<u>République Algérienne Démocratique et Populaire</u>

<u>Ministère de l'Enseignement Supérieur et de la Recherche Scientifique</u>

<u>Université de Médéa</u>



Faculté des Sciences et de la Technologie

Département du Génie Electrique et Informatique

Mémoire de Fin d'Etudes pour l'Obtention du Diplôme de Master en
Génie Electrique

Option : **Automatique des système**

<u>Thème</u> :

Étude et réalisation de régulateur intelligent du niveau D'eau dans une enceinte métallique avec appoint

<u>Proposé par</u> : *Mr.Adnane.MESSAI*

<u>Etudié par</u> *:*

- M elle MAIZI Samiha
- M elle KHANAFIF Aicha

2012-2013

Dédicace 1

Je dédie ce modeste travail à mes très chers parents, et mon père avec toute ma reconnaissance. surtout l'étoile de ma vie celle que je vois jour et nuit, et je parle bien sur de ma mère symbole de patience de tendresse et d'amour celle qui a été toujours là pour moi, c'est grâce a celle que je suis ce que je suis.

Je le dédie également à mon chers frères MOUSSA, MAHFOUD, BOUFATEH, ISSLAM. mes sœurs TOURIA et FATMA, FATIHA ,et petite enfant MOHAMED ABD ALHAYE.

A tous mes amis et mes collègues qui étaient avec moi dés le début de mon parcours surtout .

A ma binôme SAMIHA MAIZI.

A mes amies Mariem.M, ma chère Razika , Hadjira Wahiba,mina, FATI,NADJIA, ,MADINA,AMEL ,CHERIFA,MASSOUDA , ASSMA ,HOURIA,khaoula,Ahlam,sara , Linda ,Souria , soumia.

A tous mes amis 2bobaker, hamza,abd akarim,Brahim

toute ma famille Cherifa,nassira ,Aicha ,Fatima….. ,Ramdan ,Nour din….ect .

A toute la promotion de 2eme année master2013 .

A tous les enseignants qui m'ont aidé de proche ou de loin pour être un jour un master en section d'option automatique.

A tous ceux que j'aime et qui m'aiment et tous ceux qui me connaissent.

Je dédie enfin ce travail à tous ceux qui ont partagé mes joies et peines.

Dédicace 2

Je dédie ce modeste travaille qui est le fruit de mes études : à mes très chers parents, et mon père avec toute ma reconnaissance. surtout l'étoile de ma vie celle que je vois jour et nuit, et je parle bien sur de *ma mère* symbole de patience de tendresse et d'amour celle qui a été toujours là pour moi .

c'est grâce a celle que je suis ce que je suis.

A tous les membres de la famille grand et petit, mes frères SOUFIAN ,HAMZA, WALID, yazid.

Ma seule sœur. RITADJ HIBAT ALRAHMAN.

A ma grande mère que dieu la garde.

A mes oncles et mes tantes paternelle et maternelle.

A mes cousins et cousines maternelle et paternelle.

A toutes mes fammiles MAIZI .MELIANI

Ma binome AICHA KHANAFIF.

A mes amises , FATI,NADJIA,AMINA ,

zizo ,MADINA,AMEL ,CHERIFA,MASSOUDA…KHAOULA, ASSMA ,HOURIA

A tous mes amis mohamed 2bobaker, hamza,abd ELkarim,

A tous les étudiants de la faculté en génie électrotechnique surtout les étudiants de la 2eme année promotion 2013.

A tous les enseignants qui m'ont aidé de proche ou de loin pour électrotechnique d'option automatique.

A tous ceux que j'aime et qui m'aiment et tous ceux qui me connaissent.

Remerciement
Résumé
Liste des Figures
Liste des tableaux

Introduction Générale 1

CHAPITRE I : Régulation Industrielle : Contrôleur PID & contrôleur Flou

I.1-Introduction…… 2
I.2-Principe générale de régulation…..... 3
I.2.1-Comportement en régulation……. 3
I.2.2-Comportement en asservissement……. 4
I.3- La régulation en boucle ouverte…… 5
I.4- La régulation en boucle fermée……. 5
I.5- Les autres formes de régulation……. 6
I.5.1- Régulation en cascade…….. 7
I.5.2- Régulation mixte…….. 7
I.5.3- Régulation de rapport……... 8
I.6- Structure et comportement des processus…….. 8
I.6.1- Comportement statique des processus…….. 8
I.6.2- Comportement dynamique des processus…… 9
I.7- Les caractéristiques du régulateur PID…….. 11
I.7.1-Les actions PID…….. 11
I.7.2- Le régulateur PI…… 15
I.7.3. Le régulateur PID…….. 15
I.8-Les caractéristiques du régulateur « Flou »…… 16
I.8.1-Historique…….. 16
I.8.2- Les sous-ensembles flous……. 17
I.8.3-Notions de bases de la commande floue…… 18
I.8.4- Structure d'une commande floue……. 20
I.8.5- Quelques techniques de fuzzification et défuzzification…….. 20
I.8.6- Les régulateurs flous de type Mamdani…… 22
I.8.7- Les régulateurs flous de type SUGENO……..……. 23
I.9 – Conclusion……... 24

CHAPTRE II : Présentation du langage de programmation graphique LabVIEW

II.1- Introduction à la programmation graphique……. 25
II.1.1-programmation flot de contrôle……. 25
II.1.2-Programmation flot de donnée…….. 25
II.2- L'environnement LabVIEW……... 26
II.3- Structure de données manipulées via LabVIEW……. 28
II.3.1- le type tableau (structure de données homogènes) …….. 28
II.3.2- Le type «cluster» (structure de données hétérogènes) …… 29
II.4 - Structures de programmation…….. 29
II.4.1- Les structures itératives…… 29
II.5-Traitement numérique……... 31
II.5.1- les fonctions prédéfinies…….. 31
II.5.2 Boites à outils mathématique……. 32
II.6- Bibliothèques de commande……... 32
II.6.1-La bibliothèque de contrôle PID…… 33
II. 6. 2 - La bibliothèque de logique floue…… 34
II. 7- Conclusion…….. 35

Chapitre III : Description de régulation de niveau

III.1-Introduction……... 36
III. 2. Présentation générale de la maquette……..... 38
III.4. Modélisation mathématique de système…… 39
III. 5. Acquisition du paramètre : Débit de remplissage du réservoir……. 40
III. 5. 1. principe de fonctionnement de la carte NI 6008……... 40
III. 5. 2. Description de la Carte NI 6008…… 41
III.6. Conclusion…… 41

CHAPITRE IV : Résultat Et Discussion

IV.1- Introduction…….. 41
IV.2- Boucle de régulation……. 43

IV. 3- Phases de conception des deux contrôleurs…….. 43
IV.3.1-Configuration des paramètres du contrôleur PID…….. 48
IV.3.1.1-Résultat et simulation avec paramètre fixe……. 50
IV.3.2-Configuration des paramètres du contrôle flou……. 54
IV.4- Conclusion…….. 54
Conclusion générale 55

Remerciements

Au terme de ce Mémoire nous tenons à remercier tout naturellement en premier lieu **Allah** le tout Puissant qui nous a donné la force, le courage et la patience de bien mener ce travail.

Nous tenons à exprimer nos sincères remerciement tout d'abord à notre promoteur Monsieur Adnane. MESSAI.

Nous remercions Monsieur Nour eldine. D'avoir accepter d'être notre Co-promoteur.

Nous tenons tout d'abord à remercier le Département Electronique de CRNB (Centre de Recherche Nucléaire de Birine), dirigés par Mr **IDIR**, pour l'accueilli et la disposition un environnement propice à l'apprentissage de la recherche par la recherche. Cet environnement est bien entendu matériel mais surtout humain et nous veux profiter de ces quelques lignes pour exprimer notre reconnaissances à tous ceux qui ont, de près ou de loin, permis l'aboutissement de ce travail de Master II.

Nous adressons mes remerciements aussi à chef de département **Mr BEN TOUATI.**

Nous adressons mes remerciements aussi à mes enseignants de l'université de MEDEA «**Mr M.BOUHEDA et Mr S.MOULAHOUM, Mr TELEMCENI, Mr KABACHE, Mr MOUALDIA, Mr S.BOULKHRECHEF MADJBOURI…** », Pour la qualité de l'enseignement qu'ils nous ont bien voulu prodiguer durant nos études.

Un merci tout, qui a dû supporter au quotidien mes insomnies et mes coups de stress durant la phase de réalisation de ce travail.

Nous tenons à remercier particulièrement le président et les membres du jury d'avoir accepter de juger ce modeste travail, qu'ils trouvent ici notre sincère gratitude.

Nous terminons ces remerciements par la personne qui a été plus qu'un soutien **Mlle H.ASSEM** merci pour tous ce que vous avez faire pour nous.

LISTE DES FIGURES

CHAPITRE I

Figure I-1 : Schéma de principe d'une chaine de régulation…….(4).
Figure I.3 : Comportement en asservissement…….(5).
Figure I-2 : Comportement en régulation…….(5).

Figure I.4 : exemple de régulation en cascade et sans cascade…….(7).
FigureI.5fonctiondetransfertd'un processus……...(8).
Figure I.6 : Comportement statique d'un processus……..(9).

Figure I.7 : Comportement d'un processus naturellement stable……..(9).

Figure I.8 : Comportement d'un processus naturellement instable…….....(10).

Figure I.9: Schéma synoptique d'un régulateur PID……..(11).
Figure. I.10 : Les fonctions d'appartenances en logique classique……(17).
Figure. I.11 : Les fonctions d'appartenances en logique floue……...(18).
Figure I.12 : Opérateur NON……..(18).
Figure I.13 : Opérateur ET……(19).
Figure I.14 : Opérateur OU……..(19).

Figure I.15 : Structure d'une commande floue……(20).

Figure I.16 : Méthode de fuzzification pour une mesure exacte…….....(21).
Figure I.17 : Méthode de fuzzification pour une mesure incertaine…….....(21).
Figure I.18 : Défuzzification par centre de gravité……(21).
Figure I.19 : Défuzzification par valeur maximum……(22).

Figure I.20 : Structure d'un PI flou de type Mamdani…….....(23).

CHAPITRE II

Figure II.1 : Exemple de diagramme de flot de données…….....(26).
Figure II.2 : Fenêtre de l'environnement de développement sur LabVIEW
Face avant (à droite) et Diagramme (à gauche)…….(26).

Figure II. 3 : Illustration de la palette d'outils……(27).

Figure II. 4 : Illustration de la palette des commandes…….....(27).
Figure II.5 : Illustration de la palette des fonctions…….(27).
Figure II. 6 : Les différents types de structures de données dans LabVIEW..(28)
Figure II. 7 : Exemple d'initialisation d'un tableau à 2 dimensions (matrice 5x3)
……..(28)
Figure II. 8 : Exemple de manipulation des données avec le type cluster……. (29).
Figure II.9 : Exemple d'utilisation de la structure itérative "pour"……..(30).
Figure II.10 : Exemple d'utilisation de la structure itérative "tant que"……..(30).
Figure II.11 : Exemple d'utilisation des registres à décalage dans une boucle "pour"
…….....(31).
Figure II.12 : Quelques instructions de traitement de données numérique
DansLabVIEW……(31).
Figure II.13 : Exemple de boite de calcul mathématique da LabVIEW.……....(32).
Figure II.14 : Exemple de fenêtre d'aide offrant la description d'une fonction
fournie dans la bibliothèque de LabVIEW…….(3 2).
Figure II.15 : Exemple de bibliothèque de régulation PID……...(33).
Figure II.16 : Exemple de Bibliothèque de la commande floue……...(34).
Figure II.17 : Interface graphique pour la conception d'une commande floue….(34).
CHAPITRE III
FigureIII.1.Schémablocdusystèmeglobal…….(36).
FigureIII.2.Structuredelamaquette …….(37) .
FigureIII. .3.schémafonctionnelledeprocessus……(39).
Figire III.4 : Vue perspective de la carte NI 6008……(40) .
CHAPITRE IV

Figure IV.1 :Diagramme de commande tel qu'il a été simule

parLAbView …….(42).

Figure IV.2 Page de commande sur l'interface grafique ……(43).

FigureIV.3Choixfinaldesparametresducontroleur PID…….…….(46).

FigureIV.4réponsedusystèmeapreschoixfinaldesparamètres…… …(46).

Figure IV.5 : réponse du système à une consigne de 3 m…….(47).

Figure IV.6 réponse du système après un changement de consigne de 3m à 4m (48) .

Figure IV.7 : réponse du système après une variation le débit de la pompe…….(48).

Figure IV.8 : Fonction d'appartenance pour l'erreur…….....(50).

Figure IV.9 : Fonction d'appartenance pour la variation de l'erreur……...(50).

Figure IV.10 : Fonction d'appartenance pour variation de la commande…….....(50).

Figure IV.11: réponse du système à un consigne de 3 m……(52).

Figure IV.12 : réponse du système après un changement de consigne

de 3m à 4 m……..(53).

LISTE DU TABLEAU

Tableau IV.1 :Operation du choix des parammétre du controleur PID ……..43

TableauIV.2 :Base de régle flou……...51

الملخص

الهدف من هذه الأطروحة تقديم لمحة عامة عن وحدة تحكم PID والسيطرة المنطق التشخيصيlogique flou مع ضمان مقاربة منهجية لعمل مشتق وفقا لأهداف تنظيمية. الحل المقترح يمكن أن تحقق الأنشطة التعليمية حول تخزين وتوزيع المياه.

في هذا العمل، ونحن ندرس في محاكاة LABVIEW تطبيق مراقبة لعملية هيدروليكية: تنظيم مستوى المياه في خزان مجهز مع جهاز إرسال مستوى لصمام الآلية و مضخة جار الماء. وقد أجريت دراسة مقارنة بين السيطرة المنطق التشخيصي والتحكم PID تقليدي لعملية إنتاج تغيرات كبيرة في قيمة مستوى الماء(لاختبار اليات التحكم« PID ,LOGIQUE FLOU »)

في هذا المشروع استخدمنا NI6008لمنح لرسم خريطة للإشارات الناتجة.

RESUME

L'objectif de ce mémoire a donc été de proposer une synthèse pour le régulateur PID et la commande par logique floue tout en garantissant une approche systématique de l'action dérivée selon des objectifs de régulation. Le produit proposé permet de réaliser des activités pédagogiques autour du stockage et de la distribution d'eau.

Dans le présent travail, nous étudions en simulation sur LABVIEW l'application d'une commande floue à un procédé hydraulique : une régulation du niveau de l'eau dans un réservoir équipée d'un transmetteur de niveau d'une vanne motorisée et d'une pompe de chargement d'eau. Une étude comparative entre la commande floue et la commande classique PID a été effectuée pour le procédé en produisant des grands changements de consigne (afin de tester la robustesse des deux commandes vis-à-vis d'une non linéarité).

Dans ce projet on a utilisé la carte NI 6008 pour acquises les signaux.

Le mots clés : PID , LOGIQUE FLOU,LABVIEW,CONSIGNE

Abstract

The objective of this thesis has been to provide an overview for the PID controller and fuzzy logic control while ensuring a systematic approach to the derivative action according to regulatory objectives. The proposed product can achieve educational activities around the storage and distribution of water.

In this work, we study in simulation LABVIEW application of fuzzy control to a hydraulic process: regulation of water level in a tank equipped with a level transmitter for a motorized valve and a water-loading pump. A comparative study between fuzzy control and conventional PID control was carried out for the process producing large changes in value (to test the robustness of two commands in this project we used the NI 6008 to map the signals acquired.

