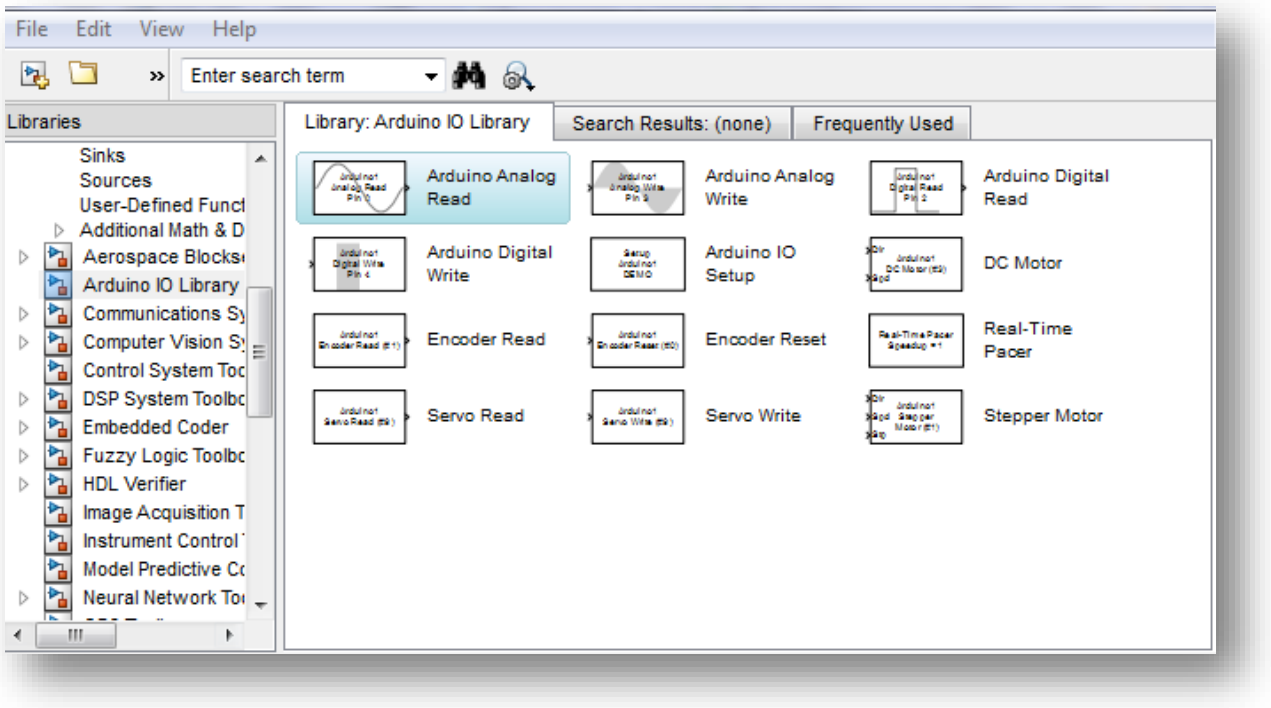


Figure II.2: Arduino-io Simulink library.

Les blocs nécessaires pour le développement de notre application prévus d'être utilisés dans le système de régulation et de commande sont comme suit :

a)- Real-Time pacer :

Ce bloc permet de ralentir le temps de simulation de sorte qu'il synchronise avec le temps réel écoulé.

Le coefficient de ralentissement est contrôlable par l'intermédiaire du paramètre Speedup. Pour le configurer voir la figure II.3.

b)-Arduino IO Setup :

Ce bloc est pour configurer sur quel port la carte Arduino UNO est connectée. Pour cela il suffit de voir dans Gestionnaire des périphériques, puis déclarer le même port dans se bloc. Pour le configurer voir la figure II.3.

c)- Arduino Analog Read :

Ce bloc sert à recevoir les données analogiques du capteur et pour configurer à partir de quel pin [0, 1, 2, 3, 4, 5] cette opération s'exécute.

d)- Arduino Analog Write :

Pour configurer à partir de quel pin [3, 5, 6, 9, 10, 11] on va envoyer la commande en PWM vers l'actionneur.

e)-servo write :

Pour configurer à partir de quel pin [3, 5, 6, 9, 10, 11] on va envoyer la commande en PWM vers le servomoteur ainsi les angles de rotation.

III) -Programmation de la carte Arduino sous Simulink:

Pour programmer la carte Uno dans la fenêtre Simulink il suffit de déclarer le port qui convient ainsi le temps real idéal pour le fonctionnement désiré comme le montre la figure II.3 suivante :

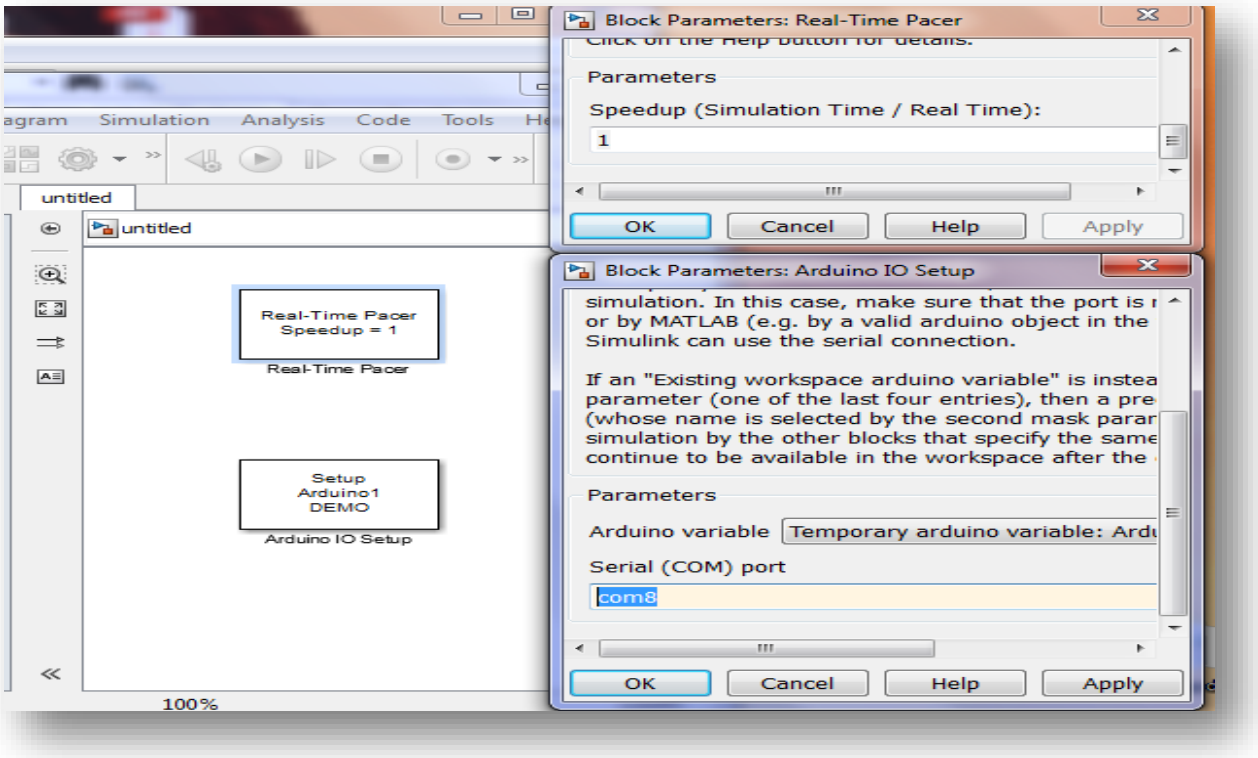


Figure II.3: Programmation d'une carte Arduino sous Simulink.

III.1)- Acquisition de données :

Il suffit d'utiliser les blocs suivants et de les configurer selon le cahier de charge désiré.

- Arduino Analog Read
- Arduino IO Setup
- Real-Time racer
- Un Display ou Scope pour assurer la lecture de données en temps réel.
- Un gain pour le changement d'échelle de l'analogique vers numérique.

La figure II.4 montre un exemple d'acquisition de données sous Simulink.

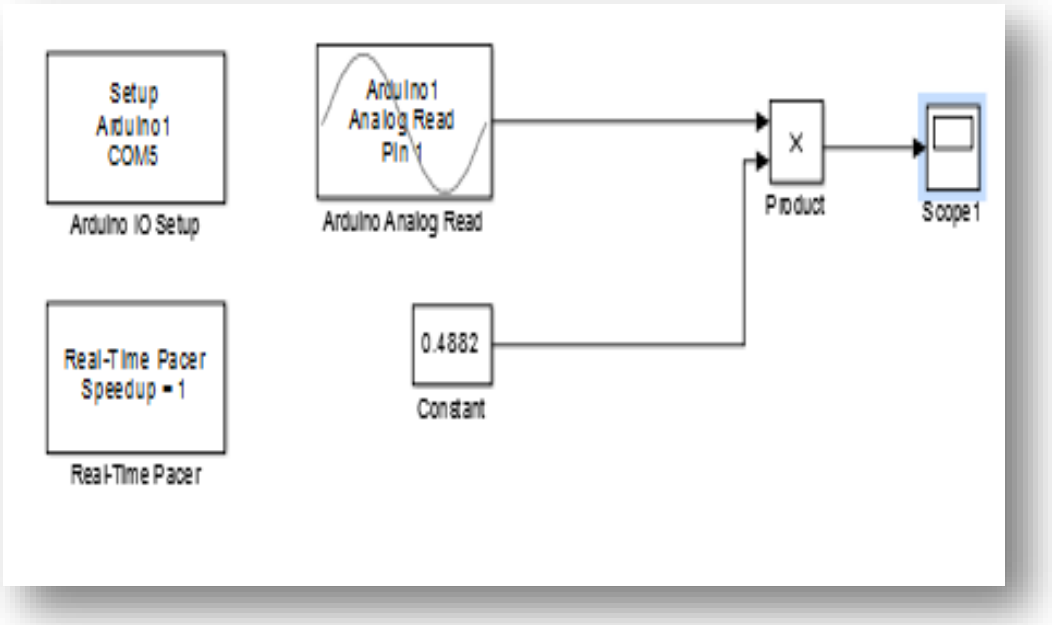


Figure II.4: Acquisition des donnés sous Simulink.

III.2)- Envoi des données :

Il suffit d'utiliser les blocs suivants et de les configurer selon le cahier de charge désiré.

- Arduino Analog write
- Arduino IO Setup
- Real-Time racer
- Un bloc Display ou Scope pour assurer la lecture de données en temps réel

La figure II.5 montre un exemple d'envoi de données à partir de Simulink.

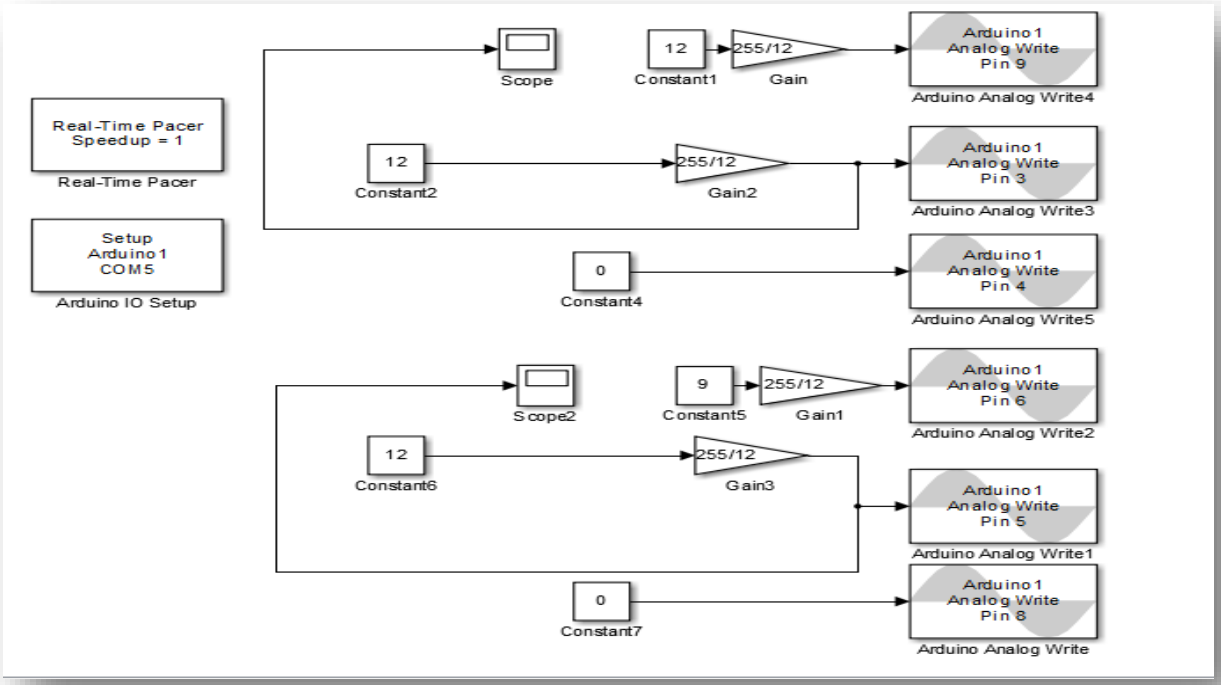


Figure II.5 : Envoie de donné a partir de Simulink.

III.3) -Utilisation de la Commande PWM dans Simulink :

Comme cité dans le premier chapitre les pines (3, 5, 6, 9, 10,11) sont capables, tout comme les autres, d'envoyer soit un +5 V, soit du 0 V en mode OUTPUT. Mais ils ont un autre avantage, ils peuvent envoyer un train d'impulsions variable. Nous les commandons avec une fonction de Simulink qui va prendre une valeur entre 0 et 255.C'est cette valeur qui va définir la part de temps pendant lequel le pin sera à l'état HAUT durant chaque intervalle, (255 correspondra à 100% du temps d'intervalle à l'état HAUT). Le temps de cycle du début d'une impulsion à un autre est de 2 ms, nous appelons ce temps la période du cycle des impulsions.

Toutes les 2 ms, le pin va rester à l'état HAUT pendant un pourcentage du temps et à l'état BAS pour le reste des 2 ms.

La figure II.6 suivante montre la variation d'un train d'impulsion.

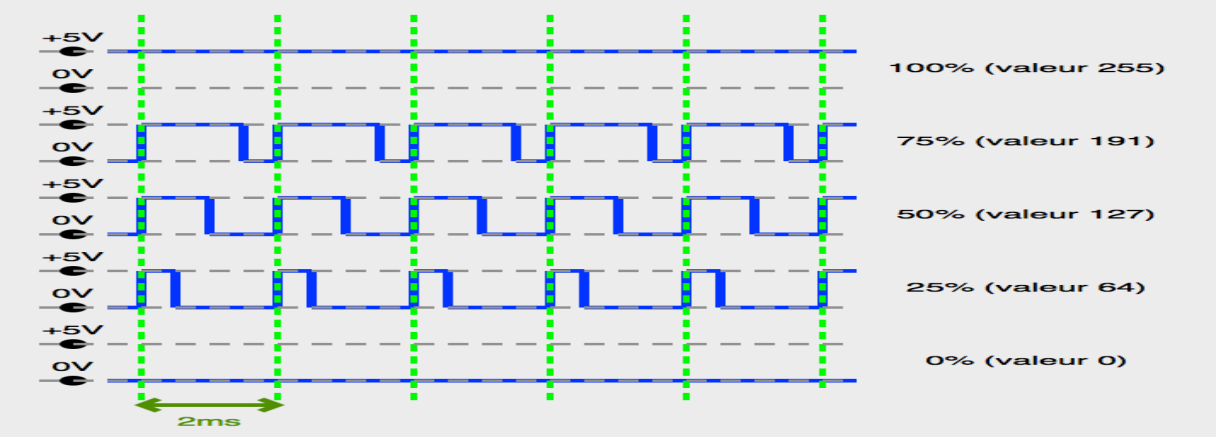


Figure II.6 : Variation d'un train d'impulsion.

Cycles d'impulsions : c'est le pourcentage du temps où la pin est à l'état HAUT (sur 2 ms), et la valeur du paramètre associé.

Nous pourrions imaginer un moteur saccadé (il s'arrête, il repart, il s'arrête, il repart...) mais en fait pas du tout.

Rappelons-nous que le moteur a une inertie mécanique (il ne s'arrête pas d'un coup mais continue de tourner même sans électricité) donc le résultat est un peu comme si on le relançait par petits coups.

Le cycle des impulsions est sur 2 ms, donc on le relance plus ou moins sur ce temps très rapide.

Nous qualifions le **rapport cyclique** : le temps à l'état haut (Th) divisé par la période du cycle (T) ce qui donne :

$$R = \frac{Th}{T} \qquad \text{(II.1)}$$

Ce rapport est situé entre 0 et 1 **[6].**

La figure II.7 suivante nous montre une description d'un rapport cyclique.

Figure II.7 : Rapport cyclique.

La figure II.8 suivante nous montre un schéma Simulink qui nous permet de contrôler la vitesse d'un moteur de 12v avec ce type de commande (PWM). Il suffit juste de varier la constante6.

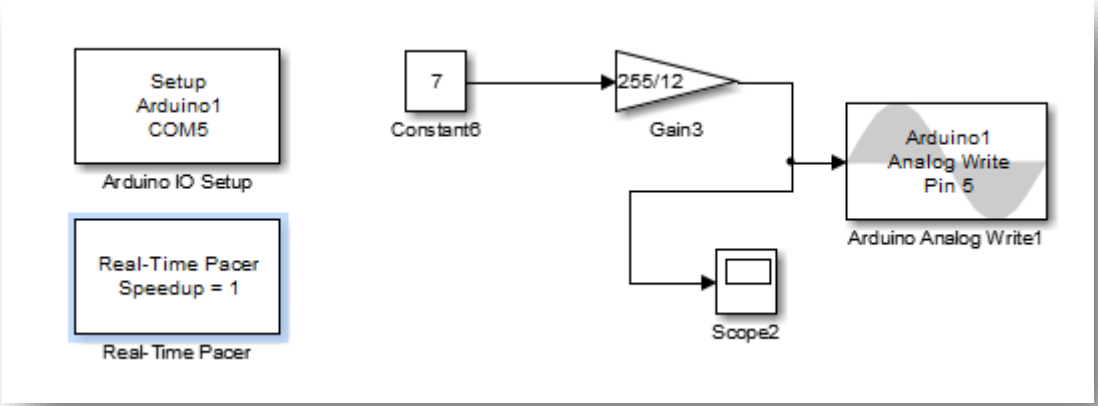


Figure II.8 : Schéma de simulation d'un moteur 12V.

III.5) -Exemples des commandes qui sont accessibles sur workspace :

1) Instancier un objet Arduino à partir de MATLAB.

```
>> a = Arduino ('COM5');
```

2) Connecter et contrôler les entrées et sorties Arduino.

```
>> av = readVoltage (a, 5) ;
>> ac= analogRead (a, 5);
>> writeDigitalPin (a, 13, 1) ;
>> supprimer (a)
```

3) Contrôler la position d'un bouclier moteur Adafruit V2.

```
>> a = Arduino ('com5', 'Uno', 'bibliothèques', 'Adafruit \ MotorshieldV2');
>> bouclier = addon (a, 'Adafruit \ MotorshieldV2')
>> s = servo (bouclier, 1);
>> writePosition (s, 0.5)
>> dcm = dcmotor (bouclier, 1);
>> commencer (dcm)
>> pause (4)
```

>> arrêter (dcm)

>> sm = stepper (bouclier, 1,200, 'RPM', 10)

>> déplacer (sm, 10).

Ect…….

Note :

L'idée générale à garder à l'esprit quand il s'agit de câbler Arduino à Matlab avec se mode de communication (serial communication) c'est qu'ils doivent communiquer en utilisant le même langage et le même protocole en partagent les informations et le port série physique dont ils communiquent.

IV) -Conclusion :

Notre choix s'est porté sur la combinaison des deux environnements Matlab et Arduino car elle nous permettra d'associer un logiciel de simulation puissant à une carte de prototypage permettant d'envisager des applications complexes. Après l'établissement de la liaison entre les deux environnements, nous allons aborder la réalisation du système, la programmation de la carte à microcontrôleur, la conception des différents blocs de simulation ainsi que les fonctions nécessaires dans l'environnement Simulink afin de contrôler ou de commander notre système a base des correcteurs et des valeurs de consignes. Cette partie sera développée dans les prochains chapitres.

I)-Introduction :

Toutes commandes ou régulations est dans le but de maitriser l'évolution d'une ou plusieurs grandeurs physique (température, vitesse…etc.) à partir d'une ou plusieurs variables de contrôle dans un environnement perturbé.

Dans ce chapitre nous allons entamer l'étude des commandes automatiques utilisées dans le domaine industriel et en particulier l'agriculture ainsi leurs objectifs. Les notions de base de la régulation d'un système sont données .Il s'agit de synthétiser les contrôleurs qui seront ajustées à l'aide des techniques d'identification.

Les techniques de régulation que nous allons voir peuvent être regroupées en deux catégories :

Les techniques mettant en œuvre l'association des boucles de régulation.

Les techniques utilisant les procédés et les algorithmes modernes de commande.

II)-La régulation industrielle :

II .1)-Définition d'un système :

Un système est un ensemble d'appareils utilisé pour obtenir un produit déterminé. L'évolution de ce dernier dépend d'une ou plusieurs grandeurs incidentes, d'ailleurs il est caractérisé à l'aide d'une ou plusieurs grandeurs physiques mesurables et maîtrisables qui vont permettre de contrôler l'objectif fixé, comme il peut être simple ou complexe.

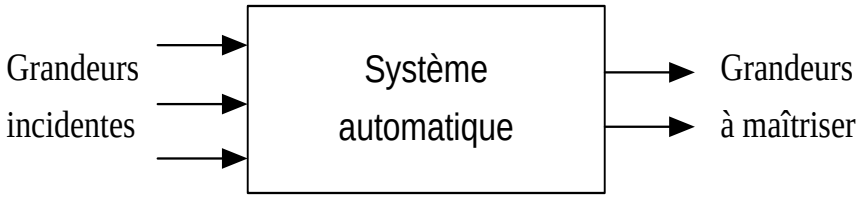


Figure III.1 : Système automatique.

Afin de pouvoir contrôler (régler, asservir) un système, il est nécessaire de connaître un certain nombre de ses propriétés:

• nombre et nature des entrées et des sorties.

• comportement statique et dynamique (temps de montée, nombre et période des oscillations)

• linéarité ou non-linéarités.

• stabilité.

•etc.

II.2)-Identification d'un système :

La première démarche à suivre lors du contrôle d'un système physique est l'application d'une méthode d'identification qui permet d'obtenir un modèle mathématique liant les variables du processus et leur actionneur ainsi définir la particularité du système

II.2.1)-Identification dans le domaine temporel :

Cette technique consiste à attaquer le système par un échelon d'amplitude donnée et s'intéresser par la suite à l'évolution dans le temps de sa sortie. Nous comparons ensuite la réponse obtenue avec une fonction de transfert normalisée.

Il faut donc utiliser une méthode d'identification qui ne demande aucune connaissance préalable si ce n'est l'enregistrement de la réponse indicielle. Le modèle que l'on obtiendra alors sera un modèle de représentation qui peut n'avoir aucune correspondance physique avec le système considéré, mais si l'identification est bonne, les réponses indicielles ou fréquentielle seront aussi proches que possible de celles du système étudié.

II.3)-Objectif de la régulation automatique :

La régulation industrielle ou automatique est une des principales caractéristiques de l'industrie moderne. Elle regroupe l'ensemble des moyens matériels et techniques mis en œuvre pour maintenir une grandeur physique à réguler, égale à une valeur désirée appelée consigne. Le fonctionnement de l'installation ne requiert que certaines grandeurs physiques y 1(t), y2(t), . . . d'un système aient un comportement fixé par les consignes c 1(t), c2(t), . . ., malgré la présence de perturbations p 1(t), p2(t), . . . d'origine externe et imprévisibles.

A cet effet, les grandeurs physiques sont mesurées, traitées puis une action correctrice est entreprise sur le système au moyen des commandes u1(t), u2(t).

Pour cela, le but de la régulation est d'ajuster la (puissance) à apporter en fonction des besoins, autrement dit (pour maintenir la consigne, il faut compenser les perturbations).

Suivant les procédés et les objectifs à réaliser, il existe une grande variété de matériels et de techniques utilisés en régulation ou en commande.

La figure II.2 montre une structure d'un système de régulation automatique (mimo).

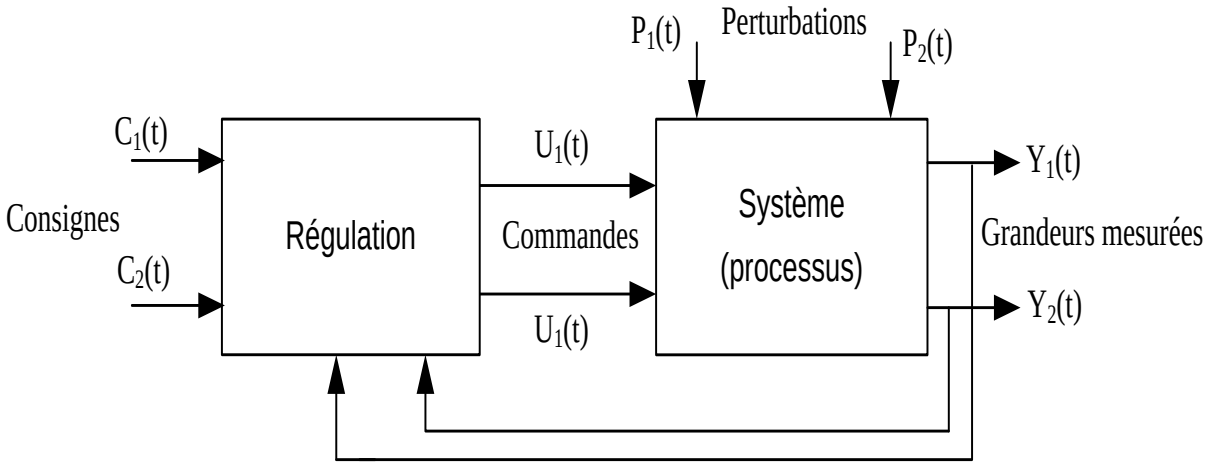


Figure III.2 : Structure d'un système de régulation (mimo).

II.4)-Chaines de régulations :

II.4.1)-Régulation en boucle ouverte :

Une régulation est dite en boucle ouverte lorsque les signaux d'entrées e(t) ne sont pas influencés par les signaux de sorties s(t). Dans ce cas l'action ne modifie pas la grandeur observée (la grandeur à maîtriser). La mise en œuvre d'une telle régulation nécessite la connaissance des lois régissant le fonctionnement du processus, c'est-à-dire la corrélation entre la valeur mesurée et la grandeur régulant.

La figure III.3 illustre la structure d'une commande en boucle ouverte.

L'inconvénient majeur est que l'objectif fixé n'est généralement pas atteint complètement, (réside dans ses limites).

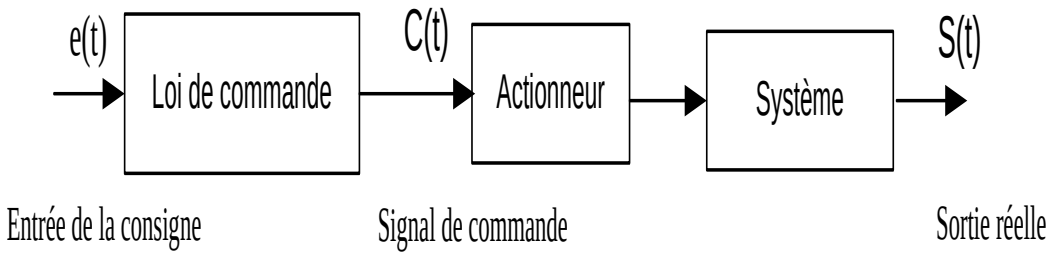


Figure III.3 : Commande en boucle ouverte.

II.4.2)-Régulation en boucle fermée :

La grandeur réglant exerce une influence sur la grandeur réglée pour la maintenir dans des limites définies malgré les perturbations.

Cette chaîne de régulation est dite fermée car l'action modifie la grandeur observée.
Le principe de commande en boucle fermée est illustré sur la figure III.4 et définit la structure de commande.

L'avantage d'une chaîne fermée est qu'une variation de la grandeur observée (la grandeur à maîtriser) entraîne une variation de l'action donc l'objectif fixé peut alors être atteint.

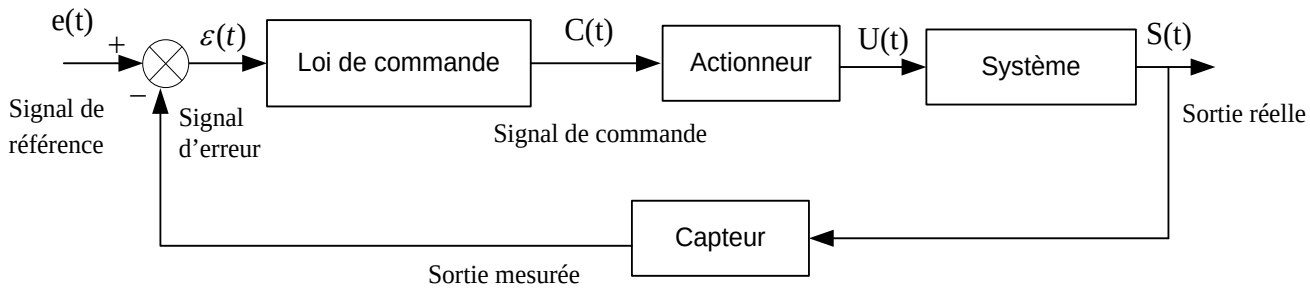


Figure III.4 : Commande en boucle fermée.

II.5)-Eléments et signaux caractéristiques d'un système de régulation :

II.5.1)-Constitution d'une régulation :

a)-Partie commande :

Le régulateur est composé d'un comparateur qui détermine l'écart (ou l'erreur) entre la consigne et la mesure et d'un correcteur, qui élabore à partir du signal d'erreur l'ordre de commande.

b)-Actionneur:

C'est l'organe d'action qui apporte l'énergie au système pour produire l'effet souhaité. Il est en général associé à un pré-actionneur qui permet d'adapter l'ordre (basse puissance) et l'énergie.

c)-Capteur:

Le capteur prélève sur le système la grandeur réglée (information physique) et la transforme en un signal compréhensible par le régulateur. La précision et la rapidité sont deux caractéristiques importantes du capteur.

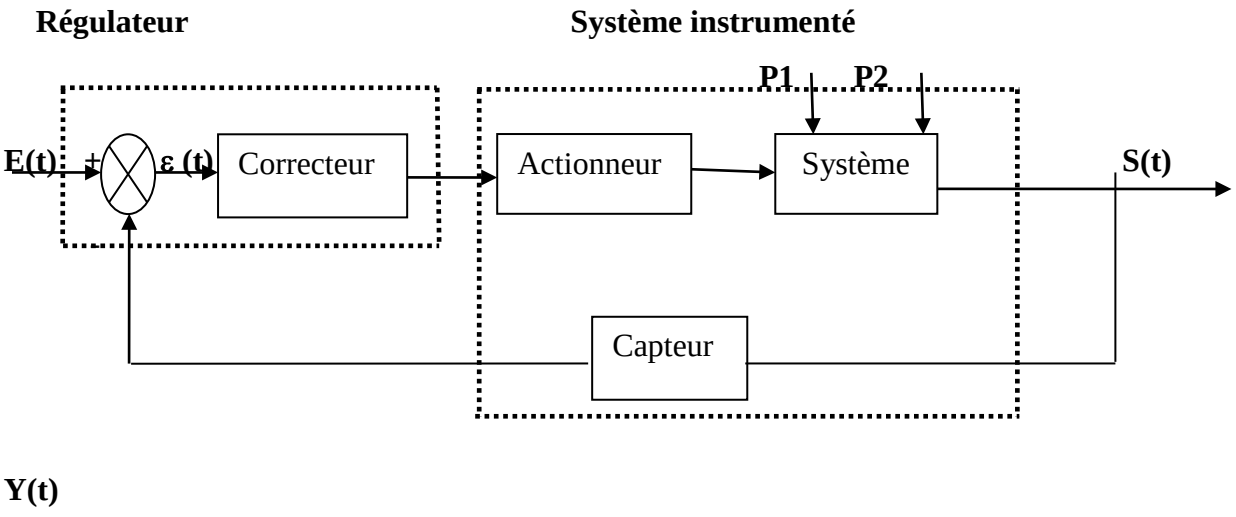


Figure III.5 : constitution d'une chaine fermée.

-La consigne noté e(t) : est la valeur que l'on désire atteindre, c'est la grandeur régulant du système.

-La mesure de la grandeur à maîtriser noté s(t) : représente le phénomène physique que doit réguler le système.

-L'écart (ε) ou l'erreur entre la consigne et la mesure est représenté par l'équation III .1 :

$$\varepsilon = e - S \qquad \text{(III.1)}$$

-P1 et P2 grandeurs non contrôlées sont les grandeurs perturbatrices appelées perturbations.

Dans l'étude d'un système de régulation, le système est défini par sa fonction de transfert en utilisant la transformée de (Laplace) pour les systèmes analogiques et la transformée en (z) pour les systèmes numériques.

II.5.2)-Différents types des signaux d'une régulation :

Les signaux intervenant dans le schéma général d'un système de régulation sont résumés dans le tableau III.1.

Signal	Notation	Signification
consigne	e(t)	Signal à poursuivre, à caractère généralement déterministe, par opposition à aléatoire: ce signal est défini pour une application donnée.
Grandeur réglée brute	S(t)	Grandeur physique réglée. Seule une image peut en être obtenue par l'intermédiaire d'un capteur.
Grandeur réglée mesurée	Y(t)	C'est la seule information dont dispose le régulateur, lequel asservit donc en réalité la grandeur réglée mesurée
commande	U(t)	Signal délivré par le régulateur au système à régler.
perturbation	P(t)	Signal aléatoire représentant les perturbations intervenant sur le système à régler.
Ecart	ε(t)	C'est la différence entre la consigne et la grandeur réglée mesurée. ε(t) = c(t) – y(t)

Tableau III.1 : Signaux de la régulation.

II.6)-Contraintes d'exploitation des systèmes de régulation :

Parmi les contraintes liées à l'exploitation des systèmes de régulation nous avons la stabilité, rapidité et la précision.

a)-Stabilité :

Un système est dit stable si pour une entrée constante, la sortie reste constante quelles que soient les perturbations.

Il existe plusieurs méthodes pour déterminer la stabilité d'un système, D'où nous pouvons distinguer les critères analytiques (règle de Routh,…) et des critères graphiques.

La Figure III.6 désigne un ensemble de systèmes stables et instables.

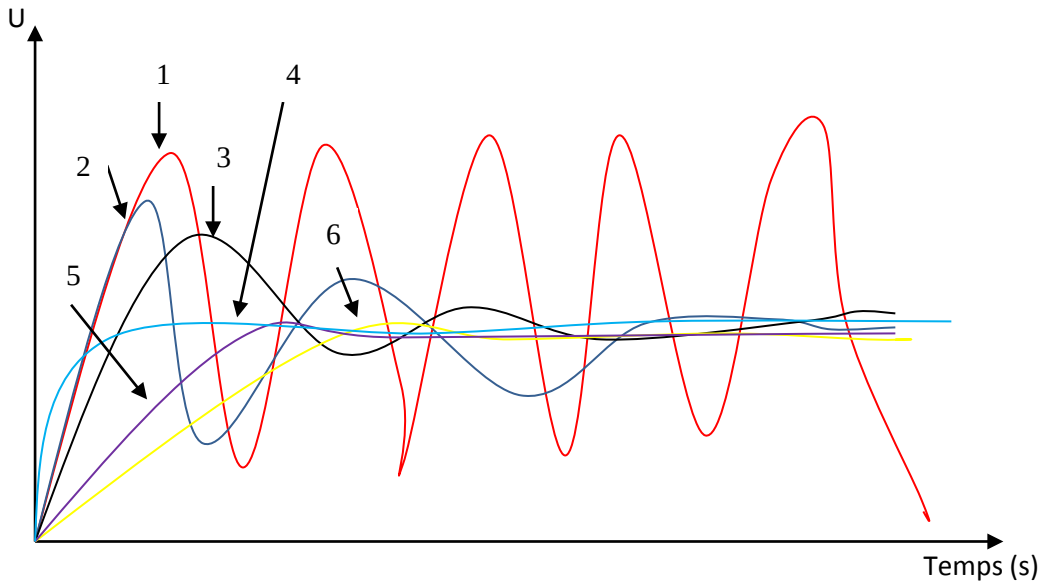


Figure III.6 : Stabilité d'un système.

Les courbes de (4 à 6) sont caractéristiques de la réponse d'un système stable, pour une entrée constante, la sortie évolue vers une sortie constante, Contrairement aux autres courbes caractérisant des systèmes instables.

b)-Rapidité :

La rapidité caractérise le temps mis par le système pour que la sortie atteigne sa nouvelle valeur. Pour caractériser la rapidité, nous définissons le temps de réponse à 5% ou (t5%), comme suivant(le temps que met le système pour rester la bande des 5% de sa valeur finale).

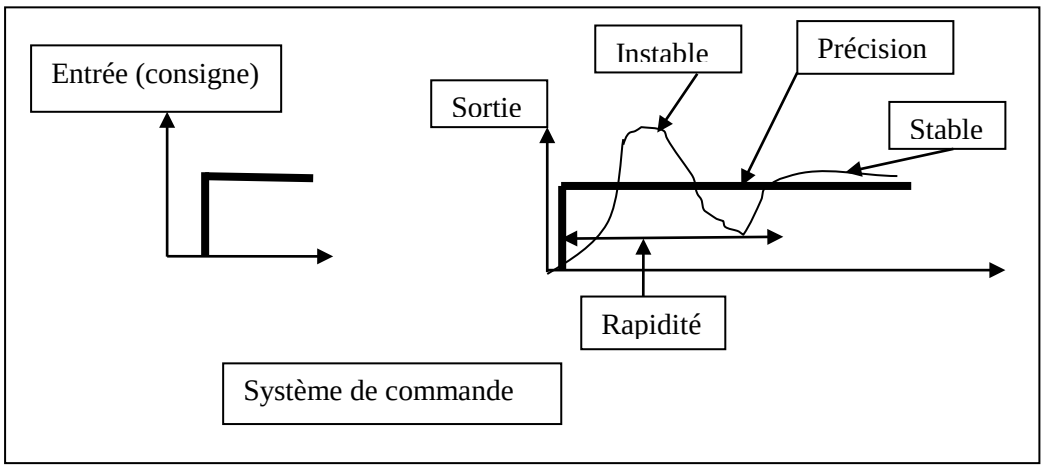


Figure III.7 : Représentation des performances d'un système de régulation.

c)-Précision :

La précision d'un système est définie par son signal d'erreur ε(t) selon lequel le système se trouve en régime statique(c(t) =constante (régime permanent constant)) ou en régime dynamique (c(t) = f(t) (régime permanent variable)), on parle de précision statique ou de précision dynamique.

Pour tout système, nous essaierons alors de minimiser ces erreurs.

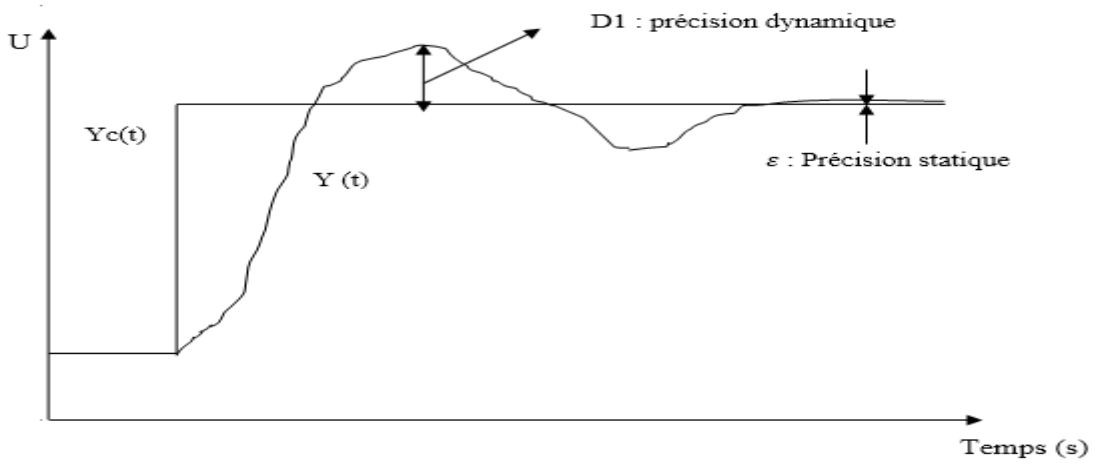


Figure III.8 : Réponse à un échelon de consigne.