Introduction Générale

Pour être rentable et compétitif, un procédé industriel doit être obligatoirement automatisé. En effet, la compétitivité exige de le maintenir le plus près possible de son optimum de fonctionnement prédéfini par un cahier des charges : conditions ou performances imposées telles que la qualité des produits fabriqués, la flexibilité de la production, la sécurité du personnel et des installations, l'économie de l'énergie et le respect de l'environnement.

La régulation industrielle, ou l'asservissement, occupent généralement la grande partie de la tâche de l'automatisation d'un procédé industriel. Le travail de fin d'étude qui nous a été proposé est : l'étude, la conception et la mise au point d'un contrôleur flou pour la régulation de niveau d'un fluide stocké dans un réservoir avec appoint. Le premier objectif de ce travail consistait donc en la modélisation, et la simulation processus à commander en utilisant un langage de programmation graphique en l'occurrence : ***LABView***. La deuxième étape, serait l'adjonction d'un contrôleur flou pour la surveillance en ligne du niveau du fluide considéré.

Les résultats qui s obtenons via ce contrôleur seront validés à travers une comparaison avec ceux qui seront éventuellement obtenus avec un contrôleur PID classique. Pour cela, nous aurions également à implémenter, en parallèle du contrôleur principal, i. e. le contrôleur flou, un autre contrôleur PID dont les paramètres seront fixés selon les exigences du processus à commander.

Aussi, pour tester quelques performances dynamiques de notre contrôleur en asservissement, nous aurions à introduire une perturbation externe au système global considéré (Contrôleur + processus à commander). Comme perturbation, nous avons choisi : la variation du débit volumique du remplissage du réservoir : $\boldsymbol{Q_{in}}$. Cette donné externe sera injecté au processus, simulé sur ordinateur, à l'aide d'une carte d'acquisition de données (de National Instruments : NI6008). Cette même carte, sera également utilisée, dans un futur proche, pour piloter le circuit de commande de la pompe du remplissage.

Par conséquent, le contenu de ce mémoire sera structuré comme suit :

Dans un premier chapitre, nous présentons d'une manière laconique les différentes méthodes de régulation industrielle que ça soit en boucle ouverte et en boucle fermée. Nous essayerons aussi de mettre l'accent plus particulièrement sur les contrôleurs flous ainsi que les contrôleurs classique PID.

Quant au deuxième chapitre, celui-ci sera consacré à l'outil de développement utilisé pour la mise au point de ce projet : Le logiciel de programmation graphique « LabVIEW ». Lors de cette partie, on mettra en relief les fonctions et les bibliothèques liées au contrôle-commande.

Au troisième chapitre, nous abordons la description détailles du travail réalisé. Il est question de fournir les détails nécessaires (hardware et software) requis pour l'implémentation d'un système de régulation du niveau.

Arrivés au quatrième chapitre, on s'attaque aux tests de simulation sur micro-ordinateur du système ainsi implémenté. C'est une sorte de validation par comparaison des deux contrôleurs (PID et logique floue) quant à la régulation du niveau dans un réservoir donné.

En fin, l'actuel document clôture par une conclusion générale, au cours de celle-ci on essayée de dresser un bilan sur l'ensemble des tâches réalisées, les résultats obtenus et les perspectives et les extensions éventuelles pour le présent travail.

CHAPITRE I : Régulation Industrielle : Contrôleur PID & contrôleur Flou.

I.1- Introduction :

Réguler une grandeur, c'est obtenir d'elle un comportement donné, dans un environnement susceptible de présenter des variations. On ne peut pas parler de principe de régulation sans parler des lois de commandes. En faite, les grandeurs physiques commandées varient continument dans le temps. Pour celles qui ne présentent que 2 états (système binaire ou ≪tout ou rien ≫, tel les feux de signalisation, les commandes d'ascenseurs, de transfert de pièces par convoyeurs, etc..) en utilise une autre approche différente à la structure de boucle utilise dans la plupart des systèmes y compris notre système (réservoir).

Les systèmes automatiques assurent en fait 2 types de fonctions :

- Maintenir la grandeur commandée, ou grandeur réglée, à une valeur de référence malgré les variations de conditions extérieures ; on parle de la régulation en sens strict,
- Répondre à des changements d'objectif, ou à un objectif variable tel-que la poursuite de cible, on parle d'un fonctionnement d'asservissement.

I.2- Principe général de la régulation :

Dans la plupart des appareils et installations industrielles, tertiaires et mêmes domestiques, il est nécessaire de maintenir des grandeurs physiques à des valeurs déterminées, en dépit des variations externes ou internes influant sur ces grandeurs. Le niveau d'un réservoir d'eau, la température d'une étuve, le débit d'une conduite de gaz, étant par nature variables, doivent donc être règles par des actions convenables sur le processus considère. Si les perturbations influant sur la grandeur à contrôler sont lentes ou négligeables, un simple réglage (dit en boucle ouverte) permet d'obtenir et de maintenir la valeur demandée (par exemple : action sur un robinet d'eau). Dans la majorité des cas, cependant, ce type de réglage n'est pas suffisant, parce que trop grossier ou instable. Il faut alors comparer, en permanence, la valeur mesurée de la grandeur réglée à celle que l'on souhaite obtenir et agir en conséquence sur la grandeur d'action, dite grandeur réglant. On a, dans ce cas, constitue une boucle de régulation et plus généralement une boucle d'asservissement. Cette boucle nécessite la mise en ouvre d'un ensemble de moyens de mesure, de traitement de signal ou de calcul, d'amplification et de commande d'actionneur, constituant une chaine d'éléments associes : ***la chaine de régulation***.

Toute chaine de régulation (ou d'asservissement) comprend trois maillons indispensables : l'organe de mesure, l'organe de régulation et l'organe de contrôle. Il faut donc commencer par mesurer les principales grandeurs servant à contrôler le processus. L'organe de régulation récupère ces mesures et les compare aux valeurs souhaitées, plus communément appelées valeurs de consigne. En cas de non concordance des valeurs de mesure et des valeurs de consigne, l'organe de régulation envoie un signal de commande à l'organe de contrôle (vanne, moteur, etc.), afin que celui-ci puisse agir sur le processus. Les paramètres qui régissent le processus sont ainsi stabilisés en permanence à des niveaux souhaités.

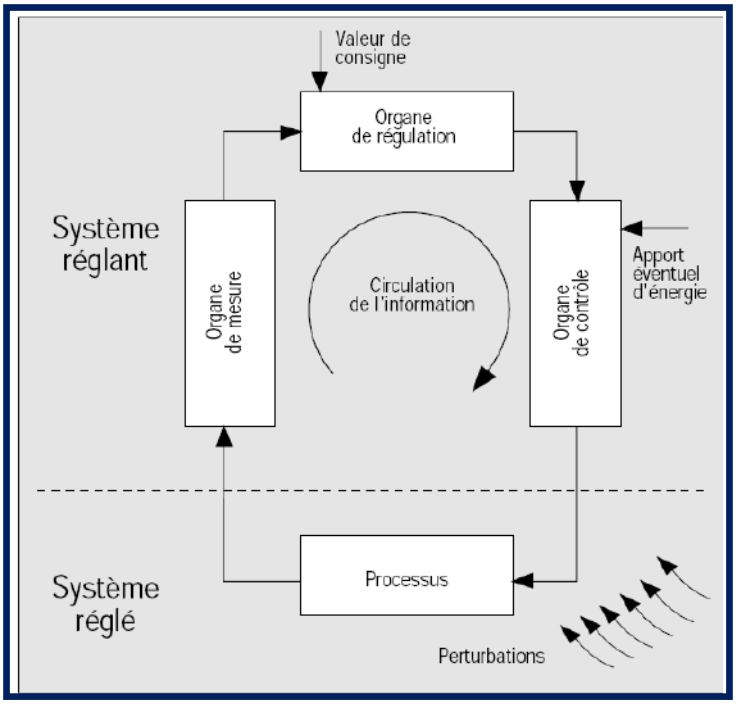


Figure I-1 : Schéma de principe d'une chaine de régulation.

Le choix des éléments de la chaine de régulation est dicté par les caractéristiques du processus à contrôler, ce qui nécessite de bien connaitre le processus en question et son comportement.

I.2.1- Comportement en régulation :

Faire varier le débit

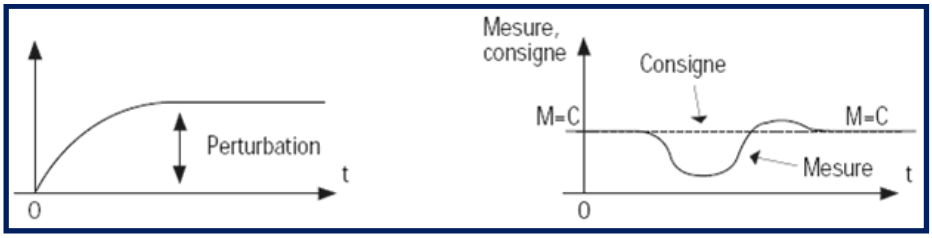


Figure I-2 : Comportement en régulation.

I.2.2- Comportement en asservissement :

Faire varier la consigne sp

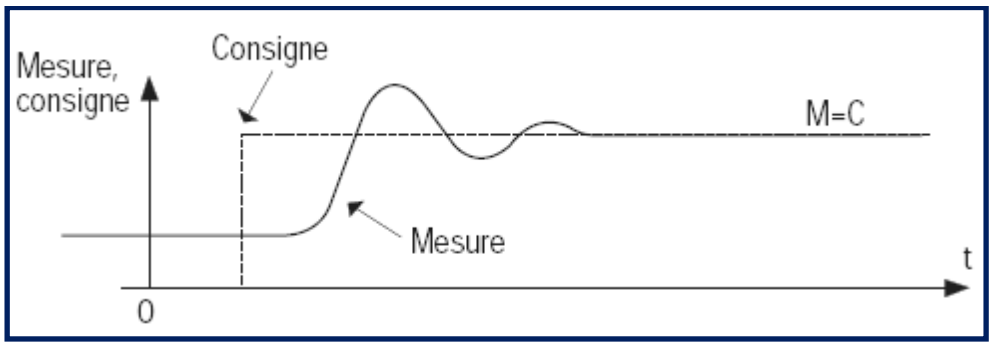


Figure I.3 : Comportement en asservissement.

I.3- La régulation en boucle ouverte :

Dans un asservissement en boucle ouverte, l'organe de contrôle ne réagit pas à travers le processus sur la grandeur mesurée (celle-ci n'est pas contrôlée). Une régulation en boucle ouverte ne peut être mise en œuvre que si l'on connait la loi régissant le fonctionnement du processus (autrement dit, il faut connaitre la corrélation entre la valeur mesurée et la grandeur réglant). Contrairement à un asservissement en boucle fermée, un asservissement en boucle ouverte permet d'anticiper les phénomènes et d'obtenir des temps de réponse très Courts. De plus, il n'y à pas d'oscillation à craindre (car il s'agit d'un système dynamiquement stable). Enfin, l'asservissement en boucle ouverte est la seule solution envisageable lorsqu'il n'y à pas de contrôle final possible.

Au niveau des inconvénients, la régulation en boucle ouverte impose de connaitre la loi régissant le fonctionnement du processus, et il est très fréquent que l'on ne Connaisse pas la loi en question. Autre inconvénient sérieux, il n'y à aucun moyen de contrôler, à plus forte raison de compenser, les erreurs, les dérives, les accidents qui peuvent intervenir à l'intérieur de la boucle ; autrement dit, il n'y à pas de précision ni surtout de fidélité qui dépendent de la qualité intrinsèque des composants. Enfin, la régulation en boucle ouverte ne compense pas les facteurs perturbateurs.

I.4- La régulation en boucle fermée :

Dans ce qui vient d'être dit, la variable de sortie (de la chaine de régulation), ou grandeur réglant, exerce une influence sur la valeur de la variable d'entrée (de la chaine de régulation) ou variable contrôlée, pour la maintenir dans des limites définies : il s'agit d'une régulation ou d'un asservissement en boucle fermée. L'action de la grandeur réglant sur la variable contrôlée s'opère à travers le "processus" qui boucle la chaine. Dans une régulation en boucle fermée, une bonne partie des facteurs perturbateurs sont automatiquement compensés par la contre-réaction à travers le procède. Autre avantage, il n'est pas nécessaire de connaitre avec précision les lois, le comportement des différents composants de la boucle, et notamment du processus, bien que la connaissance des allures statistiques et dynamiques des divers phénomènes rencontrés soit utile pour le choix des composants.

Parmi les inconvénients d'une régulation en boucle fermée, il faut citer le fait que la précision et la fidélité de la régulation dépend de la fidélité et de la précision sur les valeurs mesurées et sur la consigne. Autre inconvénient, sans doute plus important, le comportement dynamique de la boucle dépend des caractéristiques des différents composants de la boucle, et notamment du processus, enfaite un mauvais choix de certains composants peut amener la boucle à entrer en oscillation.

Enfin, la régulation en boucle fermée n'anticipe pas. Pour que la régulation envoie une commande à l'organe de contrôle, il faut que les perturbations ou les éventuelles variations de la valeur de consigne se manifestent sur la sortie du processus : ceci peut exiger un délai parfois gênant.

I.5- Les autres formes de régulation :

Le but d'une chaine de régulation est de contrôler un processus. Au niveau des organes de mesure et de contrôle, il n'a pas une très grande marge de manœuvre, car ces organes, dans une certaine mesure, s'imposent souvent d'eux-mêmes.

Il reste un domaine ou son savoir-faire va s'exercer pleinement : c'est au niveau de l'organe de régulation. Bien entendu, les caractéristiques de cet organe vont dépendre du processus à contrôler, des perturbations à prendre en compte, des caractéristiques des organes de mesure et de contrôle. Bien souvent, les systèmes de régulation comportent, au lieu des chaines linéaires ouvertes ou fermées, des ensembles des boucles imbriquées dont tout ou partie peut induire des contre-réactions à travers le processus.

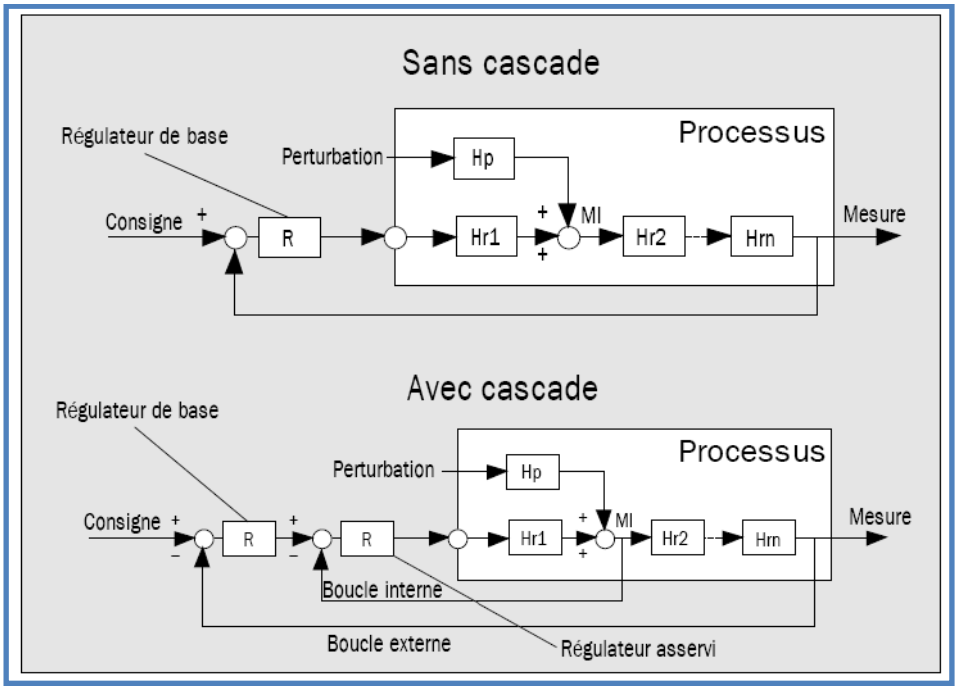


Figure I.4 : exemple de régulation en cascade et sans cascade.