III)-Régulation numérique :

La vulgarisation de l'électronique numérique fait que la totalité des commandes sont faite à l'aide de microcontrôleurs, calculateur…….

Le traitement numérique de l'information à des instants discrétisés implique l'utilisation d'outils telle la transformation en (z) ou les variables d'état discrètes.

Les avantages des régulateurs numériques sont :

-Leur stabilité.

-Leur souplesse dans la programmation des algorithmes de commande.

L'application des calculateurs numériques utilisés en temps réel pour commander des systèmes physiques qui par essence sont le plus souvent continus a donné naissance aux systèmes commandés échantillonnés (discrets/numériques).

La commande par calculateur, ou processeur, d'un système nécessite la mise en œuvre d'un certain nombre d'éléments :

-Un processeur (calculateur) qui élabore la commande U(k) et réalise l'échantillonnage,

-Des convertisseurs analogique-numériques, CAN.

-·Des capteurs ou organes de mesure qui transmettent au calculateur les informations recueillies sur le système continu, à travers les convertisseurs analogiques numériques,

-Des convertisseurs numérique-analogiques, CNA.

La figure III.9 illustre une structure de commande d'un système continu.

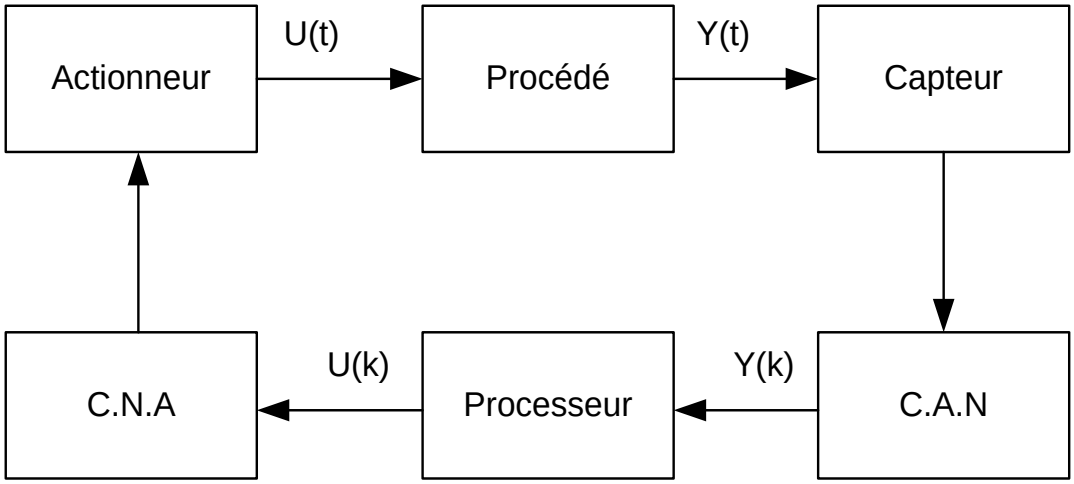


Figure III.9 : Structure de commande d'un système continu par calculateur.

III.1)-Problèmes à résoudre pour le contrôle des processus continus :

a)-Echantillonnage d'un signal continu :

Cette opération consiste à relever les informations prise par un signal continu à l'intervalle de temps régulier, appelé période d'échantillonnage. Le calculateur ne tiendra compte des valeurs prise par le signal aux instants d'échantillonnage.

b)-Passage correcteur analogique-correcteur numérique :

Il s'agit de convertir la valeur prise par un signal analogique à l'instant d'échantillonnage en une valeur numérique pour qu'elle soit traitée par le calculateur.

c)-Passage correcteur numérique- correcteur analogique :

Cette conversation consiste à transformer le signal numérique de commande du calculateur en signal analogique de commande correspondant, l'objectif étant de commander le système physique.

IV)-Régulation tout ou rien(TOR) :

La plus simple des techniques de contrôle est la régulation Tout Ou Rien (TOR). Elle est utilisée quand la dynamique du système est très lente (grande constante de temps).

Cette régulation est acceptable pour les systèmes thermiques stables de faible puissance ou de forte inertie thermique.

Cette méthode de régulation est considérée comme une régulation discontinue car la commande envoyée aux actionneurs varie instantanément.

La réalisation de cette technique impose de se fixer une limite inférieure et une limite supérieure de la grandeur réglée.

V)-Régulation avec PID :

V.1)-principe de régulateur PID :

Le régulateur PID forme un signal de commande (grandeur de réglage) qui va faire varier la puissance de réglage par l'intermédiaire d'un actionneur (organe de réglage).

Un régulateur PID remplit essentiellement trois fonctions :

1. Il fournit un signal de commande u(t) en tenant compte de l'évolution du signal de sortie y(t) par rapport à la consigne w(t).
2. Il élimine l'erreur statique grâce au terme intégrateur.
3. Il anticipe les variations de la sortie grâce au terme dérivateur.

Il existe trois types de régulation PID, le PID série, le PID parallèle et le PID mixte.

P = KR : est l'action proportionnelle, sur la plupart des régulateurs, on règle la

Bande Proportionnelle au lieu de régler le gain du régulateur :

I = 1/Ti (min-1 en général) : est l'action intégrale

D = Td (s en général) : est l'action dérivée **[7].**

V.2)-Les types de régulation PID :

a)-Structure parallèle :

Cette structure est illustrée par la figure III.10.

Dans ce cas la sortie y(t) est donnée par l'équation III.2:

$$Y(t)=K_p .e(t)+\frac{1}{Ti} .\int_0^t e(\tau) .d\tau +T_d.\frac{de(t)}{dt} \qquad (III.2)$$

Considérons que les conditions initiales sont nulles et en appliquant la transformée de Laplace à l'équation III .2, nous obtenons la fonction de transfert du régulateur PID à structure parallèle donnée par l'équation III.3.

$$C(p) = \frac{Y(p)}{E(p)} = K_p + \frac{1}{T_i.p} + T_d.p \qquad \text{(III.3)}$$

Où : $T_i = \frac{1}{K_i}$ et $T_d = \frac{1}{K_d}$.

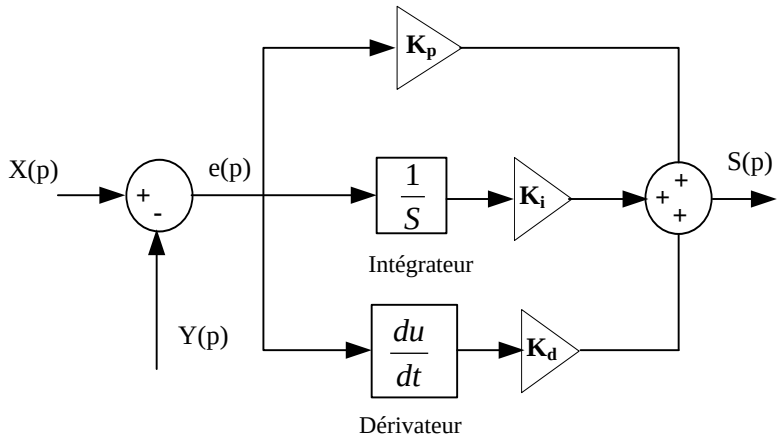


Figure III.10 : Régulateur PID à structure parallèle.

b)-Structure série:

Cette structure est illustrée par la figure III.11. Dans ce cas la sortie Y(t) est donnée par l'équation III.4:

$$Y(t) = \alpha.K_p.e(t) + \frac{K_p}{T_i} + \int_0^t e(\tau)d\tau + T_d.\frac{de(t)}{dt} \qquad \text{(III.4)}$$

Où : $\alpha = \frac{T_i + T_d}{T_i}$ le coefficient théorique d'interaction entre action intégrale et action dérivée.

Dans le cas où les conditions initiales sont nulles, en appliquant la transformée de Laplace à l'équation III.4 nous obtenons la fonction du régulateur PID à structure série donné par l'équation III.5.

$$C(p) = \frac{Y(p)}{E(p)} = K_p(1 + \frac{1}{T_{i.}p})(1 + Td.p) \quad \text{(III.5)}$$

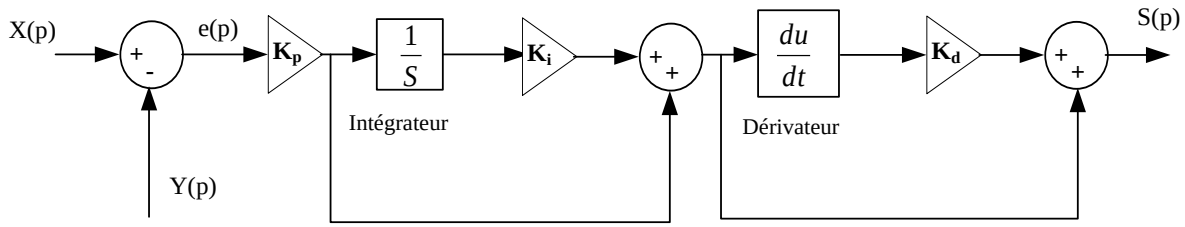


Figure III.11: Structure série.

c)-Structure mixte:

Cette structure est illustrée par la figure III.12 et c'est la plus utilisée actuellement par les constructeurs.

L'expression de la sortie du régulateur PID est donnée par l'équation III.6:

$$Y(t) = K_p.e(t) + \frac{K_p}{T_i}\int_0^t e(\tau)d\tau + k_p.T_d.\frac{de(t)}{dt} \quad \text{(III.6)}$$

Par application de la transformée de Laplace, nous obtenons :

$$C(p) = \frac{Y(p)}{E(p)} = K_P\left(1 + \frac{1}{Ti.p} + T_d.p\right) \quad \text{(III.7)}$$

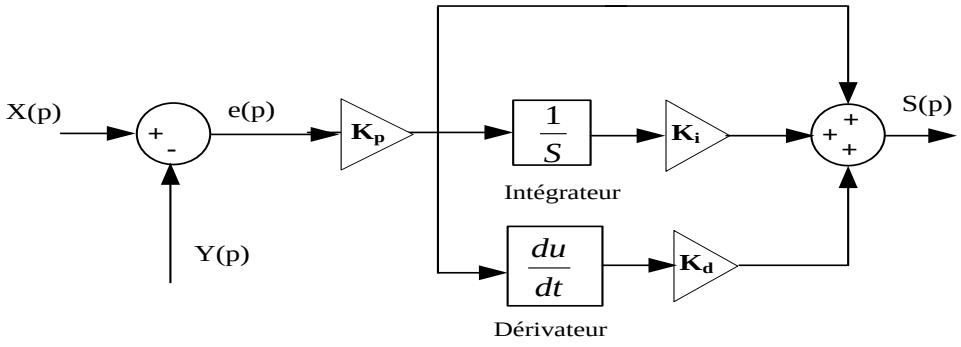


Figure III.12 : Structure mixte.

V.3)-Actions PID :

a)-Action proportionnelle :

Le régulateur à action proportionnelle ou régulateur P, a une action simple et naturelle puisqu'il construit une commande u(t) proportionnelle à l'erreur ε(t).

Les relations suivantes III.8, III.9. Nous donnent respectivement la loi de commande et la fonction de transfert de l'action proportionnelle.

$$U(t)=K_p.\ \varepsilon(t) \qquad \text{(III.8)}$$

Et pour le cas discret, cette relation reste la même :

$$U(k)=K_p.\ \varepsilon(k) \qquad \text{(III .9)}$$

Le rôle de l'action proportionnelle est d'accélérer la réponse de la mesure, et minimiser l'écart ε(t) entre la consigne et la mesure, elle réduit le temps de monter et le temps de réponse.

b-Action dérivée :

Cette action tient compte des variations de la grandeur de sortie. Elle délivre une sortie variant proportionnellement à la dérivée de l'écart(ε) :

L'action dérivée exprimé par l'équation III.10 est réglée par la constante de temps d'action dérivée, notée (*T*d) et exprimée en seconde.

$$U(t)=T_d\frac{d\varepsilon(t)}{dt} \qquad \text{(III.10)}$$

L'action dérivée sur la mesure n'a aucune influence sur la sortie (*Y)* lors d'un changement de consigne, et évite ainsi des à-coups inutiles et néfastes sur l'actionneur. L'action dérivée est obligatoirement associée à l'action proportionnelle.

Note :

En pratique, il est souhaitable de limiter l'action dérivée afin de ne pas amplifier les bruits haute fréquence et de limiter l'amplitude des impulsions dues aux discontinuités de l'écart.

c)-Action intégrale :

L'action intégrale exprimé par l'équation III.11 est réglée, au choix du fabricant, soit par :
– la constante de temps d'action intégrale, notée *T*i, exprimée très souvent en minute ;
– le coefficient ou taux d'action intégrale *K*i, exprimé en min–1 :

$$K_i = \frac{1}{T_i} \qquad \text{(III.11)}$$

Le correcteur intégral est en général associe au correcteur proportionnel, il élabore alors une commande qui peut être donnée par la relation III.12 **[10]:**

$$U(t)=\frac{1}{Ti}\int_0^t \varepsilon(t).d\tau \qquad \text{(III. 12)}$$

VI)-Régulation par Logique flou :

VI.1)-Historique :

La logique floue est une extension de la logique booléenne créée par Lot_ Zadeh en 1965 **[9]** en se basant sur sa théorie mathématique des ensembles flous, qui est une généralisation de la théorie des ensembles classiques et en introduisant la notion de degré dans la vérification d'une condition, permettant ainsi a une condition d'être dans un autre état que vrai ou faux .
Voici par exemple quelques règles inspiré d'une logique flou de conduite qu'un conducteur suit :

Si le feu est rouge…	Si ma vitesse est élevée…	et si le feu est proche...	alors je freine fort.
Si le feu est rouge…	si ma vitesse est faible...	et si le feu est loin...	alors je maintiens ma vitesse.

Tableau III.2 : Règles et conditions flou.

VI.2)-Ensembles et Systèmes flou :

Soit (X) un ensemble, un sous-ensemble flou (A) de(X) est caractérisé par une fonction d'appartenance. Cette notation veut simplement dire que quel que soit l'entrée (X) donnée à la

fonction sa sortie est un réel entre 0 et 1.

En théorie, il est possible que la sortie soit supérieure à 1, mais en pratique cela n'est quasiment jamais utilisé.

Note :

cette fonction d'appartenance est l'équivalent de la fonction caractéristique d'un ensemble classique.

La figure III.13 illustre une représentation d'un système flou **[8].**

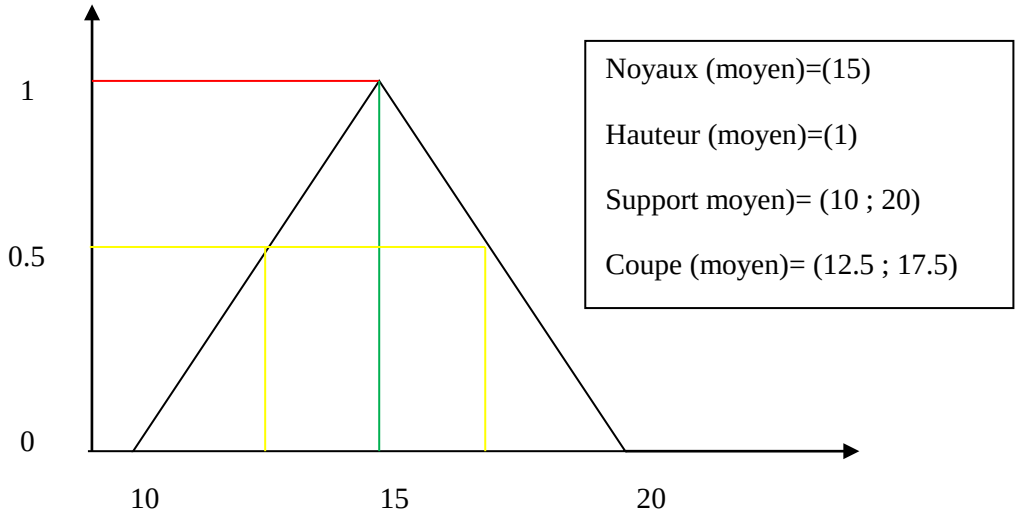


Figure III.13 : Représentation d'un ensemble flou.

Un système à logique floue comporte : des variables d'entrée sortie, des labels qui représentent les valeurs floues de chaque variable, des fonctions qui définissent le degré d'appartenance des valeurs des variables aux labels.

Une valeur mesurée peut appartenir à plusieurs labels, avec des degrés divers.

Le tableau III.3 illustre le degré de vérité avec signification.

Logique flou
0.0 Absolument faux
0.2 Plutôt faux
0.4 Quelque peu faux
0.6 Quelque peu vrai
0.8 Plutôt vrai
1.0 Absolument vrai

logique classique
0 Absolument faux
1 Absolument vrai

Tableau III.3 : Variation du degré d'appartenance.

La figure III.14 illustre une variation de température par des labels flous.

Pour T=6 C et T=30C et T=40C nous avons :

froid (6) =0.6, tiède(6) =0.2, chaud(6)=0 froid(30)= 0, tiède(30)=0.4chaud(40)=0.8 …etc.

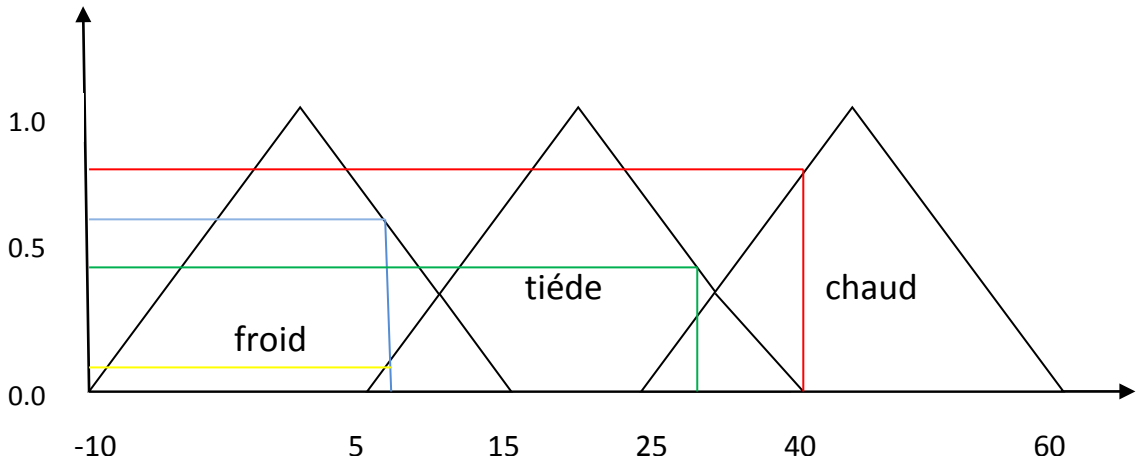


Figure III.14 : Représentation de la température par les ensembles flous.

VI.3)-Variables linguistique :

Une variable linguistique appelée aussi attribut linguistique peut être définie à partir du triplet (x, Uf, Tx) où (x) est une variable définie sur l'univers de discours (Uf).

Et (Tx) = A1, A2,… est un ensemble composé de sous-ensembles flous de (Uf) qui caractérise (x).

A chaque sous ensemble flou de (TX) nous associons toujours une valeur ou un terme linguistique (étiquette) **[9]**.

La figure III.14 illustre un exemple des variables avec trois termes linguistiques: froid, tiède et chaud.

VI.4)-Propriétés d'un ensemble flou :

Les suivis sont les principales composantes d'un system flou,

a)- Fuzzifier : Le rôle de fuzzifier est de convertir les valeurs d'entrée croquantes en valeurs floues.

b)-Base de connaissances floue : Elle stocke les connaissances sur toutes les relations floues d'entrée-sortie. Il a également la fonction d'appartenance qui définit les variables d'entrée à la base de règles floue et les variables de sortie à l'usine sous contrôle.

c)-Fuzzy Rule Base: Il stocke les connaissances sur le fonctionnement du processus de domaine.

d)-Inference Engine : Il agit comme un noyau de tout FLC. Fondamentalement, il simule les décisions humaines en effectuant un raisonnement approximatif.

e)-Defuzzifier : Dernière étape de la logique floue, elle a pour objectif de transformer la courbe d'activation finale obtenue lors de l'étape d'agrégation en une valeur réelle.

Deux méthodes sont alors applicables pour obtenir la valeur retenue de la variable à prédire :

e.1)-La méthode de la moyenne des maxima :

Cette méthode correspond à la moyenne des valeurs de sortie les plus vraisemblables et elle est définit par l'équation III.13.

$$MM : xMM = (\int s\,(xdx) / \int s\,(dx)) = \sum_{i=0}^{n}(xi)\,/N \qquad \text{(III .13)}$$

Où : $(S) = \{x \in U\ tel\ que\ (x) = SUPx \in (\mu\,(x))\}$

(N) = nombre de point appartenant à S

(x) = la variable

La figure III.15 montre une Défuzzification avec la méthode moyenne des maxima (MM).

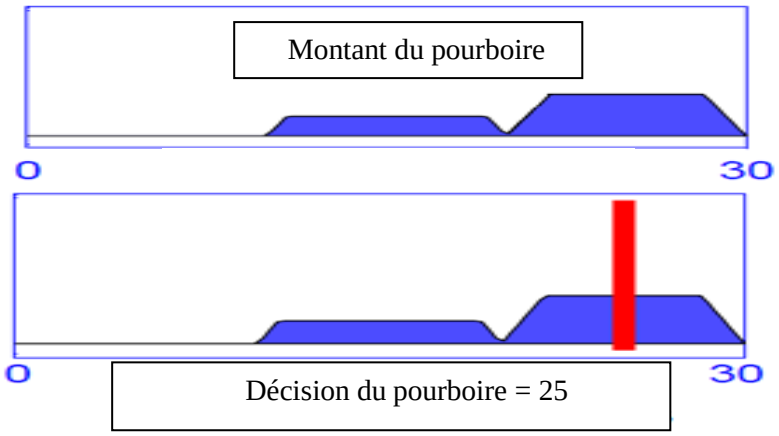


Figure III.15: Défuzzification avec la méthode moyenne des maxima (MM).

e.2)-La méthode des centres de gravité :

Cette méthode définit par l'équation III.14 correspond aux abscisses du centre de gravité de la surface de la courbe de résultats.

D'après le mathématicien cette méthode semble préférable et plus cohérente avec les principes de la logique floue.

$$COG : xG = (\int u\ x\mu x dx) / (\int u\ \mu x dx) = \sum_{i=0}^{n}(xi\mu xi) / \sum_{i=0}^{n}(\mu xi) \qquad \text{(III.14)}$$

Où (U) est l'univers du discours de la variable de sortie.

La figure III.16 montre une Défuzzification avec la méthode des centres de gravité (COG).

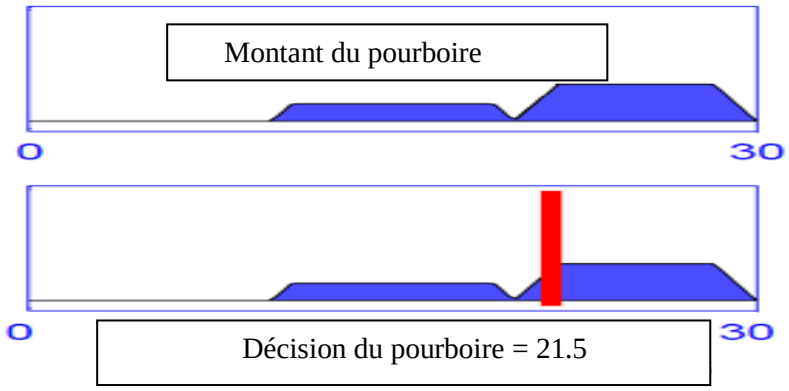


Figure III.16 : Défuzzification avec la méthode des centres de gravité (COG).

La figure III.17 illustre un aperçu d'un système flou.

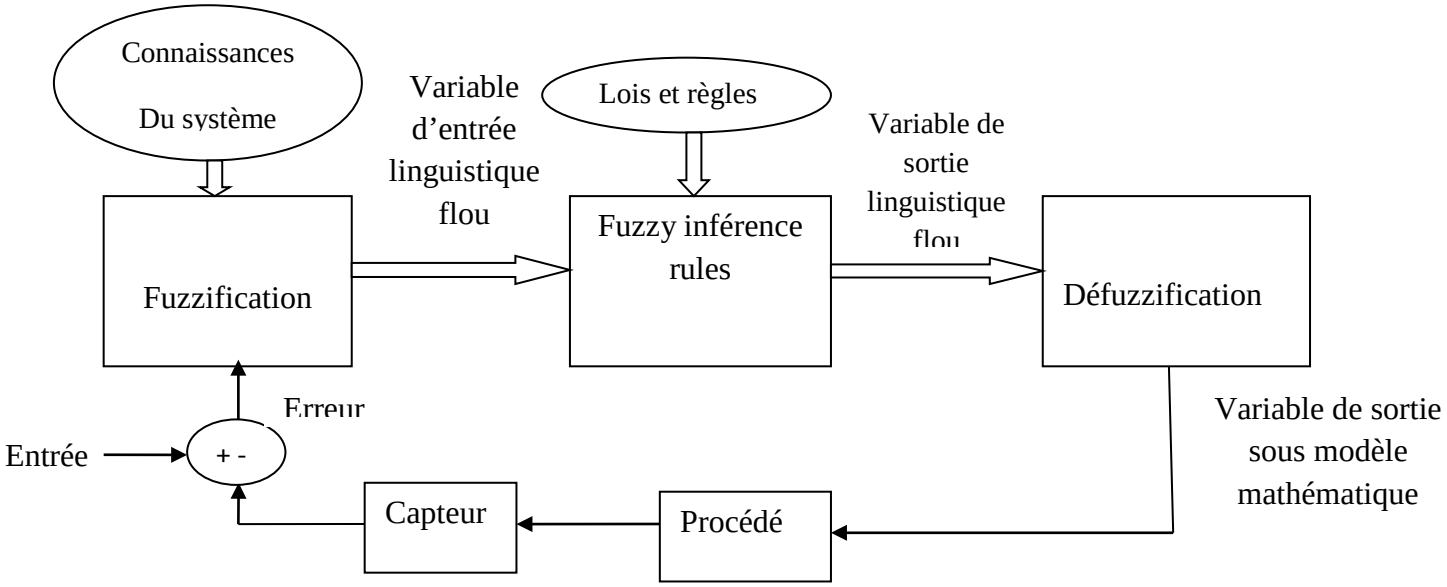


Figure III.17 : Représentation d'un système flou.

VI.5)-Opérateurs de la logique floue :

a)-Opérateur NON (complément, négation, inverse) :C'est l'ensemble complémentaire de la théorie des ensembles.

Nous pouvons l'exprimer par les fonctions d'appartenance de la manière suivante :

$$\text{NON } a = 1 - a \quad ; \quad c = a = \text{NON}(a) \qquad \text{(III.15)}$$

b)- Operateur OU (maximum) :C'est la réunion de deux ensembles (a et b), et nous pouvons le définir par l'équation III.16 :

$$a\ OU\ b = \gamma.\max(a,b) + (a+b)/2 \quad ; \quad c = a \cup b \qquad \text{(III.16)}$$

L'opérateur maximum est commutatif et associatif.

c) - Operateur ET (minimum) : il correspond à l'intersection de deux ensembles (a et b), et nous pouvons le définir par l'équation III.17 :

$$a\ ET\ b = \gamma.\min(a,b) + (1-\gamma).(a+b)/2 \quad ; c= a \cap b \qquad (III.17)$$

L'opérateur minimum est commutatif et associatif.

Ou (a, b, c) des ensembles flou et (γ) le facteur d'appartenance.

La figure III.18 suivante montre un exemple d'emploi des opérateurs sur des labels flou.

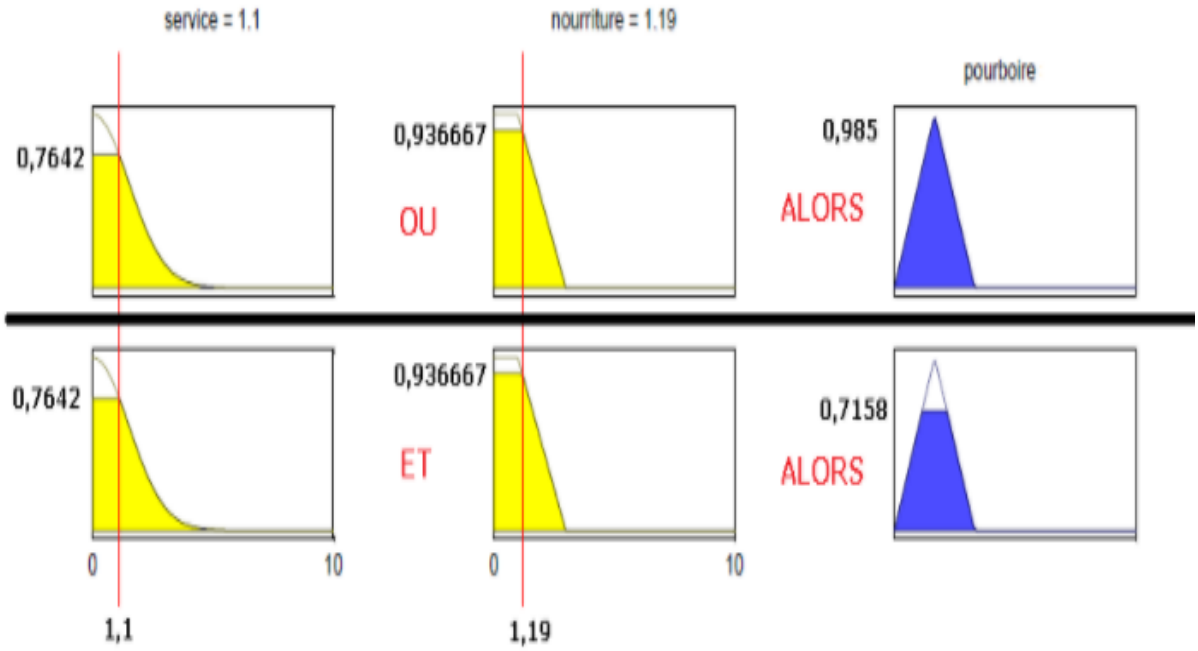


Figure III.18: Opérateurs flou.

VII)-Conclusion :

A travers ce chapitre nous avons cite les différentes composantes d'une étude d'un système ainsi que les éléments caractérisant ce dernier, comme nous avons abordé une brève description sur les commande a utilisés (PID, TOR, flou) soit en boucle ouverte ou fermée afin d'entamer une étude théorique qui nous permettras d'associer une meilleur régulation à notre propre système .cette partie feras l'objet du prochain chapitre.

I)-Introduction :

Dans ce chapitre nous allons développer notre système de régulation .A cet effet, quelques techniques d'identification sont employées, ces dernières jouent un rôle important sur la correction des lois de commande.

Une fois le système identifié on va utiliser les blocks PID et la Toolbox fuzzy logic de Matlab qui vont nous permettre d'établir des lois de commande et des règles d'inférences floues et de simuler les valeurs de sorties selon les valeurs d'entrées du système pour visualiser l'évolution des paramètres avec et sans régulation.

Par la suite, nous présentons les schémas de réalisation et les tests effectués sur le système réalisé. A cet effet, on va exploiter les fonctions offertes par la carte Arduino pour l'envoi de la commande et l'acquisition des données.

II)-Modélisation et identification du système :

Notre système est composé de quatre sous-systèmes rassemblés dans un circuit électronique qui permet de contrôler un moteur à courant continu, une résistance chauffante, une LED et un servomoteur à partir des sorties PWM de la carte Arduino.

L'entrée de chaque sous-systèmes est la tension U(s) en volts et les sorties sont : la température T(s) en degré Celsius, l'humidité H(s) du sol, la luminosité L(s) et la qualité de l'air C(s) quantifiées sur une échèle de 0 à 123.Ces valeurs sont récupérées à l'aide des capteurs.

Il existe plusieurs méthodes pour la modélisation et l'identification des systèmes comme la détermination des équations physiques, l'étude de la réponse d'un système à une entrée....etc. Dans notre cas nous utilisons les modes de dépouillement graphique de STREJC et BROIDA (voir annexe) qui sont des méthodes graphiques simples basées sur la réponse indicielle.

Nous limitons notre étude dans cette application au contrôle de température et l'humidité à l'intérieur de la chambre. Après la détermination des formes des fonctions de transferts, nous vérifions la validité et la précision de ces dernières.

Le modèle Simulink permettant l'acquisition de la réponse du système à des échelons de tension est représenté par la figure IV.1.

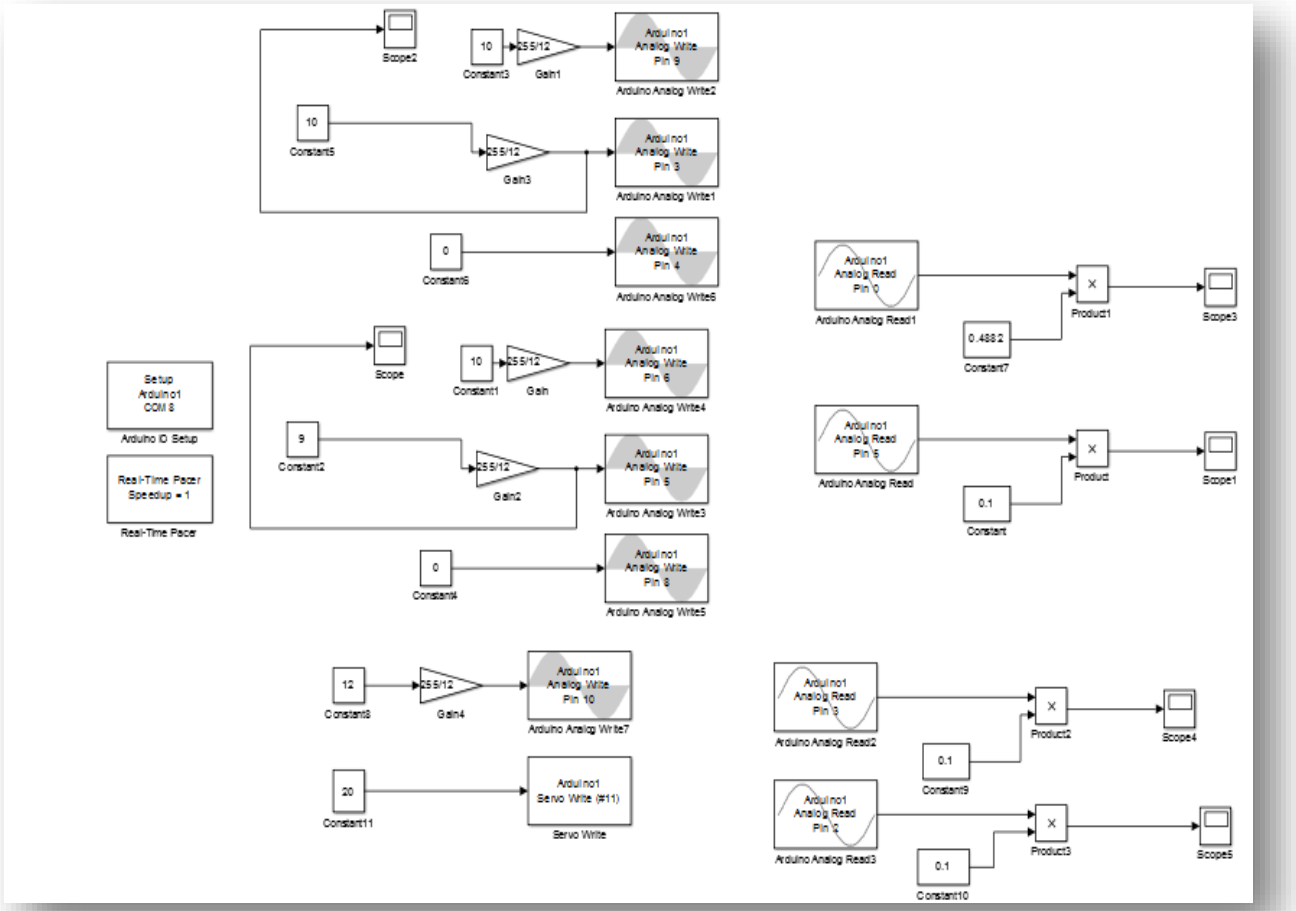


Figure IV.1: Schéma bloc pour la détermination de la réponse indicielle.

Après la simulation du système nous avons récupéré les allures IV.2, IV.3 suivantes :

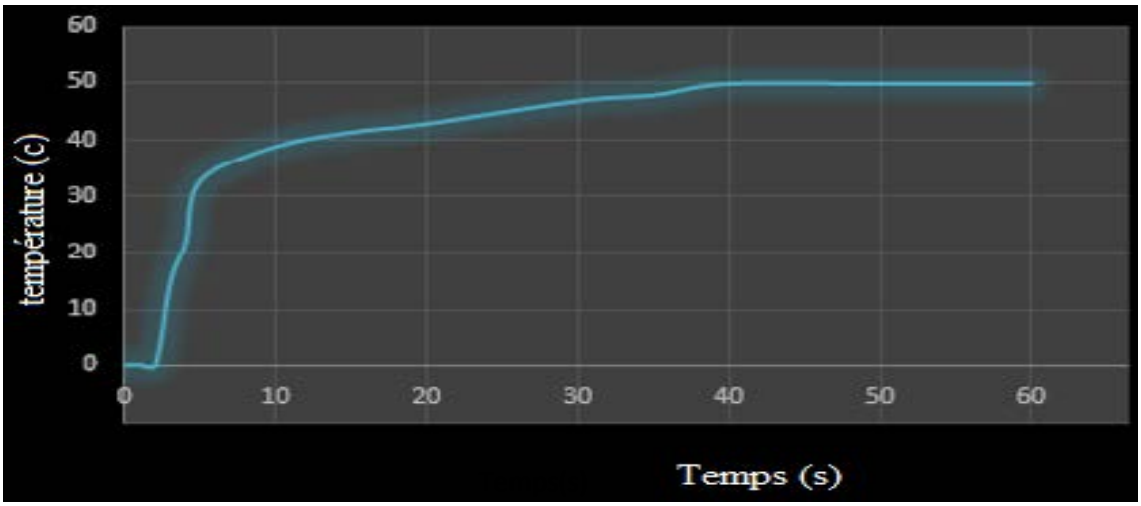


Figure IV.2 : Réponse du système à un échelon de tension.

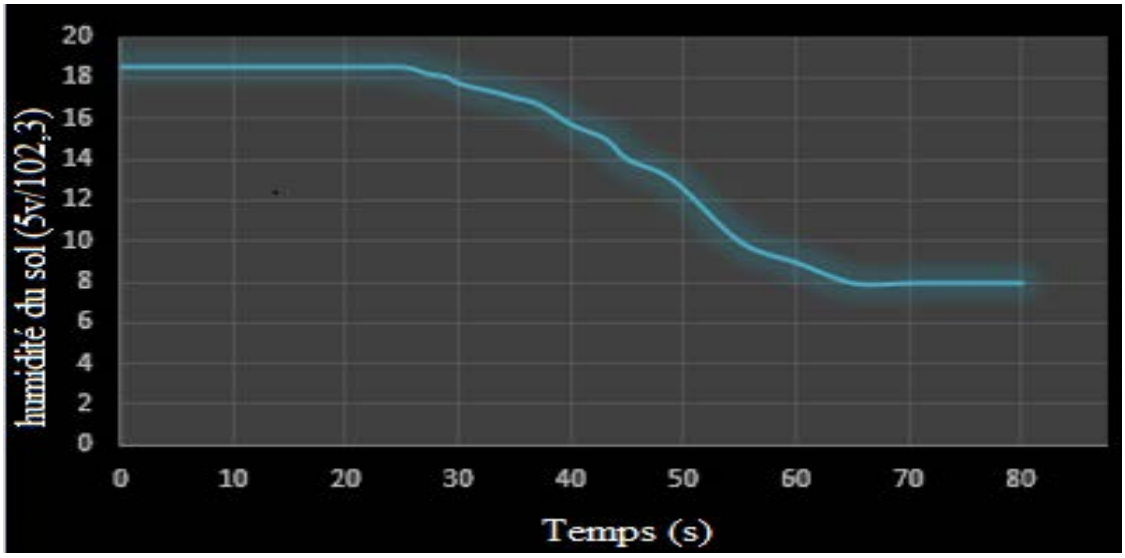


Figure IV.3 : Réponse du système à un échelon de tension.