I.5.1- Régulation en cascade :

L'objectif d'une régulation en cascade est de minimiser les effets d'une ou de plusieurs grandeurs perturbatrices qui agissent soit sur la variable réglant, soit sur une grandeur intermédiaire se trouvant en amont de la variable à régler. Ce type de régulation est intéressant lorsque l'on à affaire à des processus à longs temps de réponse. En effet, quand une perturbation se manifeste, il est nécessaire d'attendre que son influence se ressente au niveau de l'organe de mesure place en sortie de chaine. Si les temps de réponse sont longs, la correction n'intervient donc que tardivement, parfois avec la cause qui la produite et dont le sens s'est inversé, provoquant oscillations, instabilité, etc.

Bien évidemment, la régulation en cascade n'apporte aucune amélioration si la grandeur perturbatrice se produit en aval de la mesure intermédiaire. Pour que la cascade soit justifiée, il faut que la boucle interne soit beaucoup plus rapide que la boucle externe.

I.5.2- Régulation mixte :

Ce type de régulation est l'association d'une régulation en boucle fermée et d'une régulation en boucle ouverte. Les deux boucles sont complémentaires et elles associent leurs actions pour améliorer la stabilité globale. Ce type de régulation est à mettre en œuvre lorsqu'une perturbation affecte directement la grandeur à régler.

I.5.3- Régulation de rapport :

Ce type de régulation a par exemple pour objectif d'asservir un débit Q_a *à* un autre débit libre Q_1 en imposant entre ces deux débits un facteur de proportionnalité K_d, fixe manuellement ou automatiquement.

I.6- Structure et comportement des processus :

Si un processus possède une grandeur réglant *U* et une grandeur réglée *S*, son comportement peut être représente soit par une équation différentielle reliant les valeurs de *S* et de *U* en fonction du temps, soit par une représentation dite fonction de transfert déduite de la transformation de Laplace appliquée à cette équation différentielle.

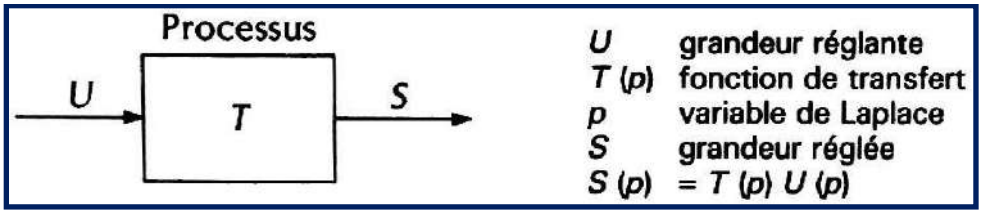


Figure I.5 : fonction de transfert d'un processus.

Dans ce dernier cas, le comportement du processus est décrit par la relation :

$$S(p) = T(p)U(p) \qquad (I \cdot 1)$$

Dans laquelle *T* (*p*) est la fonction de transfert considérée, *p* la variable de *Laplace*. Toutefois, ces relations s'appliquent seulement à des systèmes dont les variables varient de manière continue, et dans lesquels une grandeur réglant *U* n'exerce son action que sur une seule grandeur réglée *S* : il s'agit donc de processus mono variables continus. A noter ici que les processus industriels, en fait, ne sont pas toujours continus et peuvent être multi-variables.

I.6.1- Comportement statique des processus :

Lorsque le processus est dans un état stable, à chaque valeur U de la grandeur réglante correspond une valeur *S* de la grandeur réglée. Deux valeurs voisines U_1 et U_2 correspondent deux valeurs S_1 et S_2 de la grandeur réglée, telles que :

$$\frac{S_2 - S_1}{U_2 - U_1} = G_0 \qquad (I \cdot 2)$$

G_0 est appelé gain statique du processus. La valeur de celui-ci peut dépendre du point de fonctionnement considéré et n'est donc pas forcement constante dans tout le domaine de

variation des grandeurs du processus. On dit alors que le processus est non linéaire. La courbe caractéristique $S(u)$ peut elle-même dépendre d'autres paramètres. Sur la figure (I. 6) on a représenté un réseau de courbes $\boldsymbol{S(u)}$ dépendant d'un paramètre α. Il arrive aussi que le gain statique dépende du sens de l'action de la grandeur réglant : on à alors affaire à un phénomène d'hystérésis.

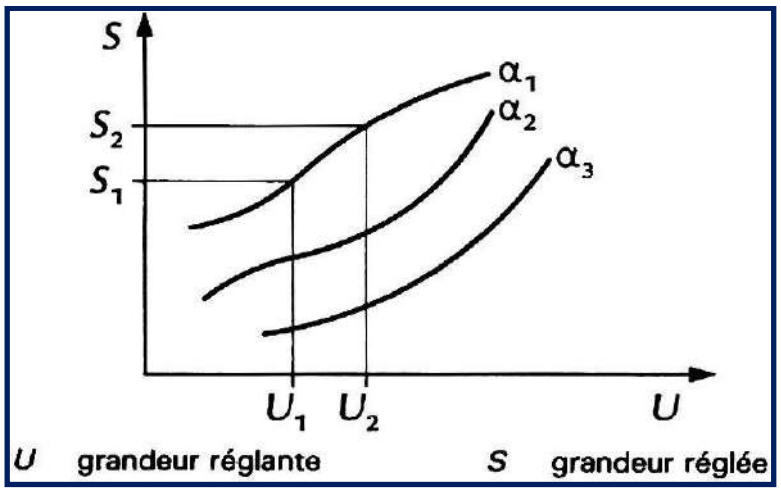


Figure I.6 : Comportement statique d'un processus.

I.6.2- Comportement dynamique des processus :

- ***Processus naturellement stables :***

La réponse naturelle d'un processus à une variation en échelon de sa grandeur réglant permet de savoir s'il est naturellement stable ou instable. Dans le cas de **procédés stables** (tels que fours et étuves, par exemple), une excitation ΔU de la grandeur réglant produit, à la fin du régime transitoire correspondant, une variation limitée ΔS de la grandeur réglée

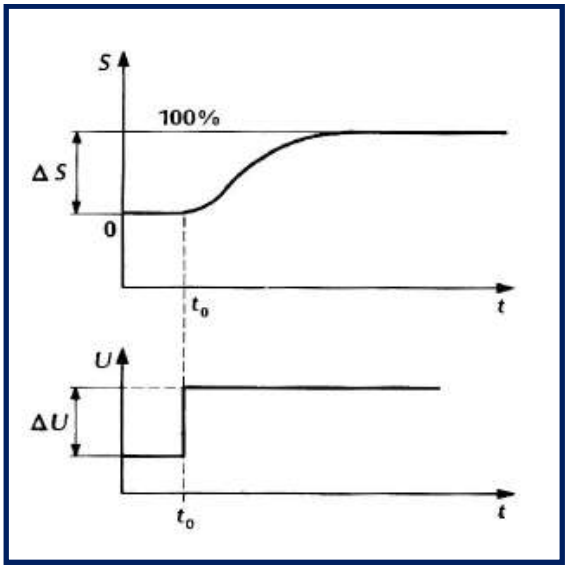


Figure I.7 : Comportement d'un processus naturellement stable.

Dans la majorité des cas, les régimes transitoires associes à ce type de réponse ne comportent ni oscillation ni dépassement de la valeur finale ; ces processus sont dits **stables apériodiques**.

Il est possible de représenter ces processus par une relation mathématique générale de la forme :

$$\boldsymbol{G(p) = \frac{G_0}{(1+\theta_1 P)(1+\theta_2 P)\cdots(1+\theta_n p)}} \qquad (\boldsymbol{I \cdot 3})$$

❖ ***Processus naturellement instables :***

Dans un processus naturellement instable, une excitation ΔU de la grandeur réglant produit une variation ΔS de la grandeur réglée dépendant continument du temps, donc illimitée.

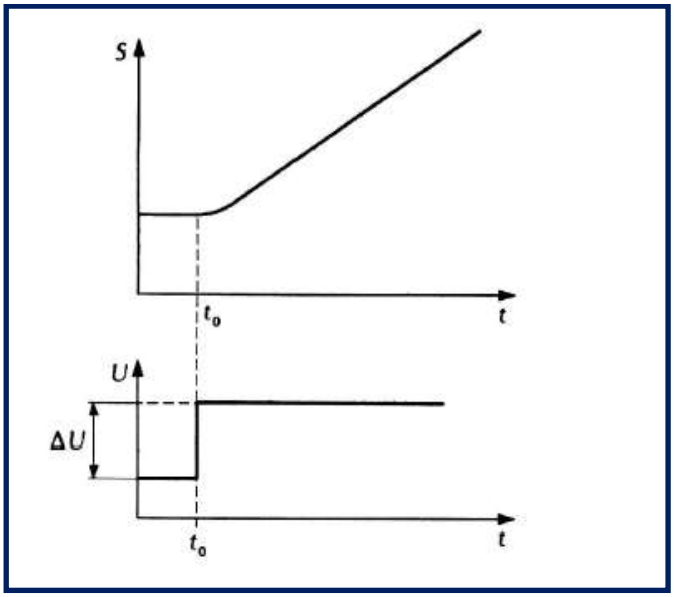


Figure I.8 : Comportement d'un processus naturellement instable.

Un exemple type de ces processus est la commande du niveau d'eau dans un réservoir, à partir d'un débit introduit ou soutiré. Si le débit entrant ou sortant est modifié d'une valeur constante, alors que le niveau était auparavant stabilisé, celui-ci augmentera ou diminuera jusqu'au débordement ou à la vidange du réservoir.

Il est possible, dans la plupart des cas, de représenter ces processus par une relation de type :

$$G(p) = \frac{1}{P} G_1(p) \qquad (I \cdot 4)$$

Avec $G_1(p)$ une fonction de transfert quelconque du type apériodique. Processus qui comporte naturellement un intégrateur pur.

I.7- Les caractéristiques du régulateur PID :

Le régulateur standard le plus utilisé dans l'industrie est le régulateur ***PID*** (proportionnel intégral dérive), car il permet de régler à l'aide de ses trois paramètres les performances (amortissement, temps de réponse, ...) d'un processus modélisé par un deuxième ordre. Nombreux sont les systèmes physiques qui, même en étant complexes, ont un comportement voisin de celui d'un deuxième ordre. Par conséquent, le régulateur PID est bien adapté à la plupart des processus de type industriel et est relativement robuste par rapport aux variations des paramètres du procédé, quand on n'est pas trop exigeant sur les performances de la boucle fermée par rapport à celles de la boucle ouverte (par exemple, accélération très importante de la réponse ou augmentation très importante de l'amortissement en boucle fermée).

Si la dynamique dominante du système est supérieure à un deuxième ordre, ou si le système contient un retard important ou plusieurs modes oscillants, le régulateur PID n'est plus adéquat et un régulateur plus complexe (avec plus de paramètres) doit être utilise, au dépend de la sensibilité aux variations des paramètres du procédé.

I.7.1-Les actions PID :

En pratique, à une catégorie donnée de systèmes à asservir correspond un type de correcteur adopte. Pour effectuer un choix judicieux, il faut connaitre les effets des différentes actions : proportionnelle, intégrale et dérivée.

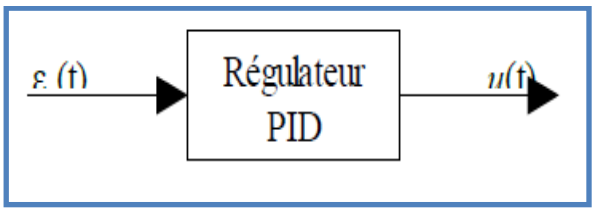


Figure I.9: Schéma synoptique d'un régulateur PID.

Un régulateur PID est obtenu par l'association de ces trois actions et il remplit essentiellement les trois fonctions suivantes :

1. Il fournit un signal de commande en tenant compte de l'évolution du signal de sortie par rapport à la consigne
2. Il élimine l'erreur statique grâce au terme intégrateur
3. **Il anticipe les variations de la sortie grâce au terme dérivateur.**

La commande $U(t)$ donnée par le régulateur PID, dans sa forme classique est décrite par :

$$U(t)_{=}\ K_P\left[\varepsilon(t)+\frac{1}{T_i}\int_0^t \varepsilon(\tau)\,d\tau+T_d\,\frac{d\varepsilon(t)}{dt}\right] \qquad (\mathrm{I}\cdot 5)$$

Elle est la somme de trois termes :

- ***Le terme proportionnel :***

$$\mathrm{P}=K_P\varepsilon(t) \qquad (\mathrm{I}\cdot 6)$$

- ***Le terme intégral :***

$$\mathrm{I}=K_P\frac{1}{T_i}\int_0^t \varepsilon(\tau)d\tau \qquad (\mathrm{I}\cdot 7)$$

- ***Le terme dérivatif :***

$$\mathrm{D}=K_P T_d\,\frac{d\varepsilon(t)}{d_t} \qquad (\mathrm{I}\cdot 8)$$

Les paramètres du régulateur PID sont le gain proportionnel $\boldsymbol{K_p}$, le temps intégral $\boldsymbol{T_i}$ et le temps dérivatif $\boldsymbol{T_d}$, les temps étant exprimés en secondes.

I.7.1.1- L'action proportionnelle :

La sortie $U(t)$ du régulateur proportionnel est donnée en fonction de son entrée $\mathcal{E}(t)$ qui représente l'écart entre la consigne et la mesure par la relation :

$$u(\mathrm{t})=K_p.\varepsilon(t) \qquad (\mathrm{I}\cdot 9)$$

Pour le cas discret, cette relation reste la même telle que :

$$u(\mathrm{k})=K_p.\varepsilon(k) \qquad (\mathrm{I}\cdot 10)$$

Le rôle de l'action proportionnelle est de minimiser l'écart $\boldsymbol{\mathcal{E}}$ entre la consigne et la mesure et elle réduit le temps de montée et le temps de réponse. On constate qu'une augmentant du gain $\boldsymbol{K_p}$ du régulateur entraine une diminution de l'erreur statique et permet d'accélérer le comportement global de la boucle fermée. On serait tenté de prendre des valeurs de gain élevées pour accélérer la réponse du procédé mais on est limité par la stabilité de la boucle fermée. En effet, une valeur trop élevée du gain augmente l'instabilité du système et donne lieu à des oscillations.

I.7.1.2- L'action dérivée :

Elle est une action qui tient compte de la vitesse de variation de l'écart entre la consigne et la mesure, elle joue aussi un rôle stabilisateur, contrairement à l'action intégrale.

En effet, elle délivre une sortie variant proportionnellement à la vitesse de variation de l'écart $\boldsymbol{\varepsilon}$:

$$u(t) = T_d \frac{d_\varepsilon(t)}{dt} \qquad (\mathrm{I} \cdot 11)$$

Avec T_d le dosage de l'action dérivée, exprimé en minutes ou en secondes. L'action dérivée va ainsi intervenir uniquement sur la variation de l'erreur ce qui augmente la rapidité du système (diminution des temps de réponses). L'action dérivée permet aussi d'augmenter la stabilité du système par apport de phase (+π/2 ce qui augmente la marge de phase). L'annulation de cette action en régime statique impose donc de ne jamais l'utiliser seule : l'action dérivée n'exerce qu'un complément à l'action proportionnelle. 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. Lorsque la période d'échantillonnage $\boldsymbol{T_e}$ est petite, la différence vers l'arrière (*Approximation d'Euler rétrograde*) nous permet d'approcher la dérivée d'un signal à temps continu par :

$$\frac{df(t)}{dt} = \frac{f(t) - f(t - T_e)}{T_e} \qquad (\mathrm{I} \cdot 12)$$

Dans le domaine de *Laplace*, l'opération de dérivation se traduisait par une multiplication par la variable de Laplace $\boldsymbol{p}$ en continu. Dans le cas discret, en appliquant la transformée en $\boldsymbol{z}$ à l'équation (I. 12) on obtient :

$$Z\left\{\frac{f(t) - f(t - T_e)}{T_e}\right\} = \frac{1 - Z^{-1}}{T_e} F(z) \qquad (\mathrm{I} \cdot 13)$$

Ceci conduit à établir l'équivalence linéaire entre la variable de *Laplace* $\boldsymbol{p}$ et la variable $\boldsymbol{z}$:

$$P \longleftrightarrow \frac{1}{T_e} \cdot (1 - Z^{-1}) \qquad (\mathrm{I} \cdot 14)$$

Donc en discret, le terme dérive peut être remplace par :

$$\boldsymbol{\mu(k) = \frac{T_d}{T_e} \cdot \big(\varepsilon(k) - \varepsilon(k-1)\big)} \qquad \boldsymbol{(\mathrm{I} \cdot 15)}$$

I.7.1.3- L'action intégrale :

L'action intégrale agit proportionnellement à la surface de l'écart entre la consigne et la mesure, et elle poursuit son action tant que cet écart n'est pas nul. On dit que l'action

intégrale donne la précision statique (*Elle annule l'erreur statique*). L'action intégrale est conditionnée par le temps d'intégrale $\boldsymbol{T_i}$.

$$u(t) = \frac{1}{T_i}\int_0^t \varepsilon(\tau)\cdot d\tau \qquad (I\cdot 16)$$

Comme dans le cas de l'action proportionnelle, un dosage trop important de l'action intégrale engendre une instabilité de la boucle de régulation. Pour son réglage, il faut là aussi trouver un compromis entre la stabilité et la rapidité. L'ajout du terme intégral permet d'améliorer la précision mais en contrepartie, il introduit malheureusement un déphasage de (- π : ce qui risque de rendre le système instable du fait de la diminution de la marge de phase. Enfin, le correcteur intégral présente le défaut de saturer facilement si l'écart ne s'annule pas rapidement, ce qui est le cas des systèmes lents. En effet, tout actionneur est limité : un moteur est limité en vitesse, une vanne ne peut pas être plus que totalement ouverte ou totalement fermée. Il se peut que la variable de commande amène l'actionneur à sa limite ce qui suspend la boucle de retour et le système aura une configuration assimilable à une boucle ouverte puisque l'actionneur demeurera saturé indépendamment de la sortie du système.

Quand l'erreur est réduite (*action intégrale non saturée*), il se peut qu'il faille un temps important pour que les valeurs des variables ne soient correctes de nouveau : on appelle ce phénomène l'emballement du terme intégral. Pour l'éviter, on peut par exemple :

- soit suspendre l'action intégrale quand la commande est saturée ;
- soit appliquer une méthode d'anti-saturation, qui consiste à recalculer le terme intégral pour ne pas saturer.

Pour le cas discret, le terme intégral peut être remplace par la somme des écarts et la différentielle ***dt*** par $\boldsymbol{T_e}$ ce qui nous donne le résultat suivant :

$$u(n) = \frac{T_e}{T_i}\sum_{k=0}^{n}\varepsilon(k) = u(n-1) + \frac{T_e}{T_i}\varepsilon(n) \qquad (\mathrm{I}\cdot 17)$$

$\boldsymbol{T_i}$: Constante de temps intégrale et s'exprime en (s).

Le problème est que lorsque $\mathcal{E}$ redevient nul, le signal *U* peut avoir atteint une valeur trop élevée pour qu'il y ait équilibre. Autrement, elle permet d'éliminer l'erreur résiduelle en régime permanent.

En résumé, nous affirmons que l'action intégrale est utilisée lorsqu'on désire avoir en régime permanent, une précision parfaite, en outre, elle permet de filtrer la variable à régler d'ou l'utilité pour le réglage des variables bruitées telles que la commande.

I.7.2- Le régulateur PI :

Le correcteur intégral est en général associe au correcteur proportionnel, il élabore alors une commande qui peut être donnée par la relation suivante :

$$u(t) = K_p(\varepsilon(t) + \frac{1}{T_i}\int_0^t \varepsilon(\tau)d\tau) \qquad (\text{I}.18)$$

La fonction de transfert du correcteur est alors donnée par :

$$C(p) = K(p)\frac{1+T_iP}{Ti\ P} \qquad (I \cdot 19)$$

Pour un régulateur intégral pur, le régime dynamique est relativement long. D'un autre coté, le régulateur proportionnel réagit immédiatement aux écarts de réglage, mais, il n'est pas en mesure de supprimer totalement l'erreur statique. La combinaison des actions proportionnelle et intégrale permet d'associer l'avantage du régulateur ***P***, c'est-a-dire la réaction rapide à un écart de réglage, à l'avantage du régulateur ***I*** qui est la compensation exacte de la grandeur pilote. La transposition de correcteurs continus consiste à discrétiser un correcteur continu afin de l'utiliser dans une commande numérique.

En utilisant cette équivalence dans l'équation (I. 20), on obtient le correcteur ***PI*** discret:

$$C(z) = \frac{U(z)}{\varepsilon(z)} = \frac{K_P + K_i - K_P.Z^{-1}}{1 - Z^{-1}} \qquad (\text{I} \cdot 20)$$

D'ou on obtient l'algorithme de commande du régulateur ***PI*** :

$$□(k) = □(k-1) + (K_P + K_i) \cdot \varepsilon(k) - k_p \cdot \varepsilon(k-1) \qquad (\text{I} \cdot 21)$$

I.7.3. Le régulateur PID :

L'action conjuguée ***PID*** permet une régulation optimale en associant les avantages de chaque action : la composante ***P*** réagit à l'apparition d'un écart de réglage, la composante ***D*** s'oppose aux variations de la grandeur réglée et stabilise la boucle de régulation et la composante ***I*** élimine l'erreur statique. Et c'est pour cela que ce type de correcteur est le plus

utilise en milieu industriel. Dans un régulateur ***PID***, il existe plusieurs façons d'associer les paramètres ***P***, ***I*** et ***D***. En effet, le correcteur PID peut avoir une structure série, parallèle ou mixte.

I.8-Les caractéristiques du régulateur « Flou » :

La logique floue a été introduite en 1965 par **L.A. Zadeh** [3]. Ses principes ont été appliqués en 1974 par E.H. Mamdani [4] à la construction d'un premier contrôleur flou. Mais ce n'est que depuis quelques années que la commande floue a connu, essentiellement au Japon, un essor remarquable ; elle a été appliquée à des problèmes aussi divers que la purification de l'eau, la fabrication du ciment, la marche automatisée d'une rame de métro, etc.. [5].

L'attitude des automaticiens à l'égard de la commande floue a d'abord été réservée : après avoir, pendant des décennies, affirmé la nécessité d'identifier le mieux possible un système pour pouvoir construire une commande ayant des performances satisfaisantes, fallait-il faire confiance à cette méthode nouvelle, qui prétendait remplacer les commandes, plus ou moins sophistiquées, basées sur l'identification, par des techniques s'appuyant sur le **savoir-faire humain** plutôt que sur des équations ?

Actuellement, cette attitude a évolué. On peut penser que la commande floue va peu à peu prendre place dans la panoplie de l'ingénieur contemporain, sans supplanter les méthodes traditionnelles, et qu'elle constituera un complément précieux **dans le cas de systèmes difficilement identifiables ou dont les paramètres subissent des variations brutales.** Dans cette partie, nous traitons de l'application des principes de la logique floue à la réalisation du contrôleur d'un asservissement de niveau destiné à une application industrielle. Nous nous limitons au cas d'un dispositif dans lequel une seule grandeur physique ***y*** (grandeur de sortie du système) est asservie à l'aide d'un seul signal de commande ***u***.

I.8.1-Historique :

Les prémisses de la logique floue sont apparues avant les années 1940, avec les premières approches, par des chercheurs américains, du concept d'incertitude. Il a fallu attendre 1965, pour que le concept de sous ensemble floue soit proposé par ***L. A. Zadeh*** [3], automaticien de réputation internationale, professeur à l'université de Berkeley en Californie, qui à contribue à la modélisation de phénomène sous forme floue, en vue de pallier les limitations dues aux incertitudes des modèles classiques à équation différentielle. En 1974, ***M.***

Mamdani expérimentait la théorie énoncée par ***Zadeh*** sur une chaudière à vapeur, matériel dont on connait la complexité, introduisant ainsi la commande floue dans la régulation d'un processus industriel. Plusieurs applications ont alors vu le jour en Europe, pour des systèmes parfois très complexes, telle la régulation de fours de cimenterie réalisée par la société « *F.L.Smidt-Fuller* » [4].

Grace au chercheur japonais ***M. Sugeno***, la logique floue était introduite au Japon dès 1985. Les sociétés japonaises comprirent l'avantage à la fois technique et commercial de la logique floue à savoir :

- facilite d'implantation ;
- solution de problèmes multi variables complexes ;
- robustesse vis à vis des incertitudes ;
- possibilité d'intégration du savoir de l'expert [5].

I.8.2- Les sous-ensembles flous

Un sous-ensemble flou *F* est défini sur un ensemble de valeur, le référentiel *U*. Il est caractérisé par une fonction d'appartenance :

$$\mu : x \in U \rightarrow \mu(x) \in [0,1] \qquad \text{(I.22)}$$

Celle-ci qualifie le degré d'appartenance de chaque élément de *U* à *F*.

Exemple : Evaluation de la température de l'eau d'un récipient par les mots qui qualifie le degré d'appartenance de chaque élément de *U* à *F*.

Exemple : Evaluation de la température de l'eau d'un récipient par les mots : Froide : *F*, Tiède : *T*, Chaude : *C*.

En logique classique :

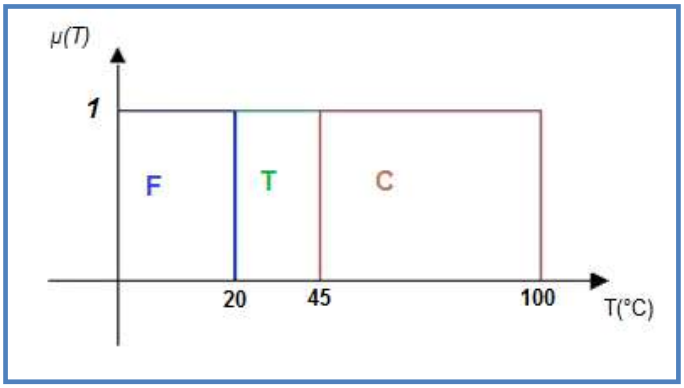


Figure. I.10 : Les fonctions d'appartenances en logique classique.

- En logique floue :

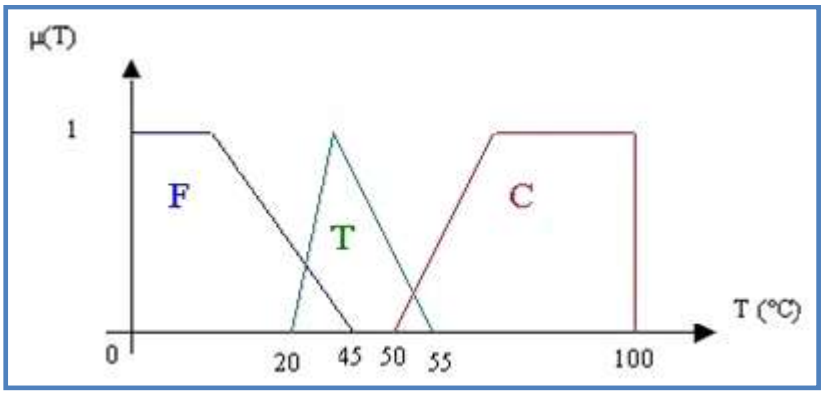


Figure .I.11 : Les fonctions d'appartenances en logique floue.

On voit que la logique classique ne peut utiliser que le ***0*** et le ***1***, ainsi l'eau est d'abord totalement froide puis tiède et enfin chaude. En dessous nous pouvons observer la représentation graphique de trois fonctions d'appartenance Froid, Tiède et Chaud. Ces fonctions nous permettent de superposer sur des plages de température données les qualificatifs froid et tiède ainsi que tiède et chaud. On se rapproche donc du raisonnement humain.

I.8.3-Notions de bases de la commande floue :

La théorie mathématique sur les sous-ensembles flous définit de nombreuses opérations sur ces sous-ensembles et sur les fonctions d'appartenances qui rendent ces notions utilisables. On ne présente ici que les opérations de base de cette théorie.

Si *X*, *Y* et *Z* sont des sous-ensembles flous et *μ(x)*, *μ(y)* et *μ(z)* leur fonction d'appartenance, on définit :

- Le complémentaire de *x*, par la fonction d'appartenance :

$$\mu(z) = 1 - \mu(x) \qquad \text{(I.23)}$$

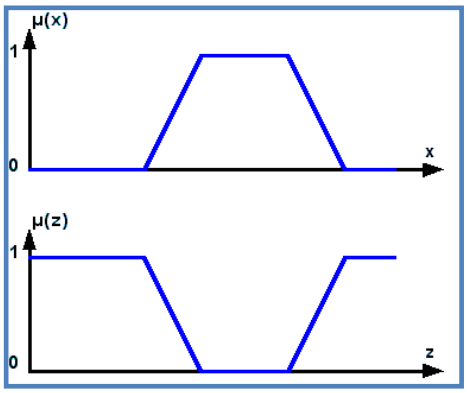


Figure I.12 : Opérateur NON.

- Le sous-ensemble *X* et *Y*, Z=*A*∩ *B*, par la fonction d'appartenance :

$$\mu(z)=min(\mu(x),\mu(y)) \qquad \text{(I.24)}$$

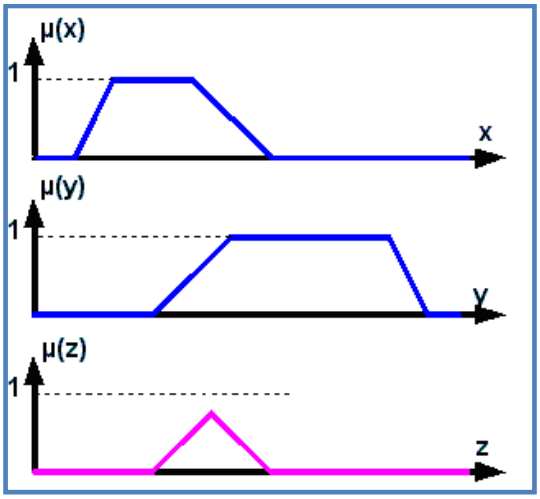


Figure I.13 : Opérateur ET.