L'examen des courbes obtenus après analyse permet à ces ensembles de leurs associer des fonctions de transferts de premier ordre pour la température de l'air et de deuxième ordre pour l'humidité du sol avec un léger retard.

Les fonctions de transferts sont exprimées par les équations IV.1, IV.2 :

$$G(p) = 4.17 \frac{e^{-1,25p}}{1+5.5p} \qquad \text{(IV.1)}$$

Avec :

t_1=3; t_2=4; θ=1.25

T=5.5

ΔE = 12,

ΔY = 50, d'où k = $\frac{\Delta Y}{\Delta E} = \frac{50}{12}$ =4,17

$$G(p) = -1.17. \frac{e^{-2.02.p}}{(1+10.67.p)^2} \qquad \text{(IV. 2)}$$

Avec :

K = -1.17 et au point d'inflexion (I) nous avons :

T1 = 5 et T2 = 29 ; $\frac{T1}{T2}$ = 0.17 ; n=2 ;

Et d'après la table de mesure (voir annexe tableau A1) nous avons :

$\frac{T2}{T}$ =2.72 ; T =10.67s ;

$\frac{T1}{T}$= 0.28 ; τ = 2.02 ;

Nous avons appliqué différents tests pour les deux systèmes identifiés (test de linéarité, stationnarité…etc.) (Voir annexeA2) ce qui nous mène à conclure qu'il s'agit de deux systèmes mono variables, stationnaire, déterministe, à paramètres localisés et non linéaire.

Note :

Il faut tout d'abord remarquer que les systèmes n'étant absolument pas linéaires, le dépouillement que nous avons fait ne nous fournira qu'une hypothèse de départ pour l'étude en boucle fermée.

Les essais effectués pour vérifier l'influence de l'amplitude de la perturbation sur l'allure des réponses n'ont pas permis de constater une variation notable dans les constantes de temps.

III)-Commande du système avec contrôleur flou et PID :

Les structures des blocs composant le système bouclé et corrigé avec des contrôleurs PID et flou sont illustrés par les figures IV.4 et IV.5**.**

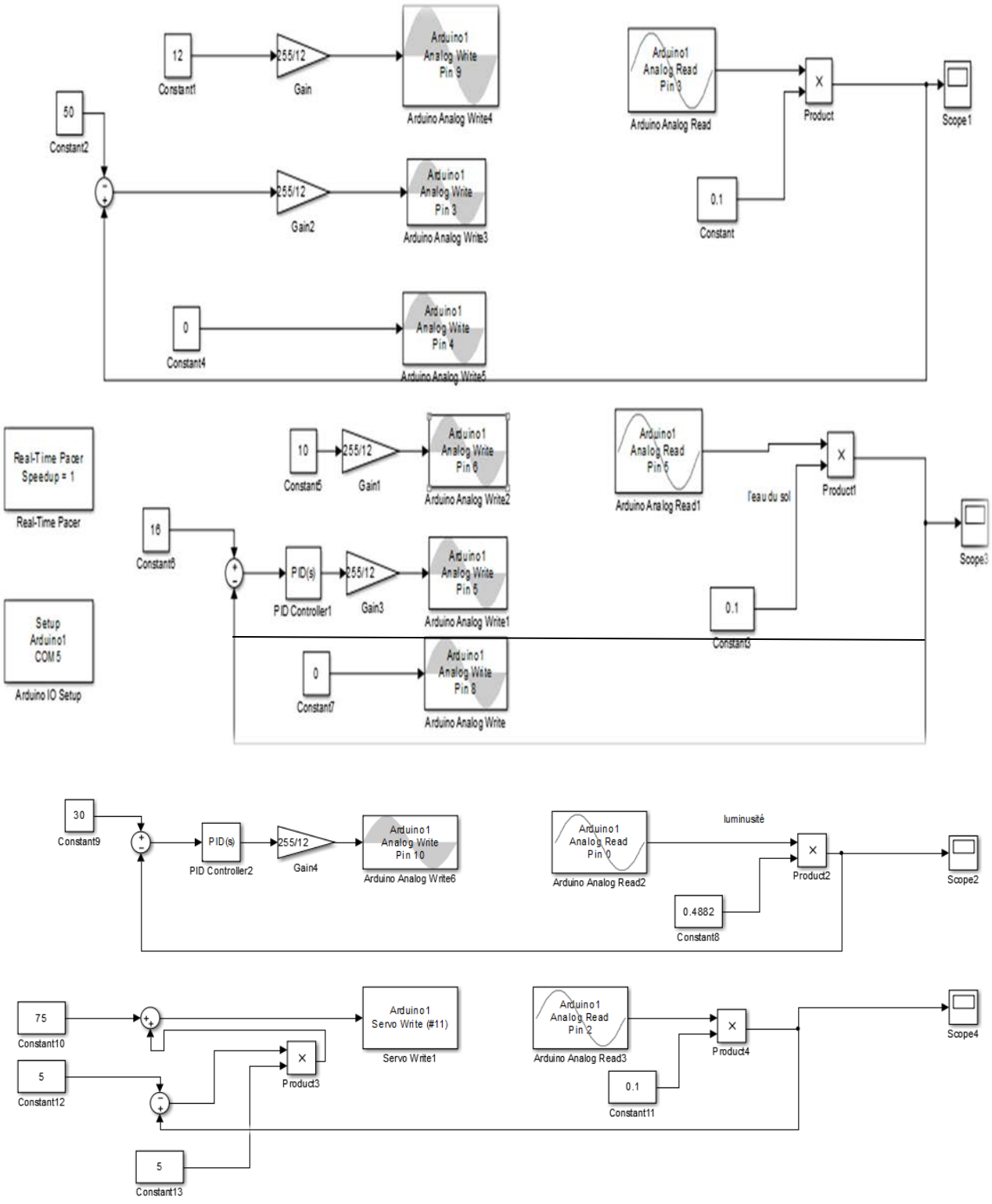


Figure IV.4 : Schéma bloc du système avec le contrôleur PID.

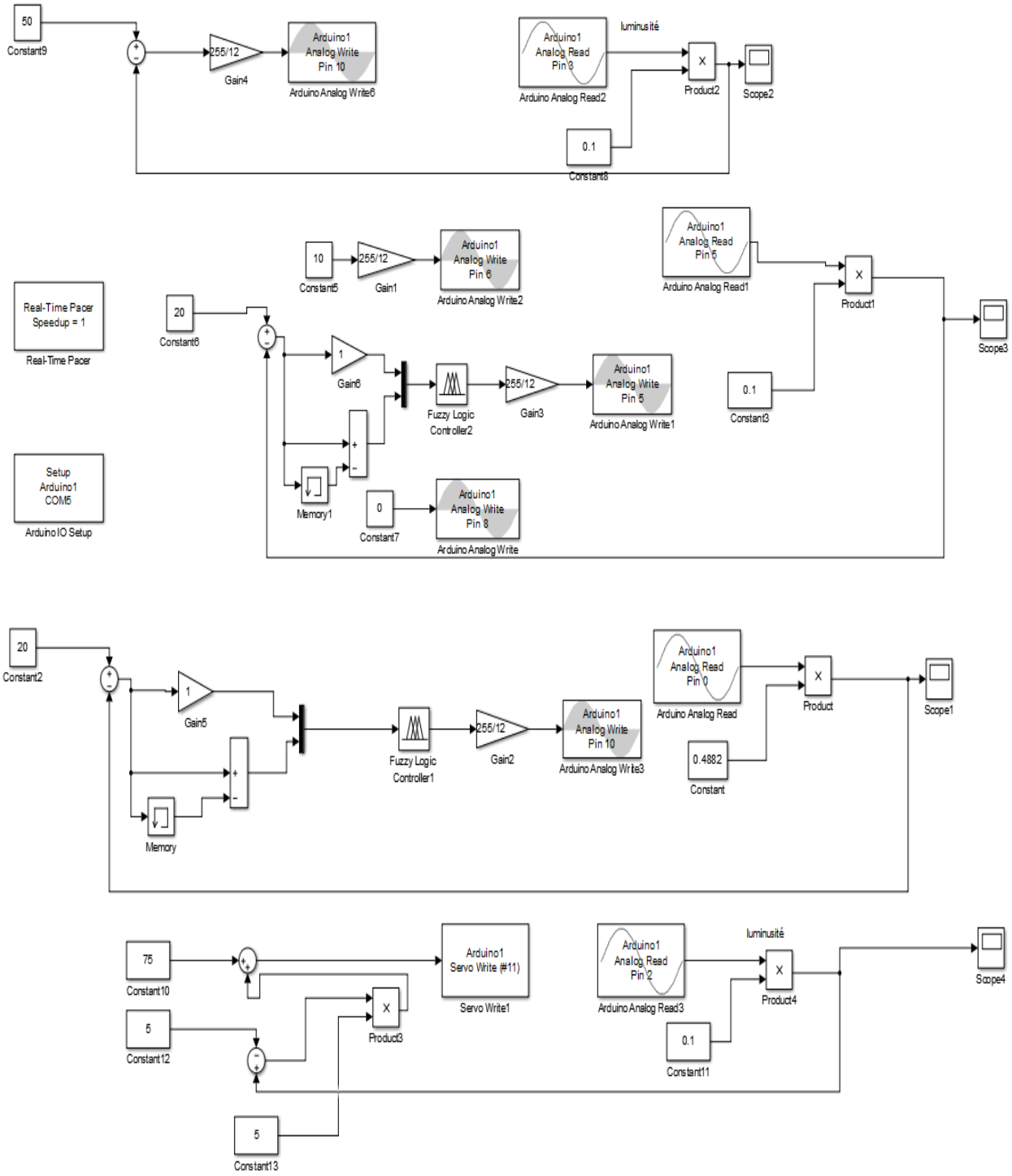


Figure IV.5 : Schéma bloc du système avec le contrôleur flou.

Note :

Pour la configuration des paramètres des correcteurs (PID et flou) voir l'annexe. Pour le contrôle de la luminosité et la qualité de l'air (co2) nous avons opté pour une commande

(TOR) tout ou rien vu que ces deux derniers sont des grandeurs difficile a identifié et a équilibré comme ils ne nécessitent pas une grande sensibilité de contrôle.

La figure IV .6 donne la réponse du système non corrigé à des échelons de consigne désirés.

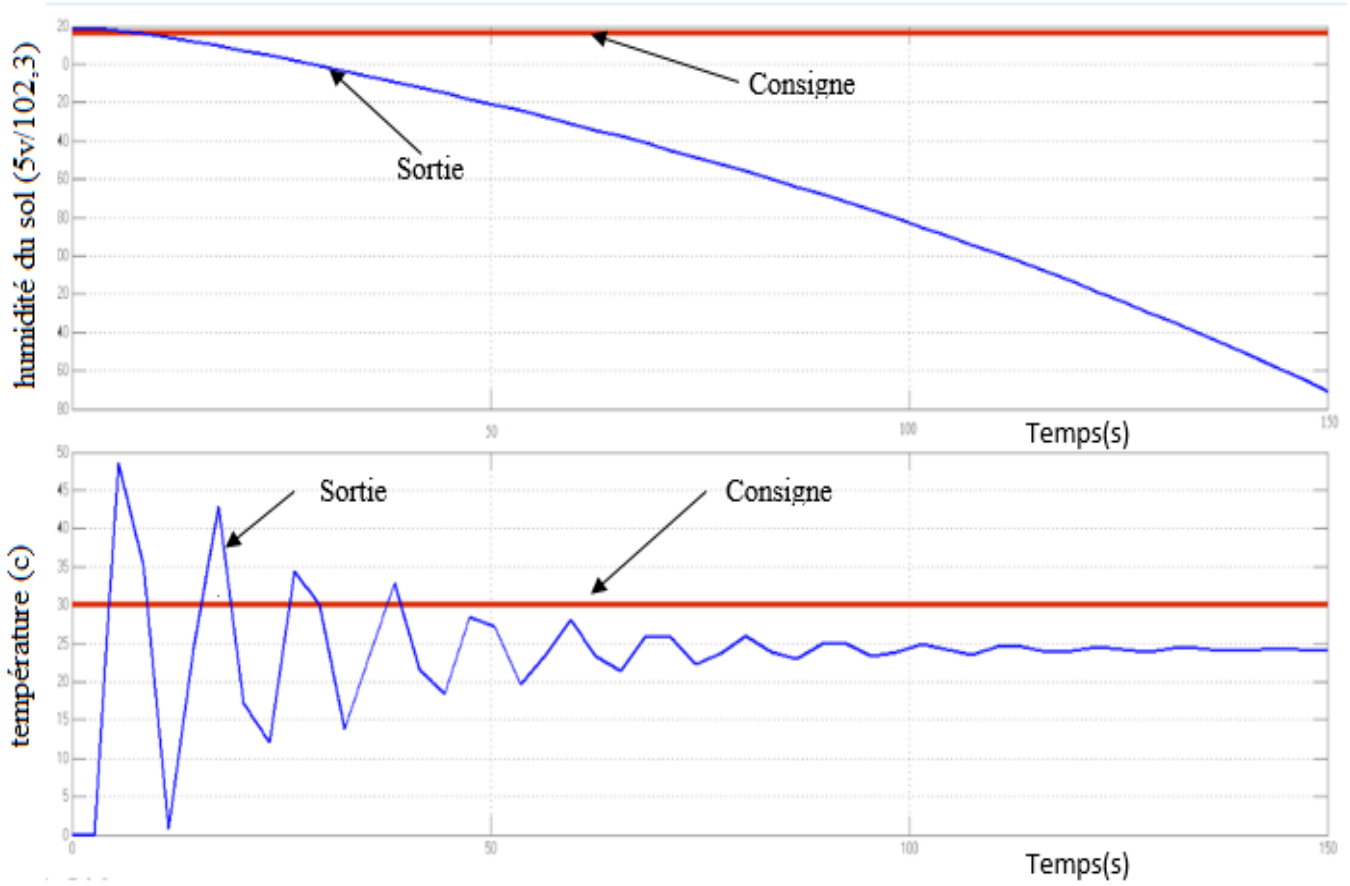


Figure IV.6 : réponse du système a un échelon non corrigé.

La figure IV.7 donne la réponse du système corrigé avec les deux contrôleurs (PID et Flou) à des échelons de consigne désirés.

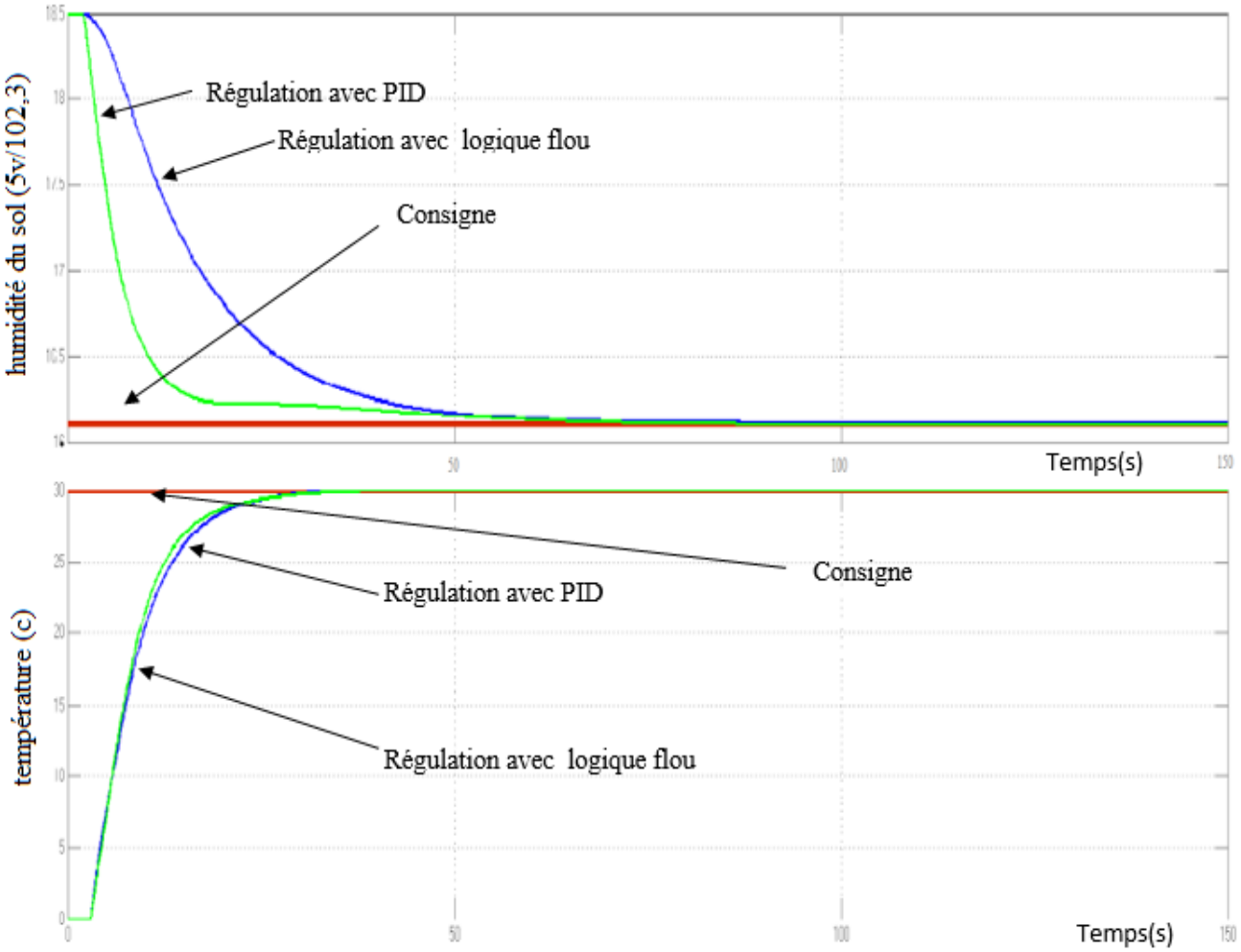


Figure IV.7 : Réponse du système a un échelon corrigé.

La figureIV.8 montre la variation de l'erreur ainsi que l'erreur après l'emploi des deux correcteurs afin de régler la température et l'humidité.