- Le sous-ensemble *X* ou *Y*, *Z*=AÈ B, par la fonction d'appartenance :

$$\mu(z)=max(\mu(x),\mu(y)) \qquad \text{(I.25)}$$

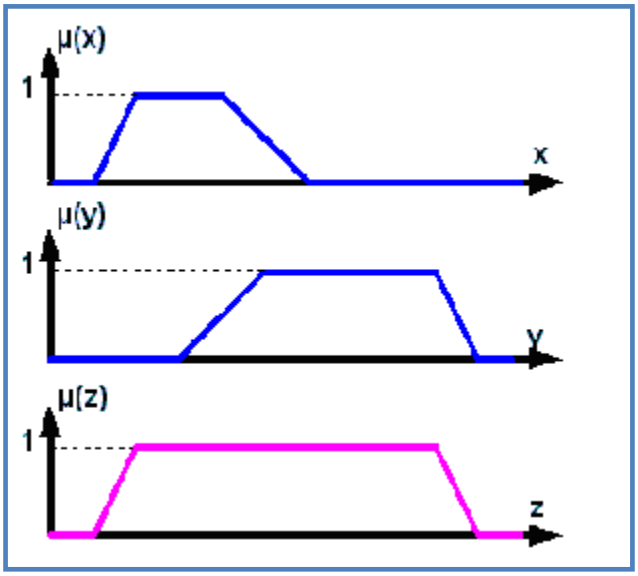


Figure I.14 : Opérateur OU.

Ces définitions sont celles les plus communément utilisées mais parfois, pour certains cas, d'autres sont plus appropriées. Par exemple l'intersection peut être définie par le produit des fonctions d'appartenance et l'union par la moyenne arithmétique des fonctions

d'appartenance. Ces différentes techniques de calcul engendrent une énorme capacité d'adaptation des raisonnements flous.

I.8.4- Structure d'une commande floue :

La structure conventionnelle d'une commande floue est présentée par figure (I. 15). Elle est composée de quatre blocs distincts dont les définitions son données ci-dessous.

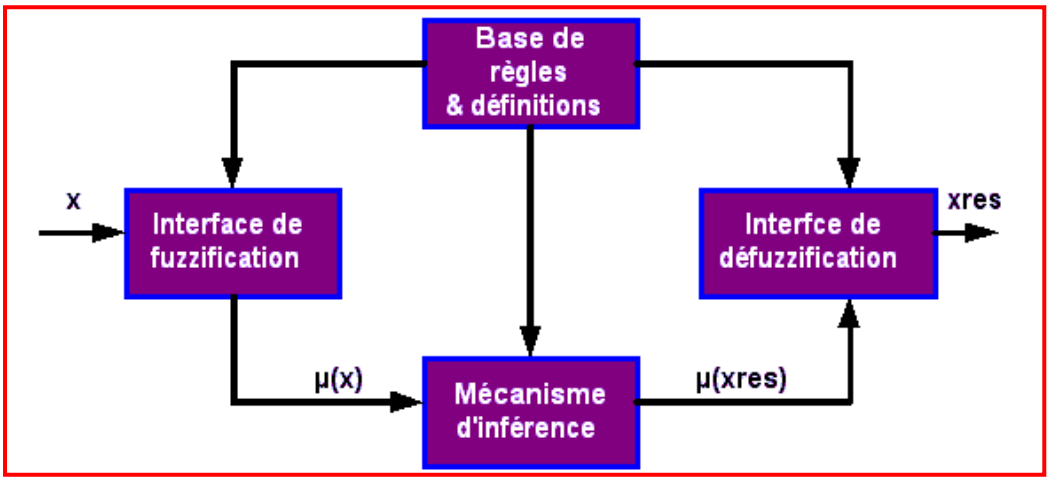


Figure I.15 : Structure d'une commande floue.

Avec **x** représente le vecteur des entrées (variable d'entrée réelles), $\mathbf{x_{RES}}$ celui des commandes (variable de sortie réelle), **μ(x)** et $\boldsymbol{\mu(x_{RES})}$ les fonctions d'appartenances correspondantes (variable d'entrée floue et variable de sortie floue).

On procède tout d'abord à la partition en sous-ensembles flous des différents univers de discours que le système impose. Ensuite on détermine la base de règles qui va caractériser le fonctionnement désiré du système. Puis il faut transformer les variables réelles, c'est à dire celles qui ont une réalité physique, en variables floues. On appelle cette étape la fuzzification On utilise ensuite ces variables floues dans un mécanisme d'inférence qui crée et détermine les variables floues de sortie en utilisant les opérations sur les fonctions d'appartenance. Enfin, on opère à la défuzzification qui consiste à extraire une valeur réelle de sortie à partir de la fonction d'appartenance du sous-ensemble flou de sortie établi par le mécanisme d'inférence.

I.8.5- Quelques techniques de fuzzification et défuzzification :

Il existe de très nombreuses techniques de fuzzification et défuzzification. On ne présente ici que les plus basiques pour rendre un tant soit peu concret ces opérations qui constituent une des parts les plus importantes de la réalisation d'une commande floue.

➢ *Technique de fuzzification* :

L'exemple trivial est la fuzzification d'une valeur exacte x_0. Le sous-ensemble flou lié à cette variable est alors caractérise par la fonction d'appartenance suivante :

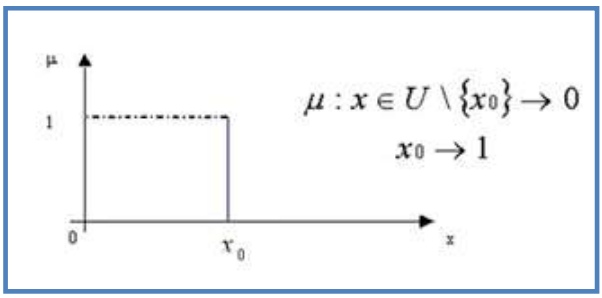


Figure I.16 : Méthode de fuzzification pour une mesure exacte.

L'autre technique de base est la fuzzification d'une valeur x_0 entachée d'une incertitude . La fonction d'appartenance est alors :

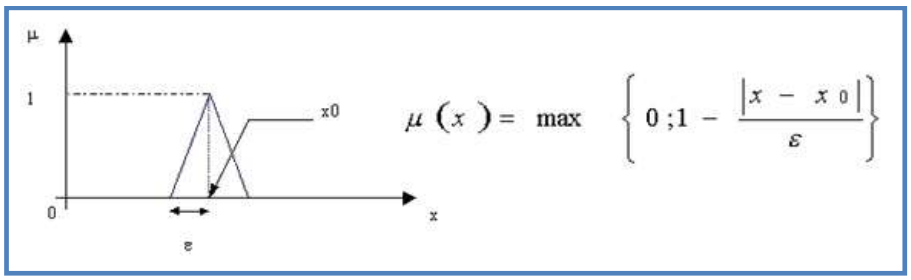


Figure I.17 : Méthode de fuzzification pour une mesure incertaine.

➢ Technique de défuzzification :

Le but de la défuzzification est d'extraire une valeur réelle y_0à partir de la fonction d'appartenance $\mu(y)$ du sous-ensemble de sortie.

➢ Défuzzification par centre de gravite :

La méthode de défuzzification la plus utilisée est celle de la détermination du centre de gravite de la fonction d'appartenance résultante $\mu_{RES}(z)$. Dans ce contexte, il suffit de calculer l'abscisse z^*. La figure I. 18 montre le principe de défuzzification.

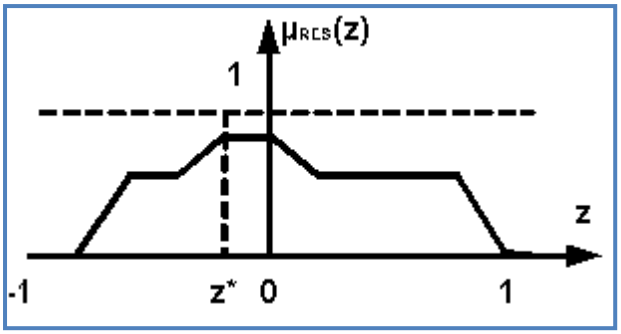
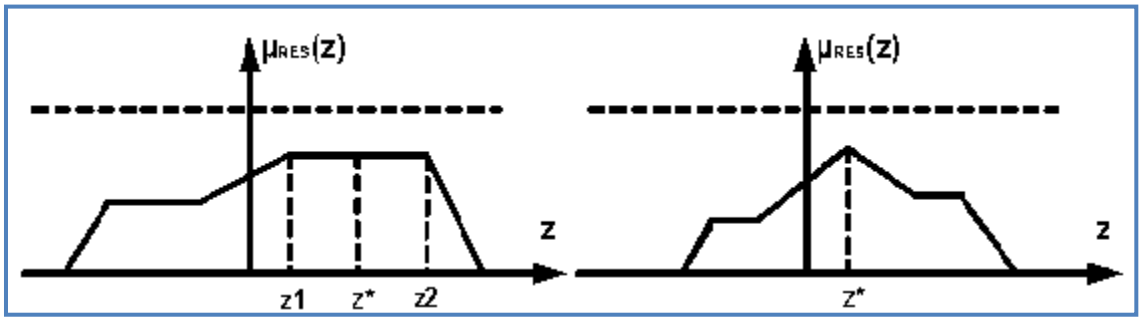


Figure I.18 : Défuzzification par centre de gravité.

- Défuzzification par valeur maximale :

La défuzzification par centre de gravite exige en général une envergure de calcul assez importante. Par conséquent, il sera utile de disposer d'une méthode de défuzzification plus simple. **C**omme signal de sortie *z**, on choisit l'abscisse de la valeur maximale de la fonction d'appartenance résultante μ_{RES} (*z*) comme le montre la figure I.19 (b).



(a) (b)

Figure I.19 : Défuzzification par valeur maximum.

Lorsque $\mu_{RES}(z)$ est écrête, toute valeur entre z_1 et z_2 peut être utilisée comme l'indique la figure I.19 (a). Afin d'éviter cette indétermination, on prend la moyenne des abscisses du maximum. Cependant cette méthode présente un grand inconvénient : le signal de sortie ***z*** saute si la dominance change d'une fonction partielle à une autre.

I.8.6-. Les régulateurs flous de type Mamdani :

Dans les modèles linguistiques, appelés aussi modèle flous de type Mamdani, les antécédents et les conséquences des règles sont des propositions floues. La forme générale des règles de Mamdani est : Un système flou de type Mandani est basé sur une collection de règles du type :

$$R^i: Si \quad x_1 est F_1^i \;\; et \;\; x_n \; est F_n^i \quad ALORS \quad U = h^I \qquad \text{(I. 26)}$$

Ou les x_i représentent les variables d'entrées du régulateur et *U* est la variable de sortie du régulateur, F_j^i étant les sous-ensembles flous.

Généralement les régulateurs flous de type Mamdani, sont des régulateurs à deux entrées, l'erreur et sa variation. Et une sortie, qui représente la variation de la commande, régie par des règles de la forme suivante :

$$R^i: Si\ e\ est\ F_1^i \;\; et\ \Delta e\ est\ F_2^i \quad ALORS\ \Delta U = h^i \qquad \text{(I. 27)}$$

La structure du processus ainsi commandé est donnée par la figure (I.20) qui correspond à un régulateur PI-flou.

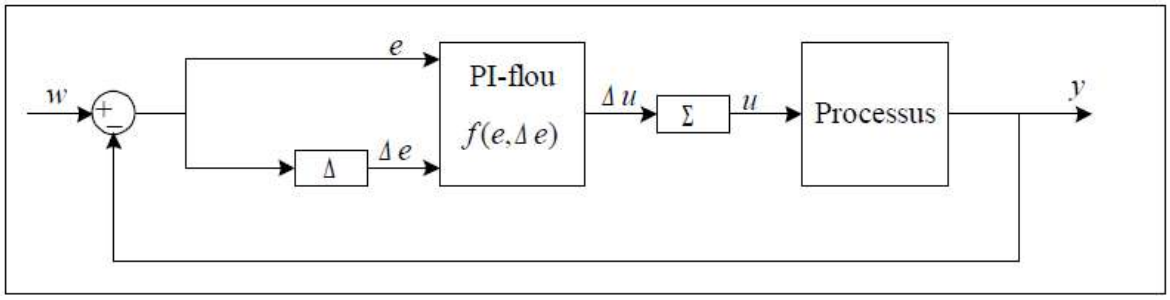


Figure I.20 : Structure d'un PI flou de type Mamdani.

La loi de commande du régulateur PD-flou, est obtenu en suppriment l'intégration à la sortie. Les règles ont la forme suivante :

$$R^i: \textbf{Si}\ \ e \text{ est } F_1^i\ \ \textbf{et}\ \ \Delta e \text{ est } F_2^i\ \ \textbf{ALORS}\ \ U = h^I \qquad (I.28)$$

Pour le régulateur PID-floue, la loi de commande est obtenue grâce à l'erreur, sa variation et sa variation seconde..Les règles ont la forme suivante :

$$R^i:\ Si\ e \text{ est } F_1^i\ \ et\ \ \Delta e \text{ est } F_2^i\ \ \ et\ \ \Delta 2e \text{ est } F_3^i\ \ \ ALORS\ \ \ \Delta U = h^i \qquad (I.29)$$

La défuzzification est généralement effectuée par la méthode du centre de gravite. Une variante de la méthode de Mamdani consiste à remplace l'operateur *min* de l'influence floue par l'operateur prod (produit). [6]

I.8.7- Les régulateurs flous de type Sugeno :

Le modèle de Sugeno est connu sous le nom de TSK, car il à été propose par Takagi, Sugeno et Kang en 1988.ce modèle à pour but de donner à la sortie de la règle des valeurs concrètes, et non pas une valeur floue, qui nécessite une étape de défuzzification.
Le système flou de type Takagi-Sugeno, utilise des règles écrites de la manière suivante :

$$R^i:\ Si\ x_1 \text{ est } F_1^i\ et \ldots\ x_n \text{ est } F_n^i\ \ ALORS\ \ U = h^i\ (X) \qquad (I.30)$$

Chacune de ces règles représente un modèle local sur une région floue d'entrée, ou sur un sous-espace d'entrée. Dans chaque région, le modèle flou est défini par la fonction h^i qui relie les entrées à la sortie numérique. Le modèle global est constitué par l'interpolation des modèles locaux.

I.9 - Conclusion :

Au cours de ce chapitre, nous avons présenté les différentes méthodes de régulation que ça soit en boucle ouverte et en boucle fermée. Nous avons également fourni une description générale des systèmes industriels ainsi que quelque méthode de régulation, qui se résument dans l'utilisation des régulateurs ***PID***. Par la fin, nous avons touché à la logique floue à travers les régulateurs de type Mamdani et Sugeno. Dans le chapitre suivant, nous présenterons les principales caractéristiques et fonctions du logiciel de programmation graphique « ***LabVIEW*** ». Celui-ci sera par la suite utilisé pour l'implémentation software des deux contrôleurs (***PID*** et ***Flou***) utilisés pour la régulation du niveau d'eau dans un réservoir donné.

CHAPTRE II : Présentation du langage de programmation graphique : LabVIEW.