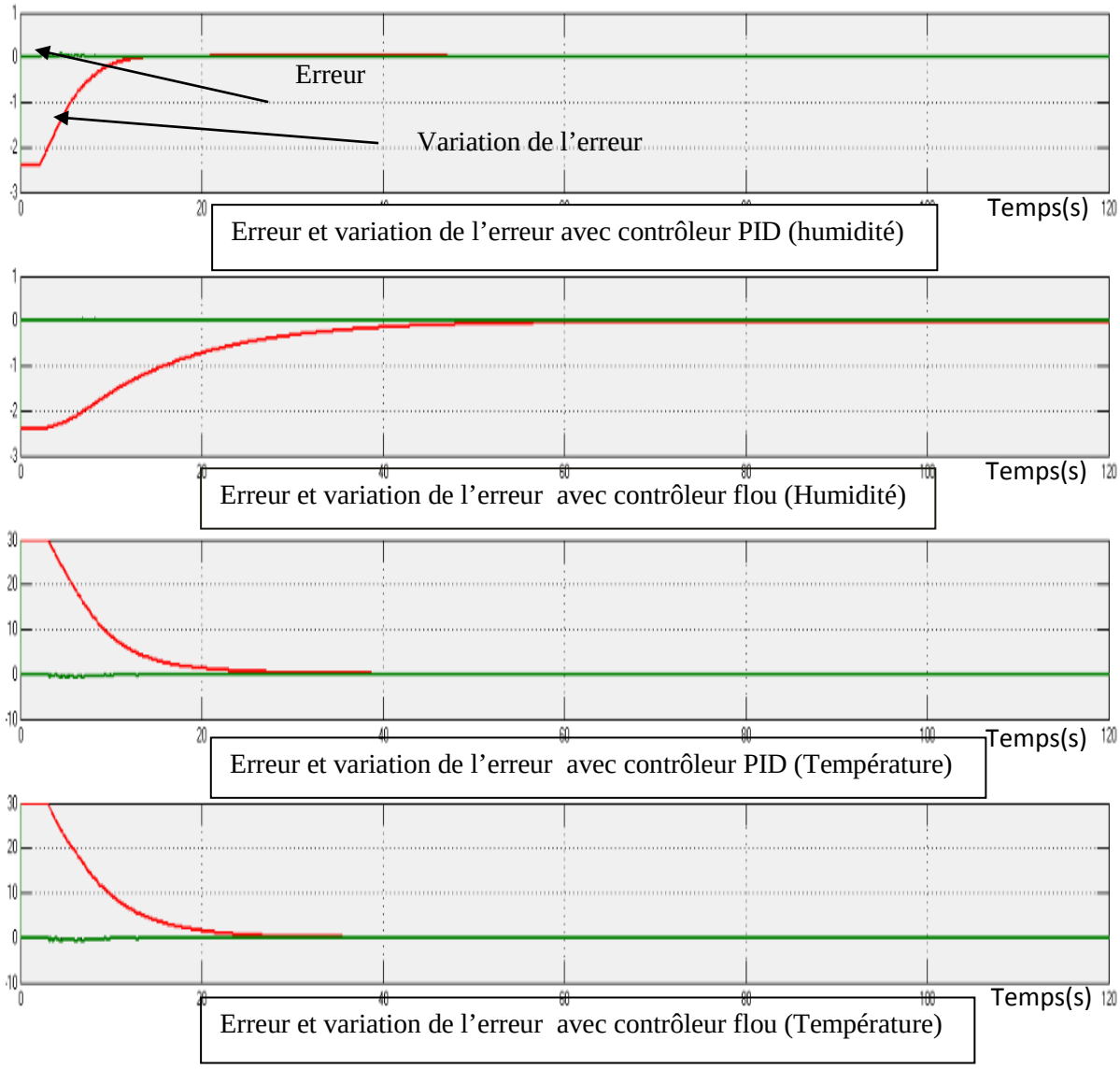


Figure IV.8 : Comparaison des résultats de la régulation.

a)-Interprétation des résultats théoriques:

Nous remarquons que les deux sorties convergent après un certain temps vers la consigne, ce qui veut dire avec les deux méthodes de régulation nous obtenons des résultats satisfaisants.

Nous avons obtenu un système corrigé en boucle fermée sans dépassement avec un temps de montée rapide et sans erreur en régime permanent (erreur statique).

Le temps de réponse est plus long avec le correcteur flou par rapport au régulateur PID. Ce dernier est un paramètre qui peut s'amélioré en considérants d'autre entrées comme :(l'intégrale de l'erreur, la dérivé de l'erreur ...etc.) Où d'ajouter plusieurs allures qui

définissent l'appartenance des variables pour le correcteur flou mais cela va augmenter la complexité du régulateur ainsi son temps de calcul.

b)-Comparaison des résultats théoriques :

Chacune de ces deux méthodes possèdent leurs avantages et inconvénients. La logique floue présente l'avantage de réguler un processus sans avoir un modèle mathématique précis du système ce qui est un avantage majeur pour notre procédé. Mais son inconvénient réside dans la complexité de sa conception qui n'est pas toujours facile à implémenter. Quant à la régulation PID, elle nécessite un modèle mathématique du système bien précis .Par contre cette méthode offre une très bonne régulation par rapport à la première. Donc le choix à faire dépend de l'application choisie et les connaissances du système.

IV)-Réalisation de procédé :

IV.1)- Dispositif expérimental :

La figureIV.9 représente le schéma synoptique du prototype réalisé.

Il s'agit d'un système d'acquisition et de contrôle, composé d'une carte d'acquisition à microcontrôleur Arduino pour le traitement et la commande servant d'interfaces entre le PC et les différents actionneurs et capteurs.

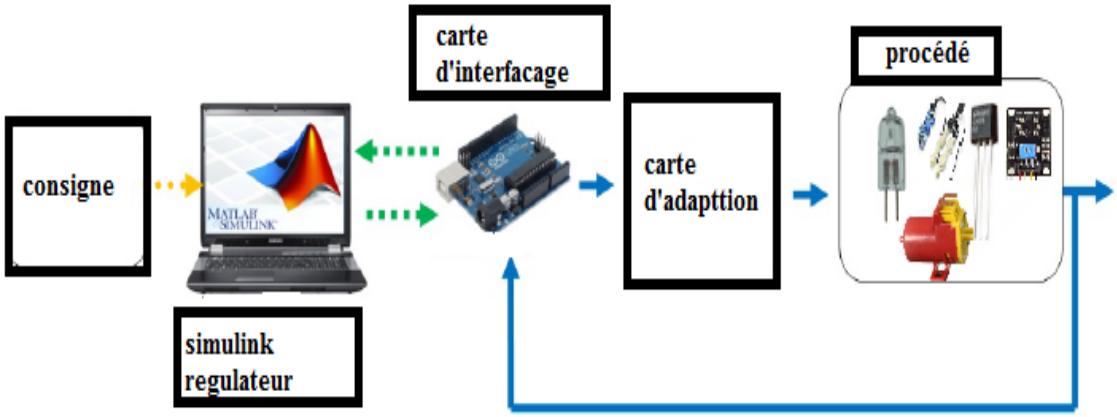


Figure IV.9 : Schéma synoptique du système.

Dans les paragraphes qui vont suivre nous donnerons une description des outils utilisés.

IV.2)- Prototype :

Le prototype dans lequel toutes les données expérimentales ont été recueillies est réalisé au niveau du laboratoire de projet d'électronique.

C'est un prototype dont le support est en bois et de surface est de 1820cm^2 et un volume de 30000cm ^{3}comme montré sur la figure IV.12.

Il est constitué d'un ensemble de matériels et programmes précités dans les chapitres précédents permettant l'acquisition et le stockage numérique des données, réalisant la commande des actionneurs.

La figure IV.10 visualise le schéma de simulation sous proteus.

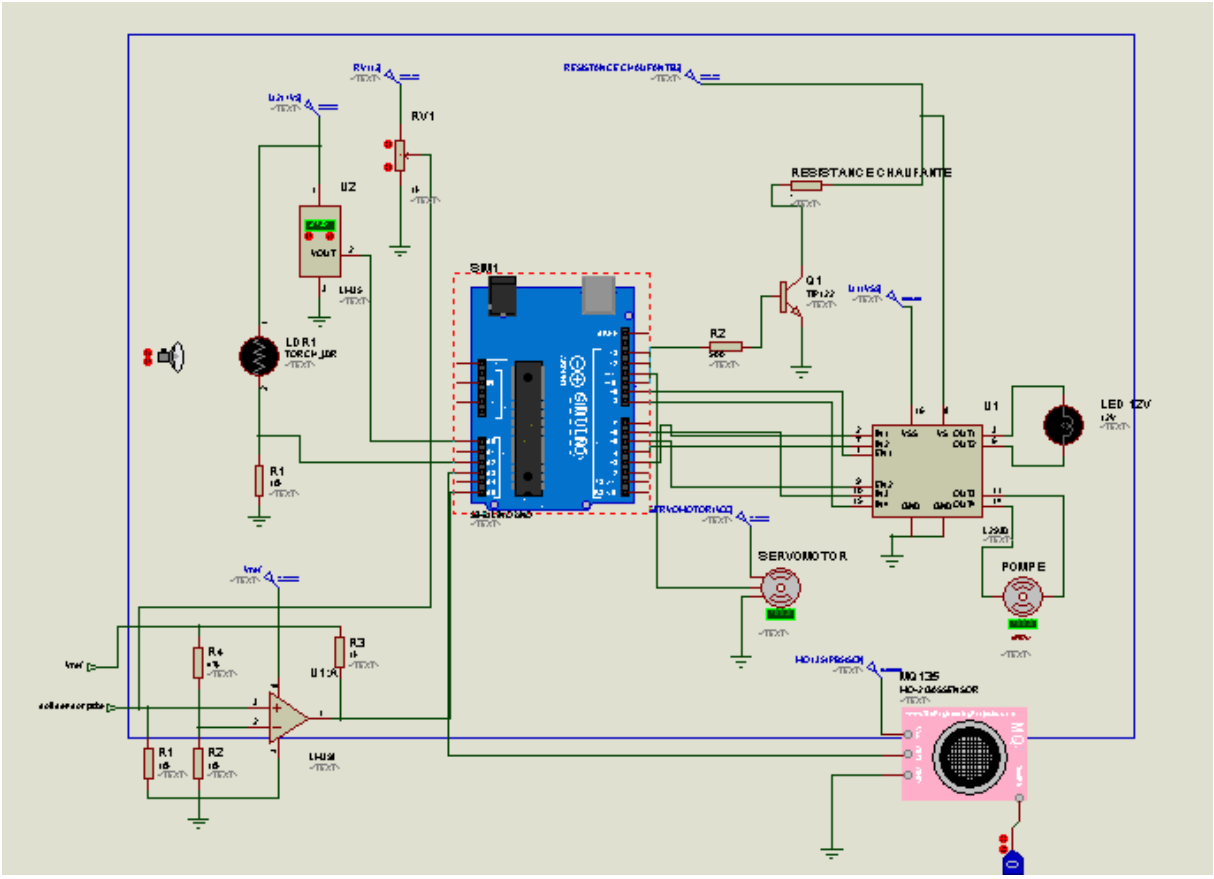


Figure IV.10 : Schéma de simulation sur Proteus.

Notons que la bibliothèque du capteur d'humidité YL-69 n'est pas disponible sur Proteus, pour contourner cette contrainte nous avons utilisé un potentiomètre qui délivre une tension qui s'étend sur la gamme de celle issue de ce capteur

La figure IV.11illustre le circuit électronique réalisé avec l'intermédiaire d'une carte Arduino UNO.

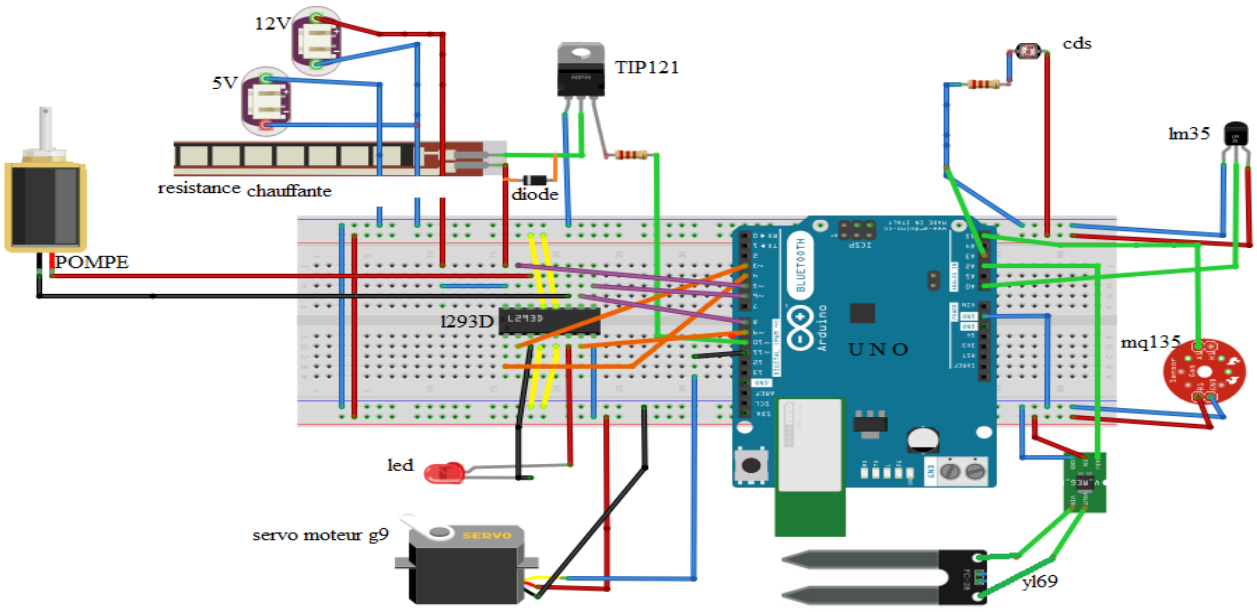


Figure IV.11 : Branchement du procédé avec la carte UNO.

La figure IV.12 nous montre le prototype réalisé.

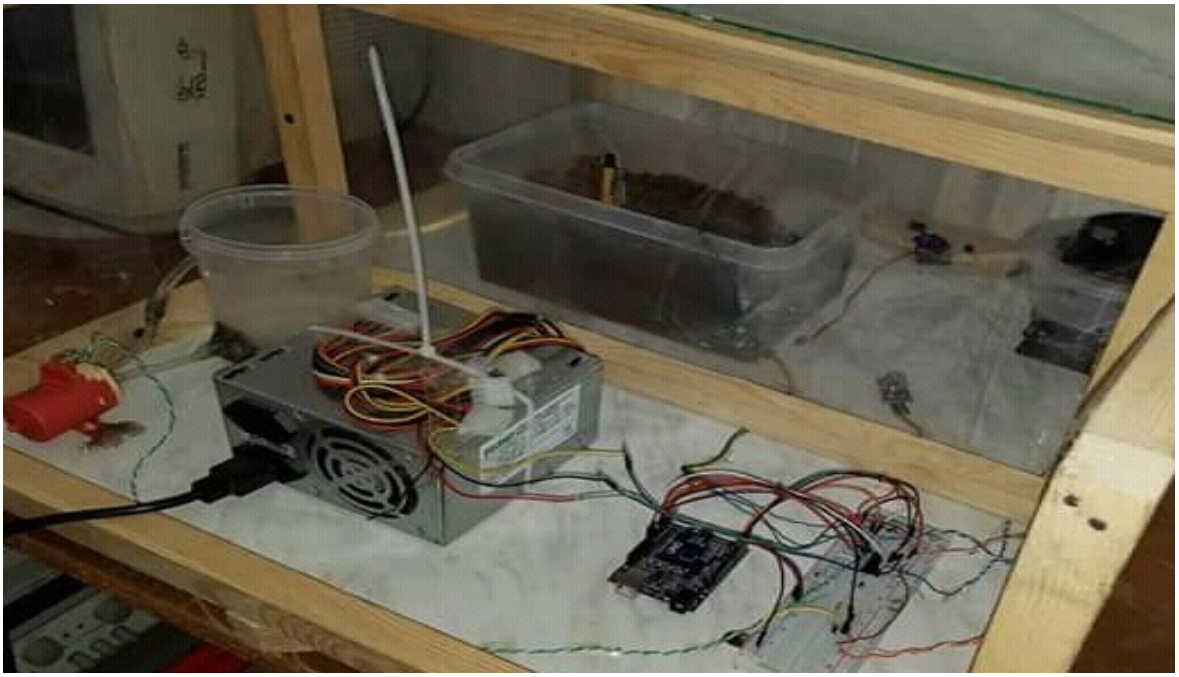


Figure IV.12 : Photo du Prototype réalisé.

L'organigramme de la figure IV.13 donne un aperçu sur le fonctionnement du procédé en boucle fermée.

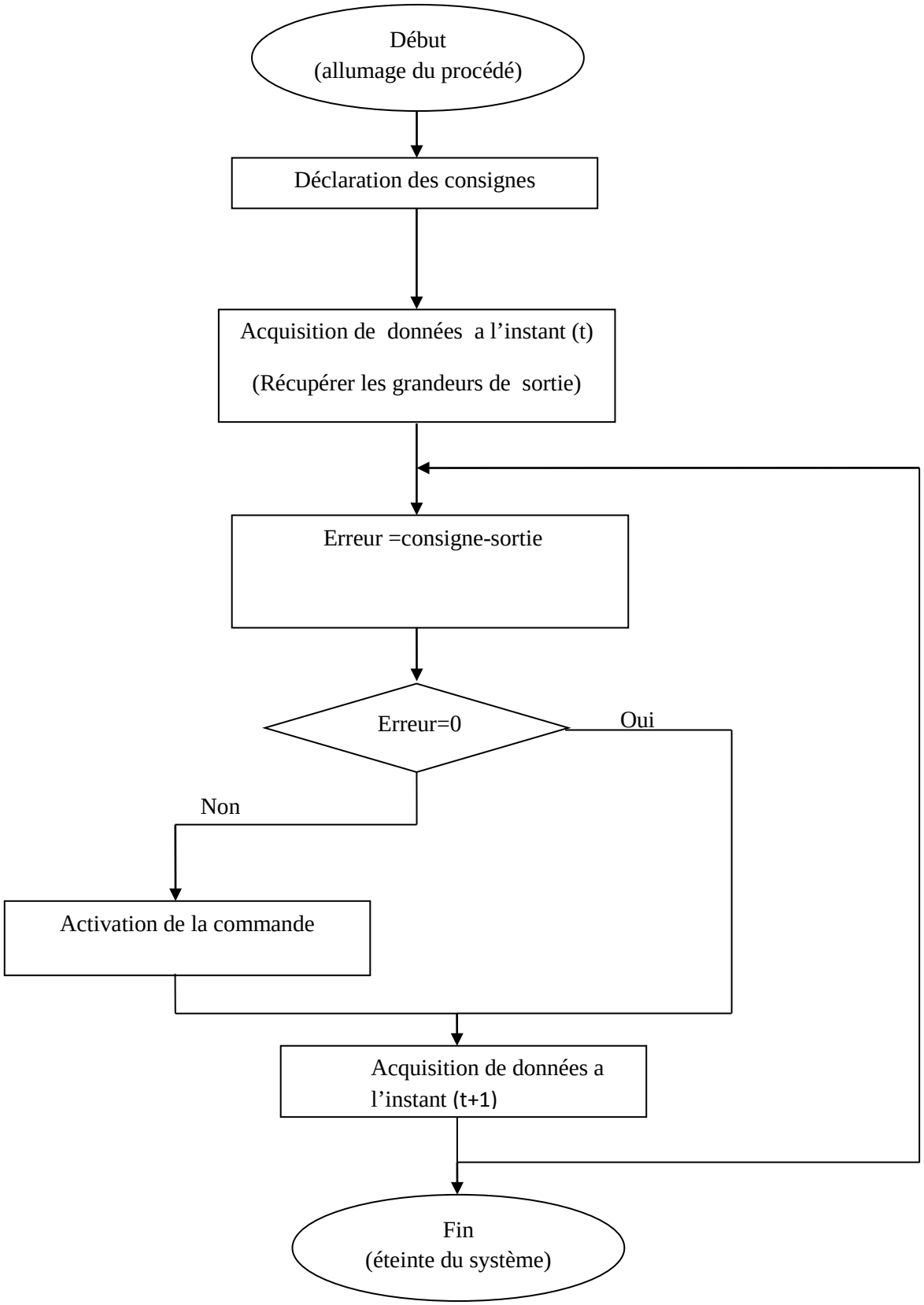


Figure IV.13 : Organigramme de la régulation.

V)-Evolution du système réel :

Afin d'assurer l'efficacité de la commande employé nous avons posté pour des tests réelles afin de visualiser l'évolution des paramètres pondant 72h avec et sans contrôle.

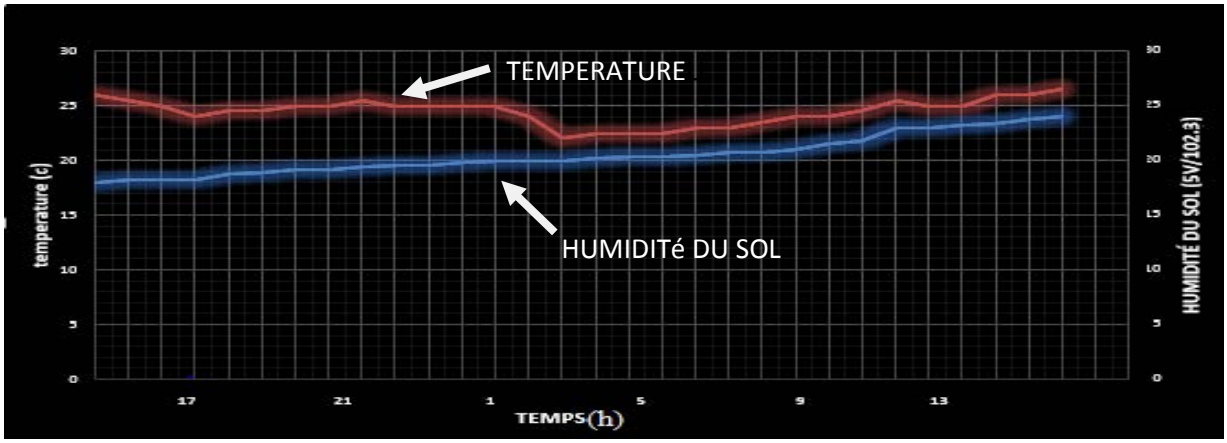


Figure IV.14 : Enregistrement de l'évolution des paramètres sans contrôle.