II.1- Introduction à la programmation graphique :

LabVIEW (**Lab**oratory **V**irtual **I**nstrument **E**ngineering **W**orkbench) est un language de programmation essentiellement dédié au contrôle d'instruments et l'analyse de données. Contrairement à la nature séquentielle des langages textuels usuels, LabVIEW est basé sur un environnement de programmation graphique utilisant la notion de flot des données pour ordonnancer les opérations. En faite il existe deux formes de programmation graphiques :
Une programmation dite flot de contrôle et une autre dite flot de données.

II.1.1-programmation flot de contrôle :

Ce type de programmation a été longtemps utilisé pour décrire le déroulement chronologique des algorithmes, ces représentations décrivent les programmes comme étant des nœuds de calcul connectés par des arcs spécifiant à chaque fois quel type de calcul doit être effectue par la suite.

Ex : Grafcet et Réseau de pétri.

II.1.2-Programmation flot de donnée :

C'est une fonction analogue à la propagation du signal à travers un circuit électrique. Le digramme flot de données est un graphe acyclique qui peut être composé des trois 3 éléments suivants :

- ***Des terminaux*** : qui sont les liens avec l'extérieur qui représente la production ou la consommation de données.
- ***Des Nœud*** : qui sont les traitements à effectue et qui sont représente par une figure géométrique pouvant contenir une image illustrant leur fonctionnalité.
- ***Les arcs orienté*** : qui relie les nœuds et les terminaux et permettent d'indique le passage de données d'un noeud vers un autre.[2]

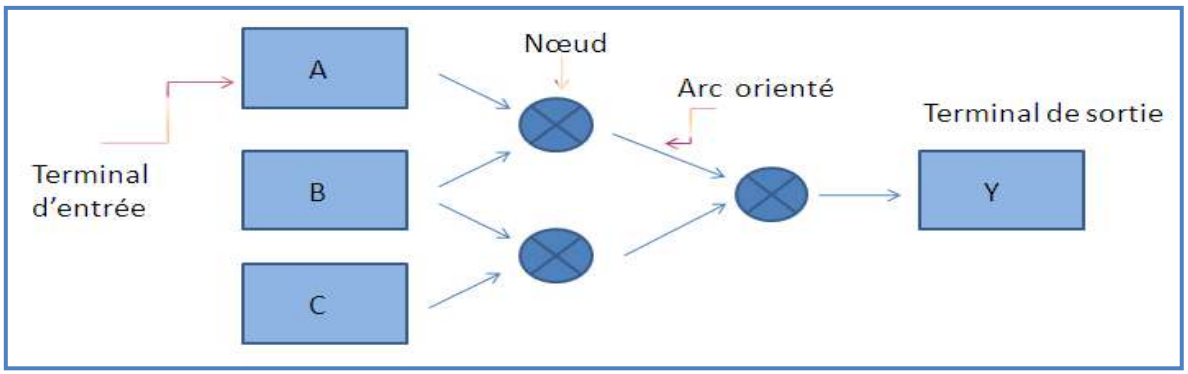


Figure II.1 : Exemple de diagramme de flot de données.

II.2- L'environnement LabVIEW :

LabVIEW est bâti autour du principe d'instrument virtuel (Virtual Instrument ou encore **VI**). Il se décompose en deux parties essentielles :

a) **La première partie (partie cachée ou interne)** : elle contient l'algorithme du programme décrit sous la forme d'un diagramme flot de données en langage graphique.

b) **La seconde partie (partie visible)** : Elle est constituée de l'interface utilisateur.

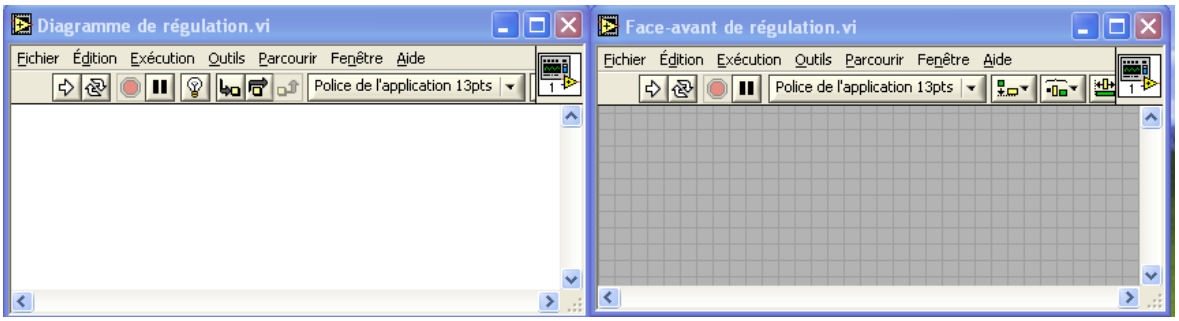


Figure II.2 : Fenêtre de l'environnement de développement sur LabVIEW
Face avant (à droite) et Diagramme (à gauche).

Pour développer un programme sur LabVIEW, on à besoins des «*Palettes*» qui nous offrent la possibilité de modifier la face avant et le digramme de LabVIEW, on trouve trois palettes :

a) Palette d'outils :

Elle est disponible sur la face-avant et sur le diagramme, elle contient les outils nécessaires pour faire fonctionner et modifier la face avant et les objets du diagramme.



Figure II. 3 : Illustration de la palette d'outils.

b) Palette des commandes :

Elle est disponible uniquement sur la face-avant, elle contient les commandes et les indicateurs de la face-avant nécessaires pour créer l'interface utilisateur.

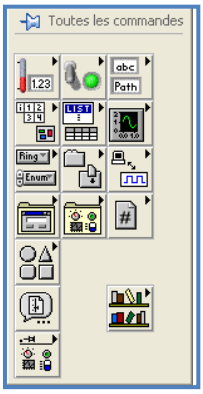


Figure II. 4 : Illustration de la palette des commandes.

c) Palette des fonctions :

Elle est disponible uniquement sur le diagramme. Elle contient les objets nécessaires à la programmation graphique, telles les opérations d'arithmétique, d'E/S d'instruments, d'E/S de fichiers et d'acquisition de données via divers instruments National Instruments.



Figure II.5 : Illustration de la palette des fonctions

II.3- Structure de données manipulées via LabVIEW :

LabVIEW utilise un langage fortement typé et toutes données ou structure de données ne peuvent être manipulées qu'avec des fonctions admettant ce type. En faite dans LabVIEW, on trouve les types de base scalaire, les types entiers (signés ou non, codés sur 8, 16 ou 32 bits), le type réel (codé sur 16, 32 ou 64 bits), le type booléen et le type chaine de caractères (figure II.6**)**. Il est important de noter que les éléments représentant ces données, ainsi que les liaisons issues de ces éléments, sont de forme et de couleur différente.

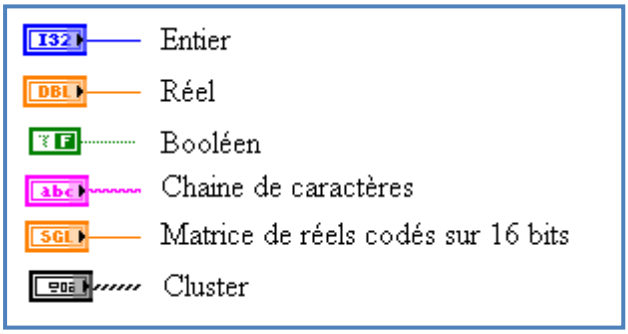


Figure II. 6 : Les différents types de structures de données dans LabVIEW.

Le langage permet également de créer des structures de données plus élaborées.

II.3.1- le type tableau (structure de données homogènes) :

Comme tous les langages de programmations, la présence de notion de tableau est obligatoire, puisqu'elle définit un outil de base pour le stockage et les traitements des informations**.** La bibliothèque : « ***tableau*** » dans LabVIEW est riche en fonctions prédéfinies, telle que, l'initialisation, l'indexation, l'inversion d'un tableau, la concaténation de deux tableaux, et plusieurs autres fonctions, bien sur, la notion des tableaux peut être élargie pour contenir les matrices. Et comme toutes les autres bibliothèques, on peut l'enrichir avec des fonctions ou des procédures créent à partir des langages de bas niveau comme le langage C.

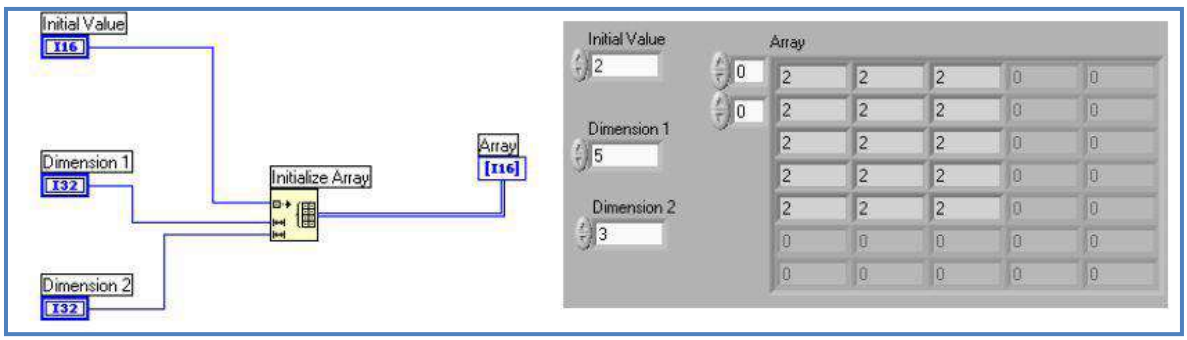


Figure II. 7 : Exemple d'initialisation d'un tableau à 2 dimensions (matrice 5x3).

II.3.2- Le type «*cluster*» (structure de données hétérogènes) :

Ce second constructeur de type est l'équivalent du « record» en langage ***Ada*** ou du « struct » en langage ***C***. Les différents composants ou champs de ce groupe de données (***cluster***) peuvent être assemblés ou récupérés par des fonctions spécifiques.

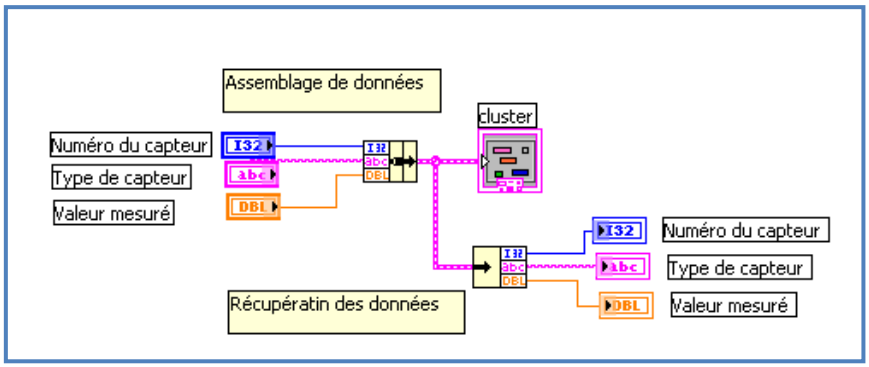


Figure II. 8 : Exemple de manipulation des données avec le type cluster.

II.4 - Structures de programmation :

LabVIEW utilise un langage flot de données pur, celui-ci a été enrichi de quatre types de structures : la séquence, deux structures d'itération (la boucle « Pour » avec un nombre d'itérations fixe et la boucle « Tant Que » avec un nombre d'itérations soumis à condition) et la structure de choix. Et vue leur importance nous ne détaillons ici que les structures itérative.[2]

II.4.1- Les structures itératives :

Les deux structures itératives, la boucle « Pour » et la boucle « Tant que », ont aussi le statut d'un nœud ordinaire. La boucle « Pour » permet d'exprimer la répétition (ou l'itération) pour un nombre de fois prédéterminé défini par une connexion d'entrée obligatoire: le nombre d'itérations à effectuer *N*. A l'intérieur de la boucle « Pour » se trouve un terminal d'entrée local générant l'entier indiquant l'indice d'itération de la boucle (*i* varie de *0* à *N–1*).[2]

- **Exemple a :**

L'utilisation de cette structure, présentée sur la figure (II.9), concerne l'acquisition de *N* mesures avec un temps fixe entre chacune, défini par un nœud de temporisation Les *N* données de sortie sont assemblées pour former un vecteur avec une auto-indexation.

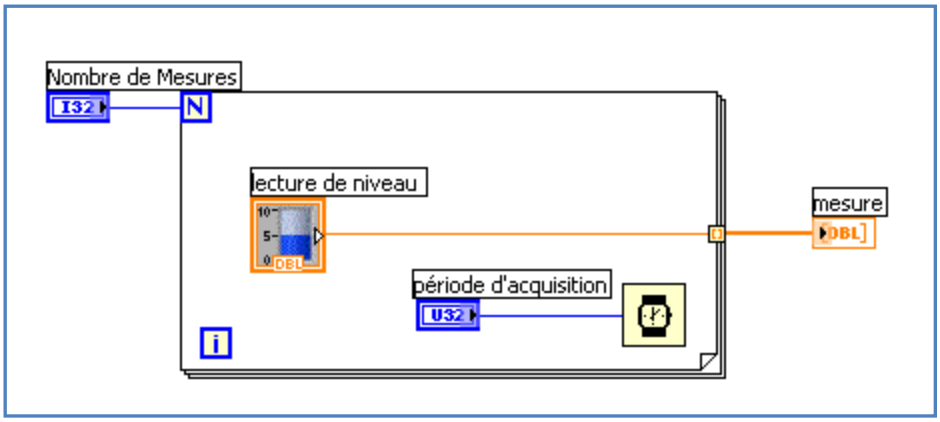


Figure II.9 : Exemple d'utilisation de la structure itérative "pour".

La boucle « Tant Que » permet d'exprimer la répétition pour un nombre de fois non connu à l'avance. à l'intérieur de la boucle «Tant Que» se trouve un terminal d'entrée local générant l'entier indiquant l'indice d'itération de la boucle. Un terminal de sortie de type booléen permet d'arrêter la boucle lorsque la valeur « False » lui est envoyée.

- **Exemple b:**

L'utilisation précédente décrite sur la figure (II. 9), est reprise avec cette structure « Tant Que » en arrêtant l'acquisition pour une valeur trop grande de la mesure (figure II.10).

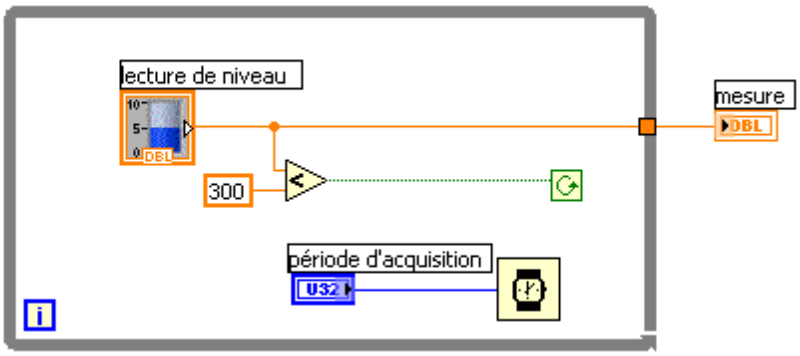


Figure II.10 : Exemple d'utilisation de la structure itérative "tant que".