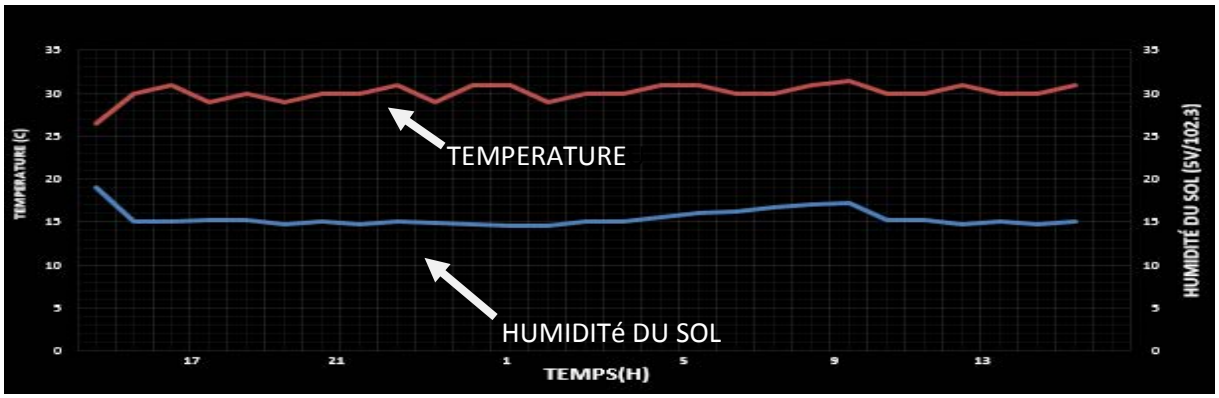


Figure IV.15 : Enregistrement de l'évolution des paramètres avec un contrôleur PID.

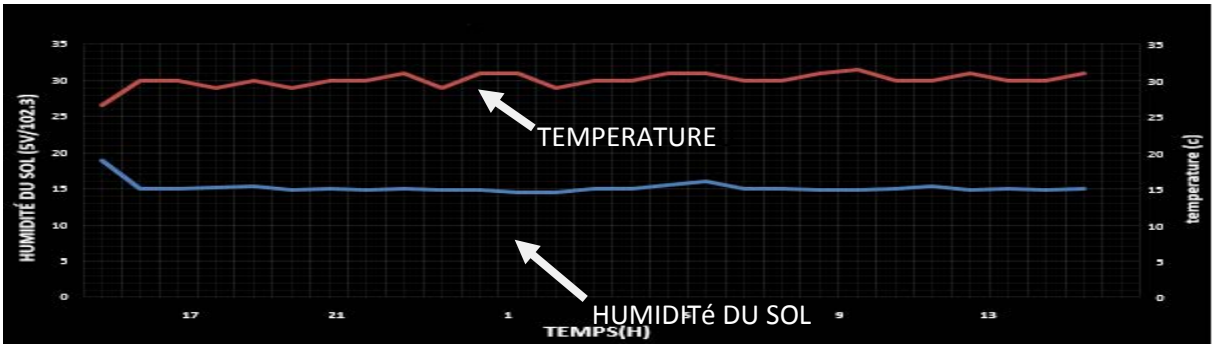


Figure IV.16 : Enregistrement de l'évolution des paramètres avec contrôleur flou

a)-Interprétation des résultats expérimentaux:

D'après ces enregistrements nous arrivons à des résultats satisfaisants très proches de l'étude théorique avec des légères erreurs (statique ou dynamique) imprévisibles peut être due au bruit de mesures ainsi que a la négligence des effets de couplages entre les deux grandeurs à réguler. Comme il est difficile de distinguer la différence entre l'emploi du correcteur PID ou flou. Nous pouvons supposer que le climat à l'intérieur de la chambre est fortement influencé par les conditions externes (température, humidité, rayonnement solaire…) nous concluions que les différentes commandes à employer dans une serre ont pour objectif d'amélioration du conditionnement interne de cette dernière.

Note :

La partie identification a été faite du 23 mai au 3 juin 2018.

Les tests expérimentaux ont été enregistrés le 14, 15, 16 juin 2018.

VI)- Conclusion :

Dans ce chapitre, nous avons présenté le système réalisé qui permet de fournir un environnement artificiel favorable à la survie des plantes. Comme nous avons procédé à la conception et l'implémentation des correcteurs flous et PID dans notre système à l'aide de la Toolbox fuzzy logic et PID tunning de Matlab. Pour conclure nous avons simulé sous Matlab Simulink l'évolution de la température et l'humidité du sol (théorique et réel) avec et sans contrôleurs et interprété les graphes résultants. L'implémentation des régulateurs à base de la logique floue ou PID offrent l'avantage d'avoir des réponses précises du système sur les grandeurs de commandes.

.

Conclusion perspectives

Tout au long de ce travail, nous avons cherché à concevoir un système de régulation automatique pour la climatisation de l'environnement dans une chambre. Ce système est composé d'un ensemble de capteurs ayant pour rôle de l'acquisition des données qui sont traitées par une carte à microcontrôleur Arduino, d'actionneurs servant à réguler les grandeurs choisis.

Pour l'élaboration de la loi de commande en premier lieu, nous avons opté pour les modes de dépouillement graphique de STREJC et BROIDA afin d'identifier les fonctions de transferts de la température et de l'humidité du sol qui sont de premier et de deuxième ordre respectivement, TOR pour la luminosité et qualité de l'air. Ensuite nous avons implémenté sur le logiciel Matlab dans le module simulink les lois de commandes à base des régulteurs PID et logique floue pour la temperature ambiante et humidité du sol dont les résulats obtenus sont largement satisfaisants .

Comme perspective de ce travail, des extensions du système peuvent être considérées tenant comptes d'autres grandeurs physiques comme la température du sol, le PH et la conductivité électrique afin d'enrichir la loi de commande. Le champ d'application de cette technique peut être exploité pour développer une serre à plus grande échelle. De plus, l'utilisation d'autres types de commandes avancées tel que les réseaux de neurones artificiels donnera un système plus fiable et plus précis. En outre les méthodes et les techniques développées au cours de ce travail peuvent être étendues à d'autres types de systèmes afin d'améliorer leurs performances et d'obtenir des réponses précises sur les grandeurs physiques à réguler.

A.1)-Méthodes d'identification utilisées :

A.1.1)-Broïda :

Broïda assimile la réponse indicielle à celle d'un système du premier ordre affecté d'un retard pur fictif, nous envoyons au système un signal d'entrée X(t) de type échelon et puis nous enregistrons le signal de sortie Y(t).

On conclut une fonction de transfert de la forme :

$$G(p)= \frac{ke^{-\theta p}}{1+Tp} \qquad (A.1)$$

-K : gain statique

-T : constante de temps

-θ: retard

Pour déterminer les paramètres de ce modèle, nous suivons les étapes suivantes :

-Déterminer graphiquement l'instant t1 correspondant à 28% de la valeur finale

-Déterminer graphiquement l'instant t2 correspondant à 40% de la valeur finale

-Calculer θ et T par les expressions suivantes :

$\theta = 2.8\ t1 - 1.8t2$

$T= 5.5\ (t2 - t1)$

-Le gain statique se calcule en régime statique.

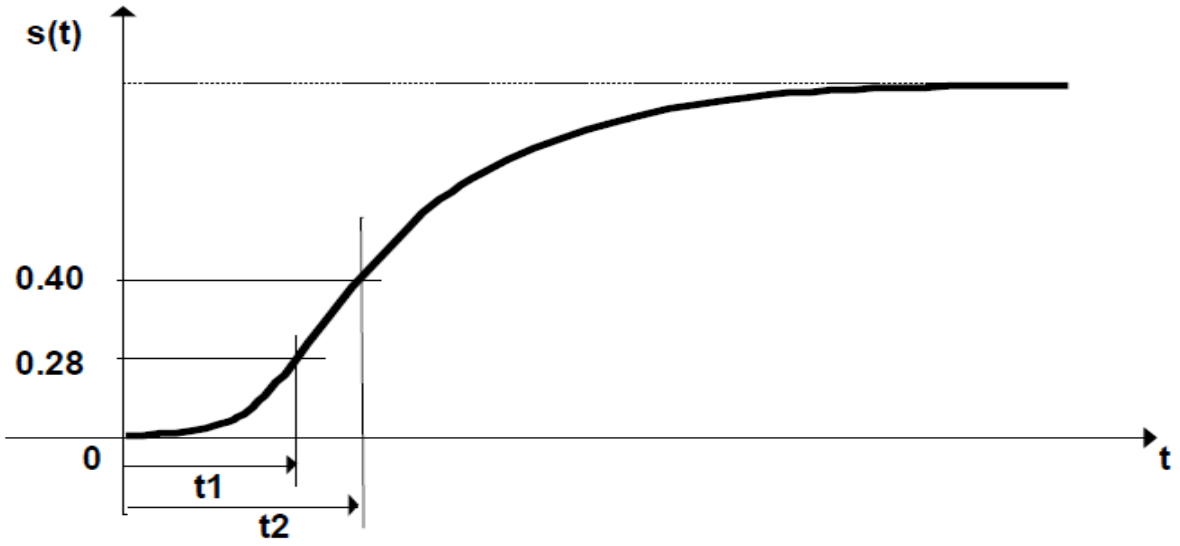


Figure A1 : Model graphique avec la méthode de Broïda.

A.1.2)-Strejc :

Cette méthode peut s'appliquer aux systèmes dont la réponse indicielle ne présente pas de dépassement. On conclut une fonction de transfert de la forme :

$$G(p) = K.\frac{e^{-\tau.p}}{(1+T.p)^n} \qquad (A.2)$$

Ou Les paramètres à identifier sont :

-le gain statique K

-le retard τ

-la constante de temps T

-et l'ordre n

Pour identifier le système, la méthode peut se décomposer en :

-calcule du gain statique qui est la valeur finale de la sortie par l'entré.

-On trace la tangente au point d'inflexion I pour déterminer les deux valeurs: T1 et T2.

-Relever T1 et T2, et puis déduire l'ordre n en utilisant le tableau suivant.

Entre deux lignes du tableau, on choisit la valeur de n la plus petite.

-Déterminer la constante de temps à partir de $\frac{T2}{T}$ du tableau.

- Déterminer le retard τ quand il existe à partir de la différence entre la valeur de T1 mesurée et celle donnée par la colonne $\frac{T1}{T2}$ du tableau.

n	$\frac{T1}{T}$	$\frac{T2}{T}$	$\frac{T1}{T2}$
1	0	1	0
2	0.28	2.72	0.1
3	0.8	3.7	0.22
4	1.42	4.46	0.32
5	2.10	5.12	0.41
6	2.81	5.70	0.49

Tableau A1 : Table de mesure.

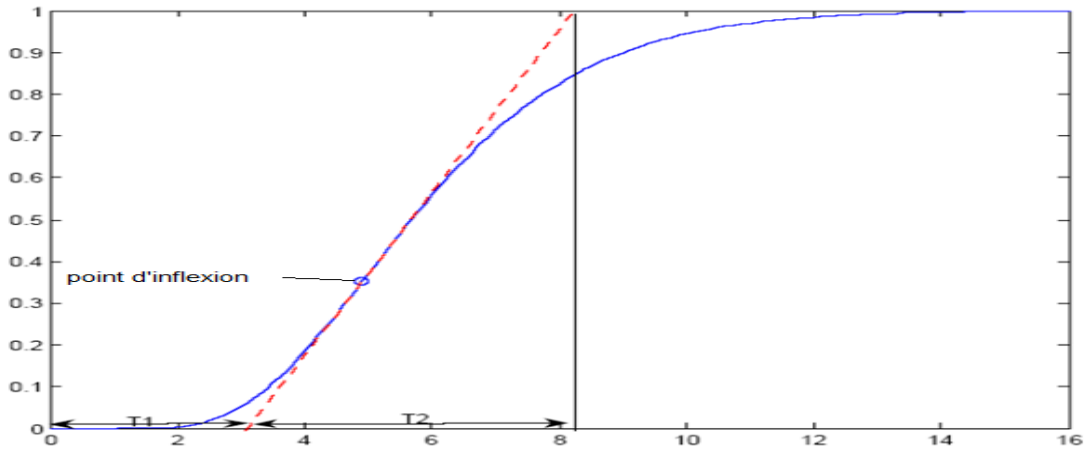


Figure A2 : Model graphique avec la méthode de Strejc.

A.1.3)-Données pour l'identification du système

(Température) **:**

Temps [s]	**T air [c°]**
0	0

1	0
2	0
3	15
4	22
5	33
10	39
15	41 ,5
20	43
30	47
35	48
40	50
50	50
60	50

Tableau A2 : Mesure de la température de l'air.

(Humidité) :

Temps [s]	**H_{sol} [SI]**
0	18.5
10	18.5
15	18.5
20	18.5
25	18.5
27	18.2
28	18.1
29	17.7
30	17.3
33	17
35	16.7
37	15.7
40	15
43	14
45	13
49	10
55	9
60	8
65	8
70	8
75	8
80	8

Tableau A3 : Mesure de l'humidité de sol.

A.2)-Tests effectués :

A.2.1)-Mono variable -multi variable :

Nous avons une seule entrée pour tous les sous-systèmes sur laquelle on agit (puissance) et une seule sortie (température ou humidité…etc.) et comme nous n'avons pas remarquer d'interaction sur les deux grandeurs à réguler c'est ce qui nous mène à supposer que notre système ne présente pas un couplage. Donc Il s'agit d'un ensemble de systèmes mono variables.

A.2.2)-Systèmes à paramètres localisés ou à paramètres repartis :

Pour notre processus, les mesures recueilles dépendent légèrement de l'emplacement des capteurs et des actionneurs, et pour cela on peut supposer que le système est à paramètres localisés.

A.2.3)-Linéarité (linéaire / non linéaire) :

Les allures de la sortie après l'injection de différents échelons de tension sont identique avec les courbes identifiés juste avec des changements dans les paramètres (temps de monté, erreur…etc.) Donc on constate que les sorties (températures et humidité du sol en fonction du temps) ne sont pas proportionnelles. Il s'agit donc des systèmes non linéaires.

A.2.4)- Déterministe/Stochastique :

Pour notre processus, nous appliquons un échelon de tension et on répète l'expérience plusieurs fois tout en maintenant le même échelon et on enregistre l'évolution de la sortie en fonction du temps pour chaque essai et pour les deux systèmes.

Nous avons constaté que pour les différents essais, les valeurs de la sortie sont très proches. Donc il s'agit des systèmes déterministes.

A.2.5)-Stationnarité :

Notre système aie des grandeurs qui varient en fonction de temps et ne dépendent pas de l'origine donc nous pouvons le décrire par un modèle mathématique

invariant dans le temps et c'est ce qui nous mène à constater que c'est un modèle stationnaire.

A.3)-Implémentation des correcteurs PID :

A.3.1)-Méthodes de réglage utilisées :

Il existe plusieurs méthodes pour régler une boucle de régulation dont les plus courantes sont :

a)-Méthode d'approximations successives :

Le système est mis en boucle fermé. Puis on règle successivement l'action proportionnel, l'action intégrale et enfin l'action dérivée.

1. On annule l'action intégrale et dérivée en mettant les valeurs de I et D à zéro.
2. On augmente Kr jusqu'à avoir la réponse la plus rapide avec un amortissement maximum et un écart minimum.
3. On diminue I pour annuler l'erreur statique et avoir la réponse la plus rapide avec un amortissement maximum et un écart minimum.
4. On augmente D jusqu'à ce que la boucle soit suffisamment rapide pour atteindre rapidement sa consigne.

Paramètres	Temps de montée	Dépassement	Temps de réglage	Erreur
P	augmente	augmente	Changement faible	diminue
I	diminue	augmente	augmente	éliminé
D	Changement faible	diminue	diminue	Changement faible

Tableau A4 : Effet du réglage des paramètres.

b)-La méthode de Ziegler et Nichols :

Pour obtenir les paramètres du régulateur PID, il suffit d'enregistrer la réponse indicielle du processus seul (c'est-à-dire sans le régulateur), puis de tracer la tangente au point d'inflexion de la courbe. Nous mesurons ensuite sa pente (p), le retard apparent (L) correspondant au point d'intersection de la tangente avec l'abscisse et le gain (K0)= y (∞)/E. (voir la figure A3.). On peut alors calculer les coefficients du régulateur choisi à l'aide du tableau 1.

Généralement, les gains Kp proposés par Ziegler-Nichols sont trop élevés et conduisent à un dépassement supérieur à 20%.

Type	K_p	T_i	T_d
P	1/ (pLK0)		
PI	0.9/ (pLK0)	3L	
PID	1.2/ (pLK0)	2L	0.5L

Tableau. A4 : Paramètres PID obtenus à partir de la réponse indicielle illustré par la figure A3.

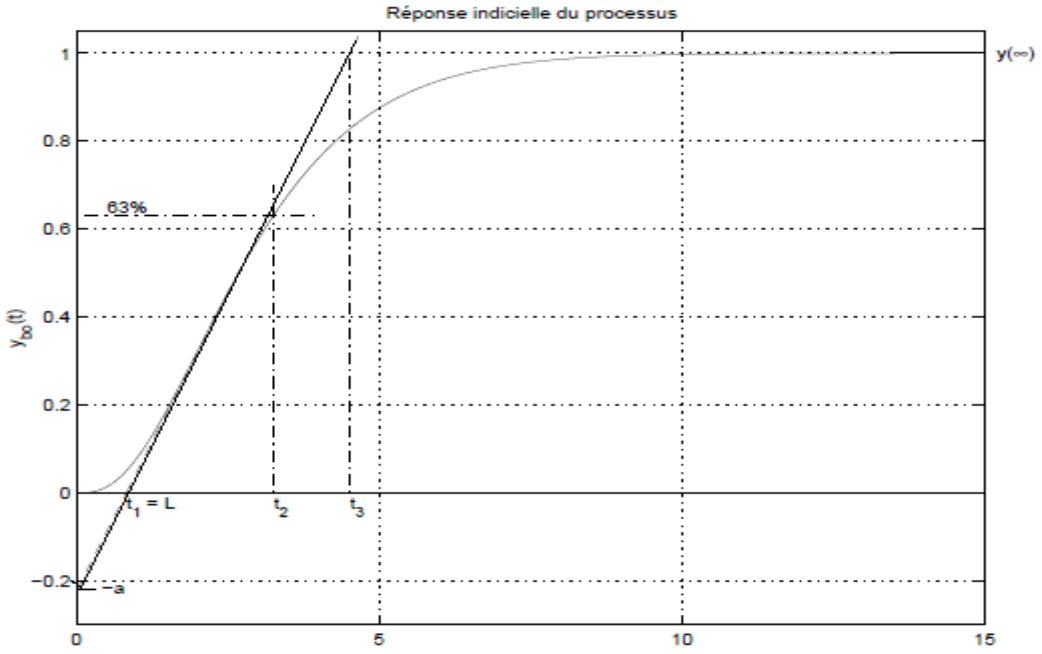


Figure.A3 : Réponse indicielle d'un processus.

Pour la régulation PID nous avons utilisé l'application (PID tunning) offerte par Matlab simulink qu'est fait référence à la méthode d'approximations successives.

L'application (PID tunning) est obtenue en frappant (tune) dans la fenêtre du bloc PID.

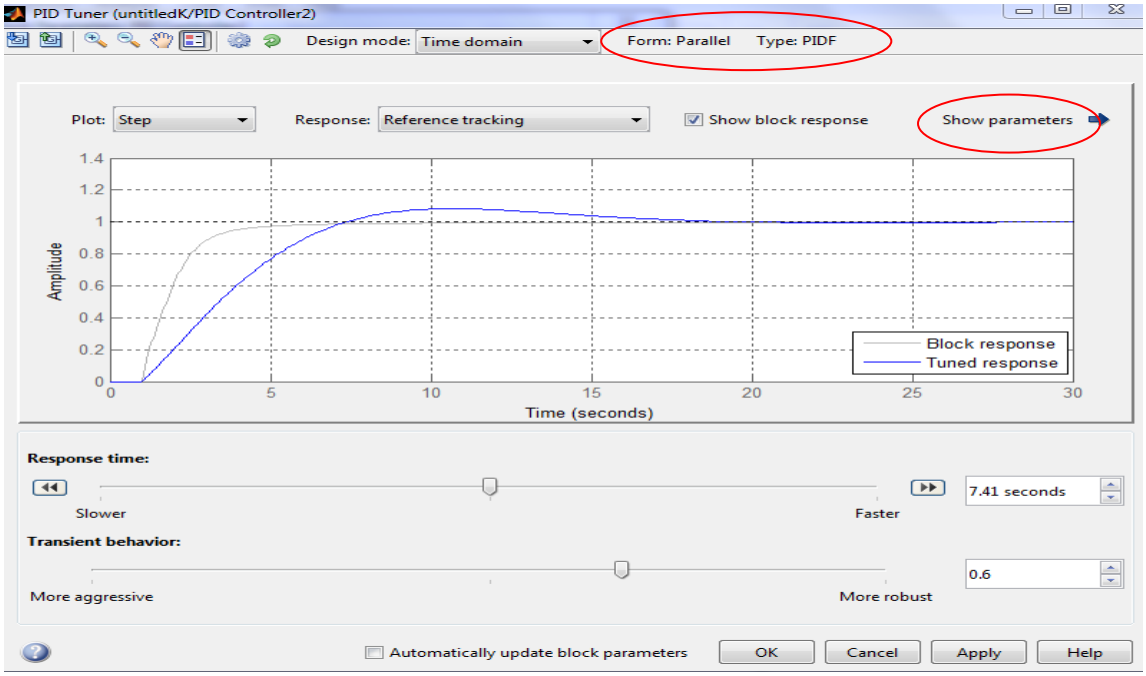


Figure A4 : Interface de l'application PID tunning.

Pour la visualisation des paramètres du correcteur PID ou le choix du correcteur PID il suffit d'exploiter les deux zones encerclé en rouge dans la figure A4 précédente.

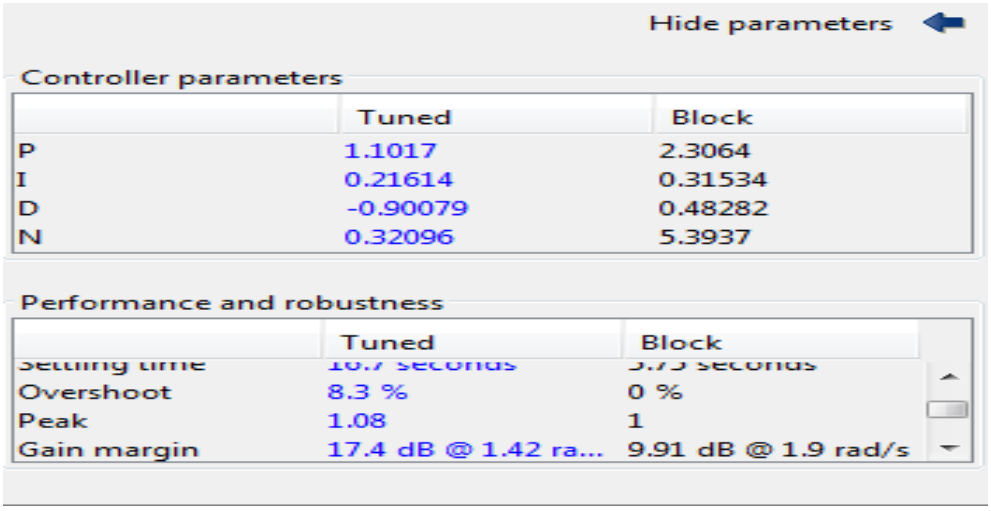


Figure A5 : Visualisation des paramètres du PID.

Les paramètres choisit pour les deux correcteurs sont motionné dans le tableau A5 :

	température	Humidité du sol
P	0.161	-2.983
I	0.0274	-0.127
D	0	-17.459

Tableau A5 : Paramètres des correcteurs.

A.4)-Implémentation des correcteurs flous :

Nous utilisons la boite à outils (fuzzy logic toolbox) de Matlab pour la conception du régulateur floue.

Cette boite à outils permet de définir les variables linguistiques ainsi que les fonctions d'appartenance associées aux variables de commande du système.

L'interface (Fis Editor) est obtenue en frappant la commande (fuzzy) dans la fenêtre de commande de MATLAB.

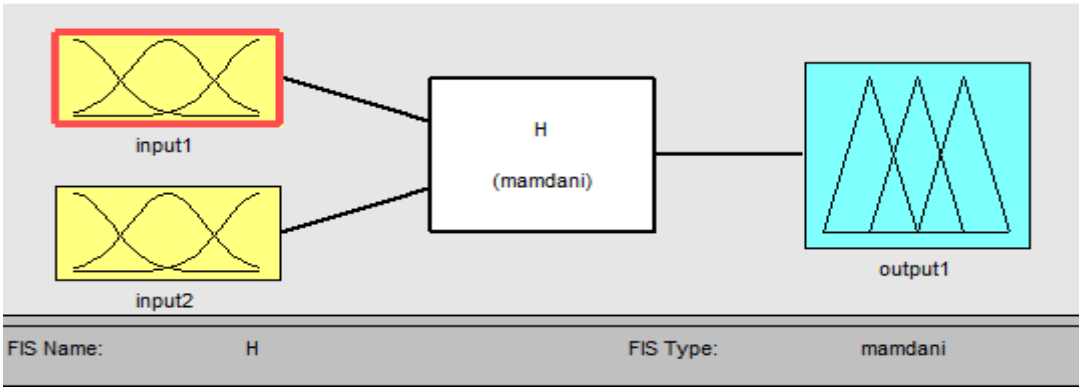


Figure A6 : Interface de l'outil fuzzy logic toolbox.

Dans le but de la réalisation du correcteur flou nous choisissons l'erreur et la variation de l'erreur comme entrés et la commande comme sortie des systèmes, comme d'autres paramètres peuvent être implémenté comme la dérivé de l'erreur ou l'intégrale de l'erreur … etc.

Le choix de ces trois paramètres peut s'expliquer mathématiquement comme suivant :

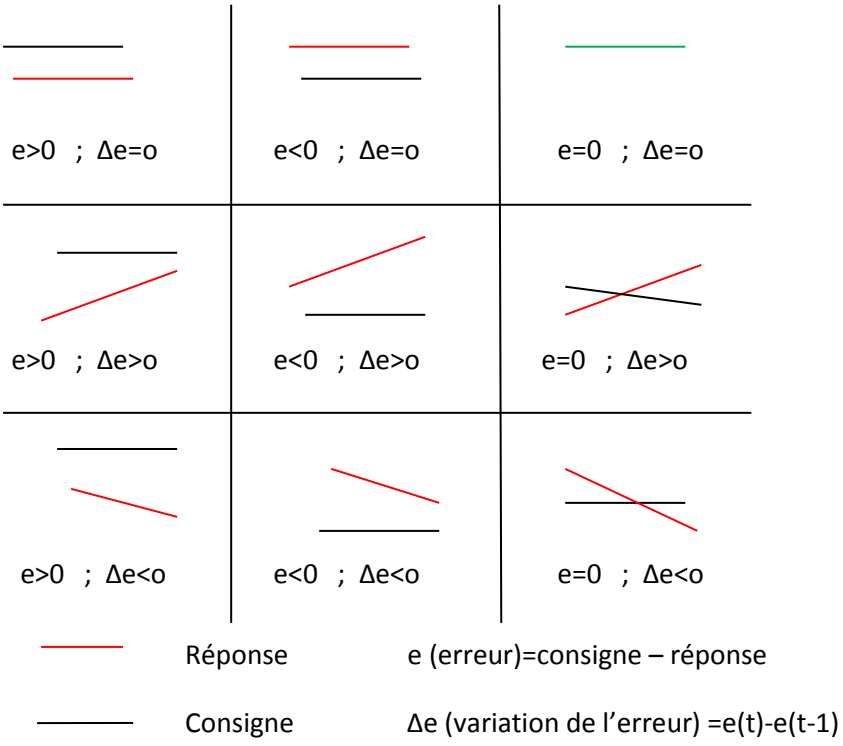


Figure A7 : Réponse du système aux paramètres choisis.

Nous pouvons définir les bases de règles utilisées par le système d'inférence avec l'interface (Rule Editor).comme suivant :

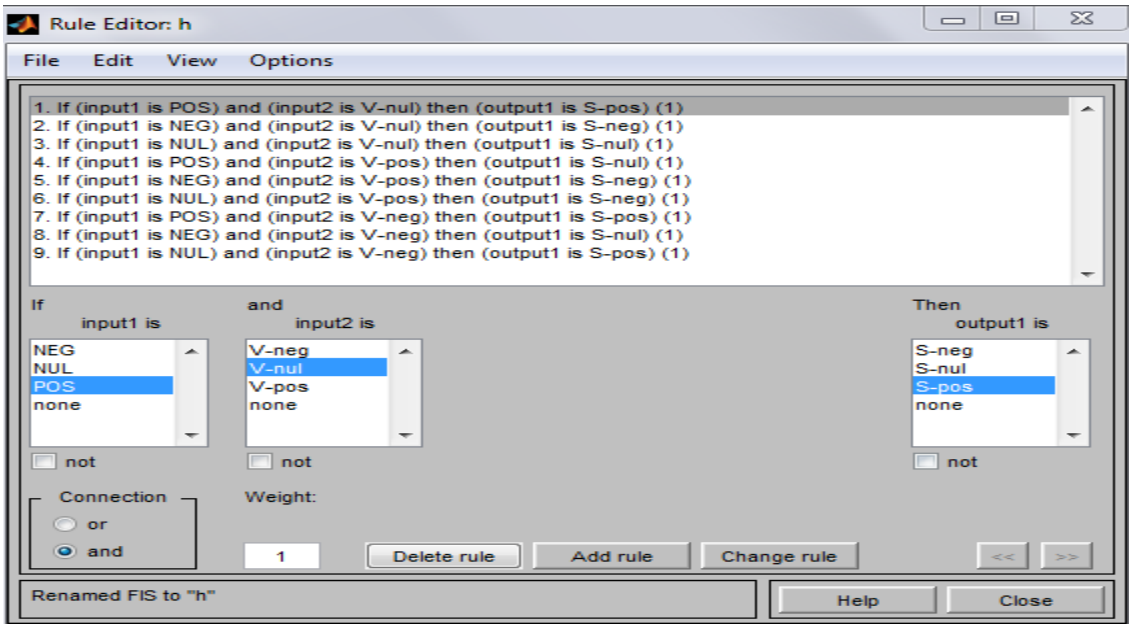


Figure A8 : Interface de définition des bases de règle.

Pour la température nous avons opté pour ces fonctions d'appartenances (labelles) suivantes :

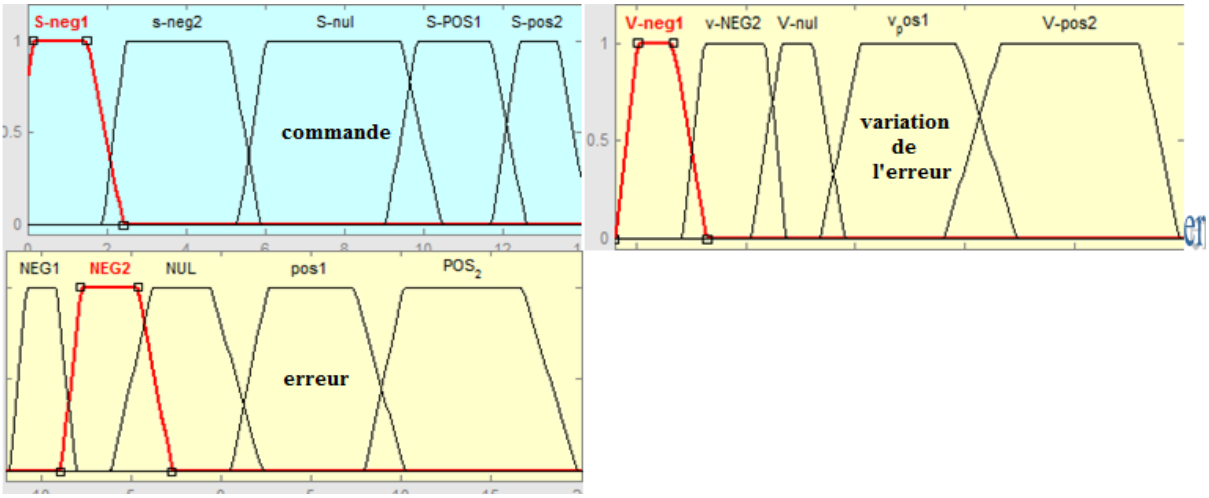


Figure A9 : Fenêtre d'édition des fonctions d'appartenance.

Pour l'humidité du sol nous avons opté pour ces fonctions d'appartenances (labelles) suivantes :

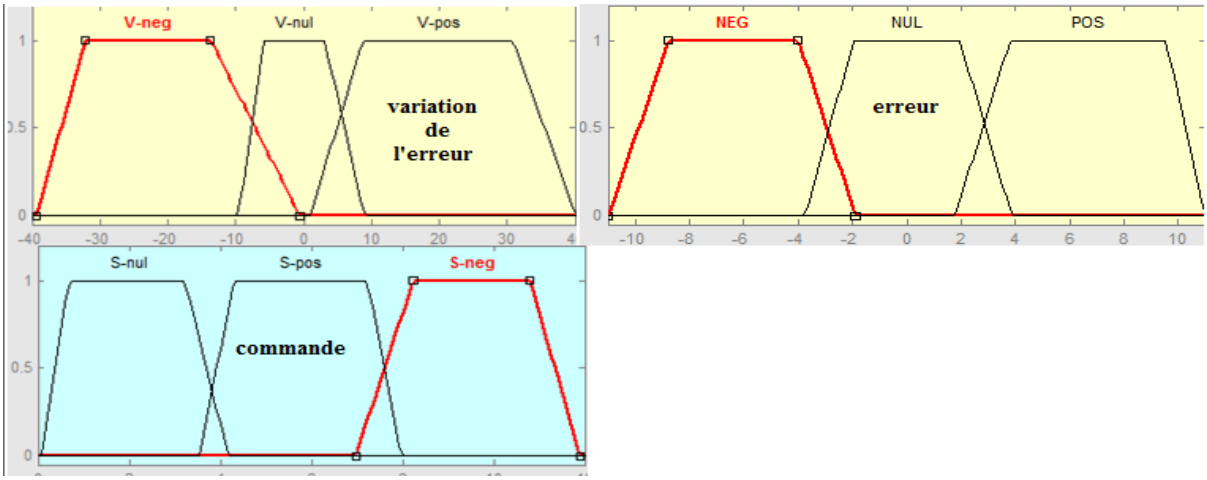


Figure A10 : Fenêtre d'édition des fonctions d'appartenance.

La figure A11 illustre le schéma block permettant d'effectuer l'étude théorique comparatif.

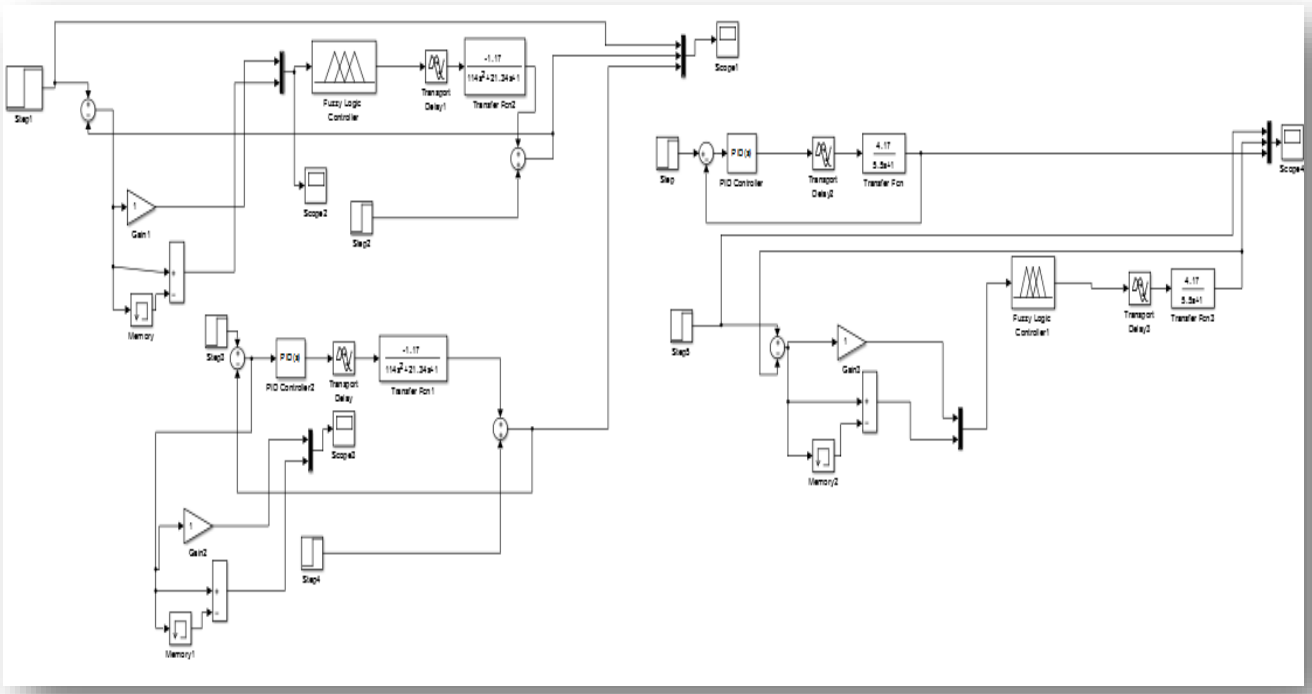


Figure A11 : Schéma block corrigé avec les deux méthodes de régulation.

Références Bibliographique

[1] Erik BARTMANN « Le grand livre d'Arduino», Eyrolles , Paris 2015.

[2] www.electroschematics.com.

[3] www.amazon.fr .

[4] Nabil-Rafik «Etude, conception et réalisation d'une plateforme pour l'automatisation et le contrôle à distance des serres agricoles», UMBB, Juin 2017 .

.

[5] http : //www.mathworks.com.

[6] http://playground.arduino.cc.

[7] JEAN- MARIE FLAUS « La régulation industrielle » HERMES science publications, Paris 2000.

[8] Bernadette Bouchon-Meunier «La logique floue et ses applications» Vuibert 1995.

[9] BOUCHON-MEUNIER Bernadette, MARSALA Christophe Logique floue, principes, aide à la décision 2003.

[10] ALINA BESANCON-VODA ET SYLVIANNE GENTIL « Régulateur PID analogique et numérique » Technique de l'ingénieur : R7416 (03/1999).

[11] Cours « Asservissement » M^{r}. DIAF, 2ème année, Automatique, UMMTO
Année universitaire 2015/2016.

[12] Cours « Modélisation » M^{elle} DJEGHALI, 2ème année, Automatique, UMMTO
Année universitaire 2015/2016.

[13] CHELLY NIZAR-CHRED AMINE « Formation Arduino Matlab/ Simulink commande d'un moteur a courant continue à l'aide de la carte Arduino.
UNO » HAMMAMET 2016.

[14] Y. El Afou, «Contribution au contrôle des paramètres climatiques sous serre». Thèse de Doctorat Université Lille 1, 2014.

Résumé

Ce travail consiste à la conception et la réalisation d'un système de régulation à base d'une carte à microcontrôleur (Arduino UNO).Le prototypage d'Arduino et Matlab Simulink est réalisé dans le but d'assurer un serial communication entre la carte à microcontrôleur (UNO) et le logiciel (Matlab).La régulation par logique floue, PID et TOR sont ici utilisées pour le réglage de la température ambiante, humidité du sol et luminosité ainsi la qualité de l'air dans notre système. Les étapes de conception de ces contrôleurs sont abordées et traitées.

La conception de ce système de régulation est passée par une identification suivie d'une simulation avec un contrôleur flou puis avec PID, enfin une comparaison des résultats obtenus est faite dans le but d'assurer une commande plus optimale pour notre système.

Mots clés :

Matlab, Arduino, capteur, actionneur, circuit électronique, logique flou, PID, régulation automatique, prototypage, microcontrôleur, identification, Broida, Strejc.