Dans les deux cas de structure itérative lorsque des tableaux de données arrivent ou partent des tunnels, ceux-ci ont la possibilité d'être automatiquement indexes : récupération un par un des éléments du tableau ou concaténation pour créer un tableau. Dans le cas des deux structures, il est possible de mémoriser des résultats produits lors d'itérations antérieures et de les récupérer ensuite. Ces registres à décalage sont des variables locales à une boucle.

- **Exemple c:**

La figure (II.11) présente une utilisation de ces registres à décalage dans le cas de la boucle « Pour ». Il est important de remarquer que ces registres à décalage constituent un des effets

de bord du langage. Il est de plus nécessaire d'initialiser ces mémoires locales pour réaliser une utilisation contrôlée et cohérente. [2].

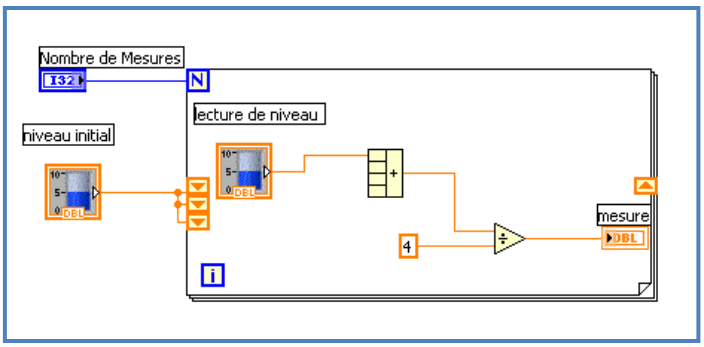


Figure II.11 : Exemple d'utilisation des registres à décalage dans une boucle "pour".

II.5-Traitement numérique :

II.5.1 : les fonctions prédéfinies :

LabVIEW possède également les instructions de base d'un langage de programmation permettant de traiter les différents types de données. Ainsi, nous avons des fonctions liées aux variables numériques (entiers, réels et complexes), aux variables booléennes, aux chaines de caractères, et aux tableaux.

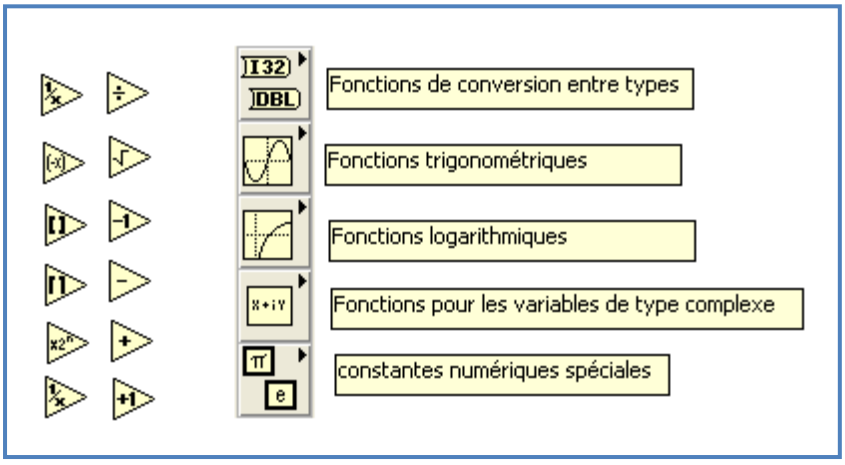


Figure II.12 : Quelques instructions de traitement de données numérique dans LabVIEW.

Nous trouvons aussi les operateurs de test et de comparaisons liées à ces différents types de données. A partir des structures de contrôle, des fonctions et des operateurs, de base, il est alors possible de traduire un algorithme quelconque et d'enrichir la bibliothèque des fonctions en utilisant le mécanisme d'encapsulation. Un diagramme complet est alors réduit à un nœud qui peut être ensuite réutilisé.

II.5.2 : Boites à outils mathématique :

LabVIEW contient aussi des boites de calcul mathématiques qui servent à introduire des commandes complexes tel que (linspace , ode45 ,sin, cos , etc..).

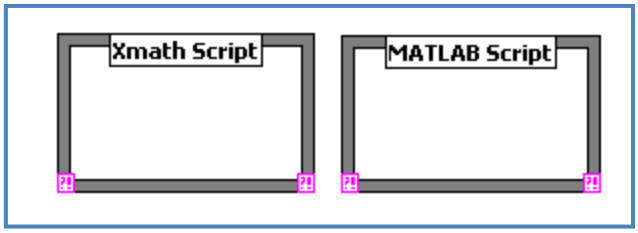


Figure II.13 : Exemple de boite de calcul mathématique dans LabVIEW

II.6- Bibliothèques de commande :

LabVIEW possède aussi des bibliothèques de fonctions spécialisées dans le domaine de la mesure, du test et du contrôle-commande. Ces bibliothèques de fonctions peuvent être réalisées soit à partir des fonctions et operateurs de base vus précédemment, soit directement par des programmes écrits en langage de haut niveau, compilé et ensuite intègré sous la forme d'un nœud graphique.

Cette dernière possibilité est aussi offerte aux utilisateurs de cet environnement LabVIEW par l'intermédiaire de la construction de nœuds particuliers appelés « nœud d'interface vers code » (CIN : Code Interface Node). L'utilisation de ces fonctions qui peuvent être assez complexes est facilitée par une aide en ligne qui donne pour chaque fonction une description sommaire, souvent suffisante de ses spécifications.



Figure II.14 : Exemple de fenêtre d'aide offrant la description d'une fonction fournie dans la bibliothèque de LabVIEW

Dans notre projet on à besoin des bibliothèques spécifiques qui n'existe pas dans les versions standard de LabVIEW, mais dans des versions nommée (LabVIEW RT) c.-à-d. des versions Real Time, comme les bibliothèques de commande et de contrôle. En faite nous

avons utilisé des bibliothèques nommées «***PID control toolset***» et «***fuzzy control toolset***» pour la commande et la régulation de notre système. Il existe d'autres bibliothèques de commande telle que «DSP control toolset» et quelques autres commandes avancées.

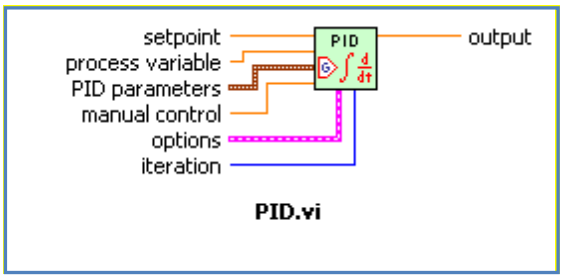
Dans ce qui suit, on va présenter les deux bibliothèques de commande que nous avons utilisé dans notre projet, à savoir :

II.6.1-La bibliothèque de contrôle PID :

Cette bibliothèque, ajoute aux fonctions standards de LabVIEW un ensemble d'algorithmes de régulation ***P***, ***PI***, ***PD*** et ***PID***, avec toutes les options que l'on trouve généralement dans les systèmes de régulation complets.

Tous les régulateurs dans cette bibliothèque ont une structure parallèle de la Forme :

$$C(p) = \frac{U(p)}{\varepsilon(p)} = K_P + \frac{1}{pT_d} + pT_d \qquad (II.1)$$



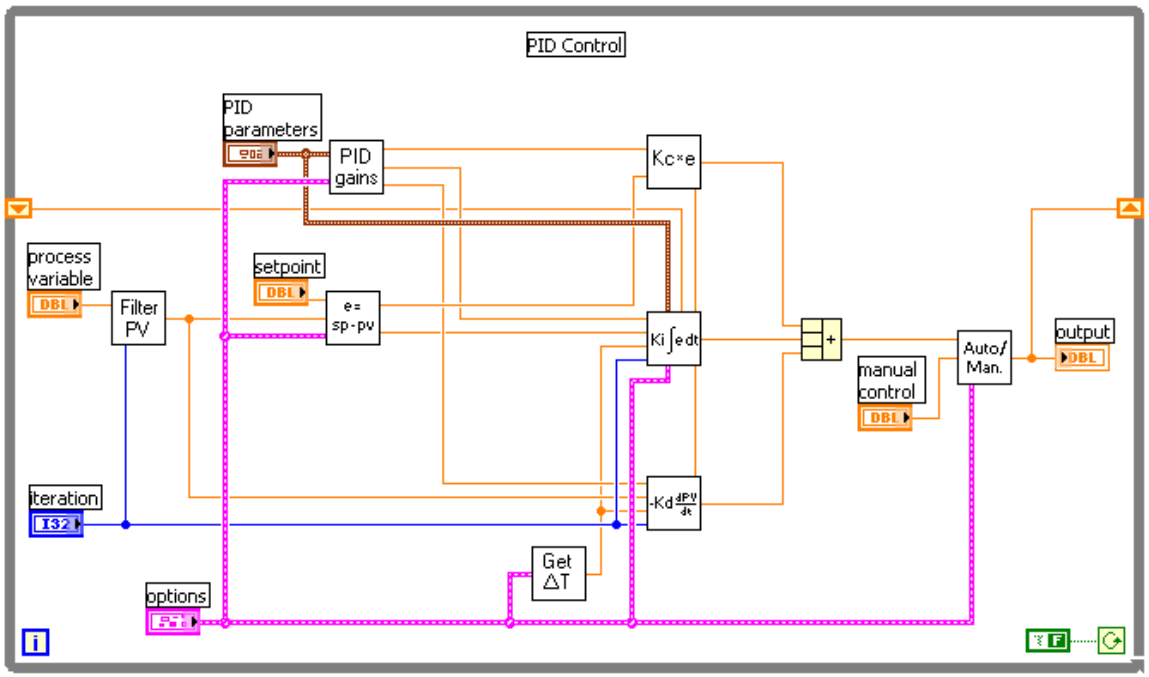


Figure II.15 : Exemple de bibliothèque de régulation PID [2].

II. 6. 2 - La bibliothèque de logique floue :

Elle est destinée à accélérer le développement d'applications de contrôle pour les systèmes non linéaires ou complexes. Elle propose une interface graphique prête-à-l'emploi pour la conception de controleurs de logique floue et des VI (Instruments Virtuels) pour créer ces contrôleurs dans LabVIEW.

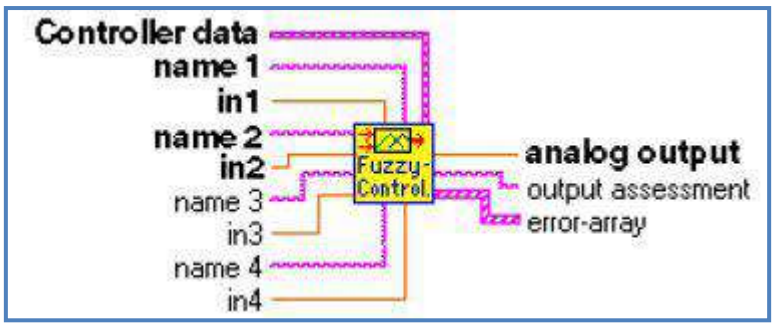


Figure II.16 : Exemple de Bibliothèque de la commande floue [2].

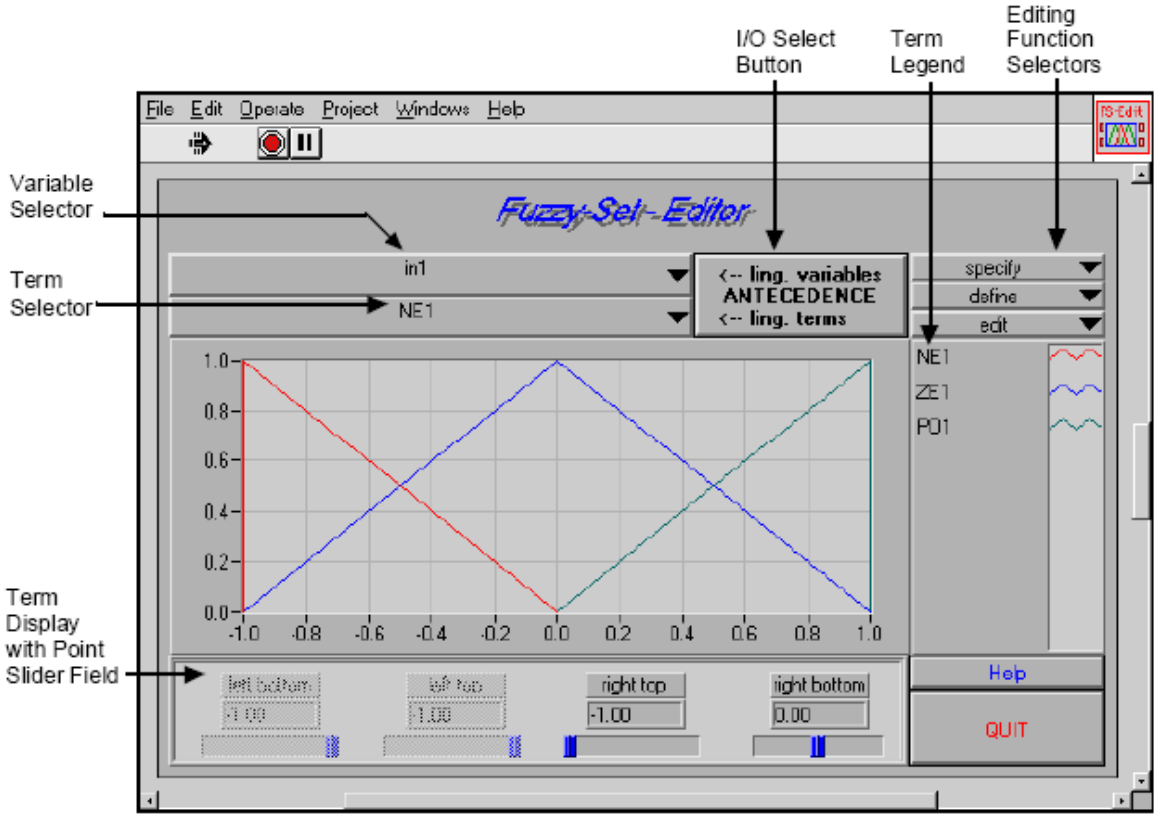


Figure II.17 : Interface graphique pour la conception d'une commande floue .

II. 7 Conclusion :

Dans ce chapitre, nous avons présenté les principales caractéristiques et outils du logiciel de programmation graphique LabVIEW. On a pu montrer que le logiciel en question possède les instructions de base d'un langage de programmation permettant de traiter les différents types de données. Il est également muni des bibliothèques de fonctions spécialisées dans le domaine de la mesure, du test et du contrôle-commande.

III : Description du système de régulation de niveau

III. 1. Introduction :

Au cours de ce chapitre, nous allons donner une description plus au moins détaillée du système pris en considération. Le processus étant un réservoir cylindrique dont on veut réguler le niveau de remplissage par un liquide donnée. Cette régulation est assurée par un ensemble d'outils (Hardware et Software), qui combinés ensemble nous ont permis d'automatiser la surveillance et la fixation du paramètre (niveau) à une consigne choisie par l'utilisateur.

Pour l'envoie des commandes et la réception des mesures depuis et vers le processus sur site. Nous utiliserons aussi une carte d'acquisition NI6008. Celle-ci est une carte d'interface universelle qui est capable de réaliser toutes les tâches d'acquisition de données requises pour la concrétisation de ce projet.

III. 2. Présentation générale de la maquette :

Comme illustré sur la figure du schéma bloc dressée ci-dessous, le système à mettre en œuvre comprend essentiellement :

- *Le processus* : Réservoir d'eau avec un niveau de remplissage à réguler,
- *Des capteurs des paramètres thermohydrauliques* : Dans notre cas, il s'agit bien d'un capteur de débit du remplissage et d'un capteur de niveau,
- *Un contrôleur industriel* : (soit un PID ou un contrôleur flou),
- *Un comparateur*,
- *Un pupitre (ordinateur dans notre cas)* : pour faire entrer la consigne.

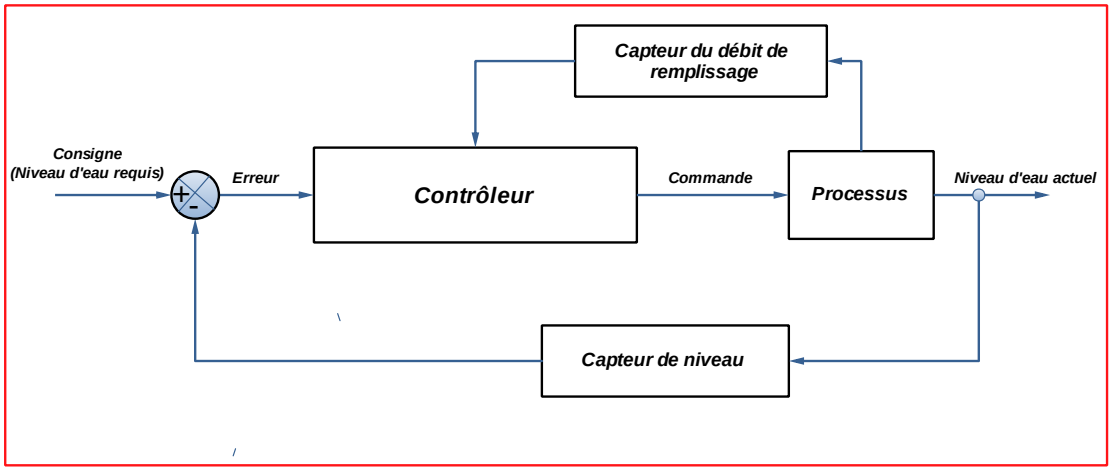


Figure III.1. Schéma bloc du système global.

Il faut noter que d'autres paramètres n'apparaissent pas volontairement sur ce schéma bloc. Ceci est dû au fait que leur présence ne fait qu'encombrer beaucoup plus ce schéma bloc. Il s'agit surtout de : Pompe de remplissage, les vannes de purge, actionneurs, systèmes électronique d'acquisition et d'envoie des commandes, …etc.

Un éclaircissement de la maquette d'étude et de travail est fourni également par l'interface graphique réalisée sous le logiciel LABView qui apparait sur la figure ().

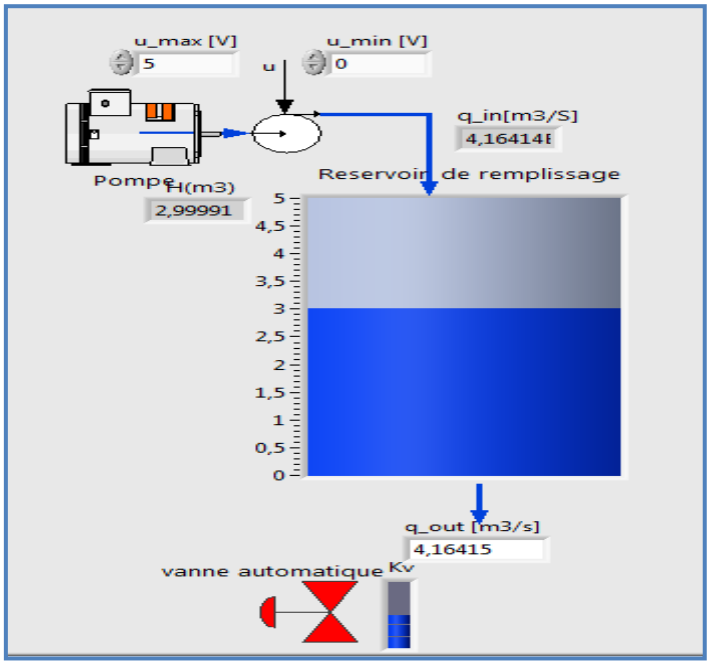


Figure III. 2. Structure de la maquette.

Comme on peut le remarquer sur cette figure, le système est composé de :

- ✓ Un réservoir est de forme cylindrique. Ce réservoir présente une entrée en haut caractérisée par un débit de remplissage (***q_inp***), et une sortie d'évacuation en bas caractérisée par un débit de vidange (***q_out***).

- ✓ Une vanne de vidange (automatique ou manuelle)
- ✓ Une pompe permet le remplissage du réservoir, elle est pilotée par un module électronique (de puissance), qui reçoit à son tour un signal de commande, issu du contrôleur, et qui varie dans une gamme s'étalant de ***0*** à ***5V***.

III.4. Modélisation mathématique de système :

N'import quel simulateur est basé sur une modèle mathématique du système simuler. Ainsi nous commençons par développer un model mathématique du système à réguler :

Selon les lois de la mécanique des fluides, on sait que la masse d'un liquide de densité ρ stocké dans un réservoir dont la surface de base est A est donnée par

$$m(t)= \rho A\, H(t) \qquad \text{(III.1).}$$

Tout en sachant que H est le niveau occupé par le liquide dans le réservoir.

Aussi, on sait que le débit volumique du liquide de remplissage traversant la pompe est proportionnel au signal de commande de cette pompe.

$$\boldsymbol{q_in}=K_u.U(t) \qquad \text{(III.2).}$$

De même, on sait que le débit de vidange ***q_out*** est proportionnel à la racine carré de la chute de pression au dessus de la vanne.

$$\boldsymbol{q_out} = k_v\sqrt{\rho g H(t)} \qquad \text{(III.3)}$$

Fixer la hauteur du liquide à l'intérieur du réservoir (le niveau H), revient à fixer la masse du liquide à l'intérieur de celui-ci (puisque la section de la base est fixe).
Ceci implique une variation nulle de cette masse par rapport au temps. Celle-ci est donnée par :

$$\frac{d\, m(t)}{d\, t} = [\rho\, q_{in}(t) - \rho q_{out}(t)] \qquad \text{(III .4)}$$

En remplaçant ***q_in(t)*** et ***q_out(t)*** par leurs valeurs retrouvées dans (III. 2) et (III. 3) on trouvera finalement :

$$\frac{dm(t)}{dt} = \rho\, Ku.U(t) - \rho\, k_v\sqrt{\rho g H(t)} \qquad \text{(III.5)}$$

En remplaçant m(t) par sa valeur dans (III. 1) on trouvera finalement :

$$\frac{dh(t)}{dt} = \frac{1}{A}\ \left(\mathrm{Ku.\,U(t)} - k_v\sqrt{\rho g H(t)}\right) \qquad \text{(III.6)}$$

Tel que :

- *m(t)* [kg] : La masse du fluide considéré.
- ρ [kg/m3]: La masee volumique du fluide .
- g [N.Kg/m.] : la gravité terrestre.

- A [m^2] : La surface de la base du réservoir.
- H [m] : hauteur du fluide dans le reservoir.
- *Kv :* Le coefficient de propotionnalité lié à ***q_out***.

C'est l'équation qui nous a permis de simuler notre système (reservoir + pompe + capteur de débit) à l'aide du logiciel de programmation graphique LABView. Une schématisation de cette équation (qui est en fait une équation d'état) est donnée par la figure (3). On voit bien que : Partant du signal de commande ***U*** (issu du contrôleur), on peut facilement agir sur la variable H représentant le niveau du liquide à réguler.

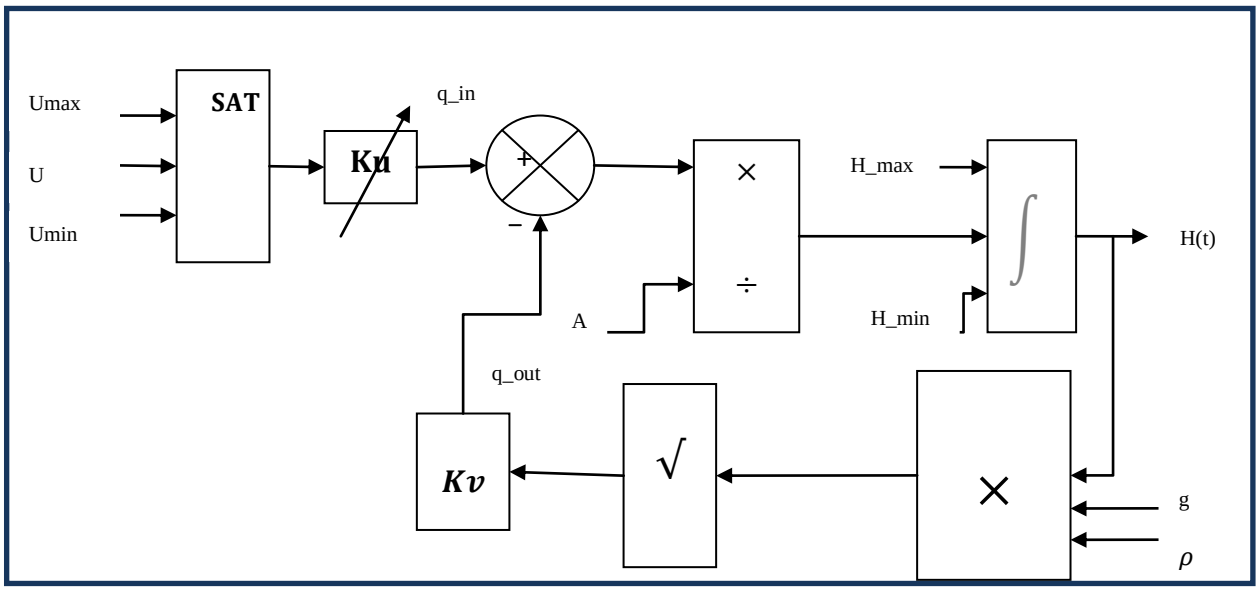


Figure III. 3. : Schéma bloc fonctionnel de processus.

III. 5. Acquisition du paramètre : Débit de remplissage du réservoir :

Dans le but d'acquérir les mesures liées au processus à commander et à appliquer les commandes appropriées, nous avons utilisé une carte d'acquisition de données dont la référence est : NI6008.

Suite à une contrainte de non-disponibilité des capteurs des mesures dans l'immédiat. Nous n'avons utilisé cette carte que pour l'acquisition du paramètre débit. Le capteur de ce paramètre à été remplacé par un potentiomètre délivrant une tension de mesure variant de 0 à 5V. La valeur de cette tension de mesure est injectée au processus de commande via la carte d'acquisition NI6008 qui communique avec la PC à travers une liaison USB ordinaire.

III. 5. 1. principe de fonctionnement de la carte NI 6008 :

Cette interface d'acquisition et d'envoie de données, dotée par de multiples fonctions, permet de lier un processus physique externe (analogique ou logique) à un PC sur lequel serait installée une loi de commande donnée. Son installation et sa mise en marche est très simple (un simple branchement sur le port USB de l'ordinateur et installation des drivers correspondant). Son emploi est surtout adapté à des travaux de laboratoire; aux usages industriels limités en fréquence d'acquisition et du nombre de paramètres à acquérir.

III. 5. 1. Description de la Carte NI 6008 :

Cette carte d'acquisition, fabriquée par la firme « National Instrument » est utilisée pour convertir les signaux de mesures analogiques, issus d'un monde réel, en signaux numériques afin de les traiter via l'ordinateur.

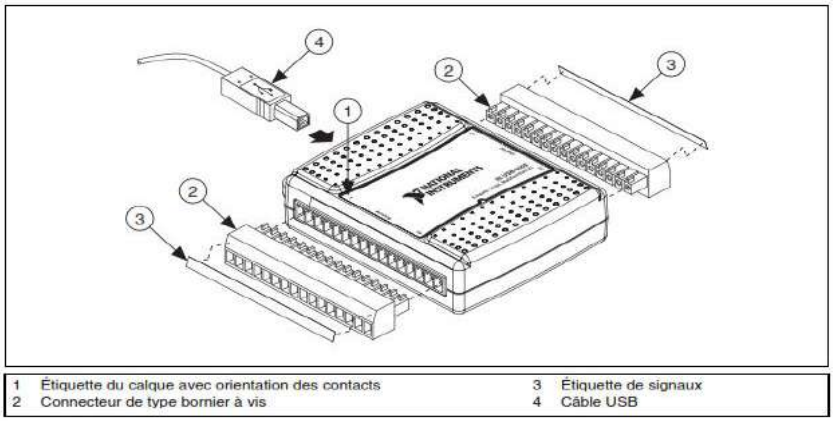


Figure III. 4 : Vue perspective de la carte NI 6008.

Nous avons choisi ce modèle de carte d'acquisition car nous l'avions à disposition. En plus, la carte en question et le logiciel de développement des commandes (LABView) sont produit par la même firme i. e. « National Instruments », donc les anomalies et les aléas de fonctionnement dus à la non compatibilité du pont de vu matériel-programme sont à écarter dans un tel cas de figure. La carte en question dispose de quatre entrées analogiques différentielles (ou 08 entrées analogiques référenciées à la masse : single-ended). La conversion analogique numérique s'effectue sur 10 bits à une fréquence d'échantillonnage de 10 KHz **[10].** (Pour de plus amples informations sur ce module, le lecteur est invité à voir l'annexe)

III.6. Conclusion :

Lors de ce chapitre, nous avons décrit la quasi-totalité des différents constituants du système régulé à mettre au point (processus, contrôleurs, système d'acquisition des paramètres externes, …etc.). On tient seulement à rappeler ici que la non-disponibilité

momentanée des capteurs, nous a poussé à utiliser des subterfuge pour surmener cet handicap. En ce qui concerne le capteur de niveau, celui-ci a été remplacé par un capteur simulé via le logiciel LABView. Tandis que le deuxième capteur, capteur du débit, il a été remplacé par un potentiomètre externe délivrant une tension réglable manuellement qu'on fournie au contrôleur via la carte d'acquisition NI6008.

IV-Résultats et discutions

IV.1-Introduction :

Après avoir décrit le processus à commander et les outils nécessaires pour son implémentation pratique lors du chapitre précédent. Nous examinerons lors de ce chapitre l'aspect de l'association des contrôleurs (Flou et PID) à ce système. Ces contrôleurs là, vont être introduit dans le système considéré en une configuration en boucle fermée et ce dans le but de fixer le niveau du liquide dans le réservoir selon une consigne bien déterminée quelques soit les conditions et la vitesse de remplissage et de vidange de ce dernier. Nous reverons également les différentes étapes qui ont caractérisée la conception et le choix des paramètres de ces deux contrôleurs et on examinera aussi leurs performances dynamiques que ça soit en asservissement ou en régulation. Les résultats de comparaisons entre les réponses des deux contrôleurs seront évoqués en fin du chapitre.

IV.2- boucle de régulation :

Le schéma de la figure (III. 3) dressée au chapitre précédent à été complété pour un inclure un contrôleur (PID ou Flou) et un comparateur associé à la consigne du niveau choisie pour donnée naissance au système de contrôle illustré à la figure (IV. 1.)**.** Un examen attentif de celle-ci révèle les différentes entités (softwares) utilisée pour simuler le comportement global du système considéré. A noter ici que seul le débit du remplissage est acquis de l'extérieur à travers la carte d'acquisition NI 6008. Les autres paramètres (niveau du liquide, action de commande, …etc.) sont obtenus par simulation.

Sur la figure (IV. 2) apparait la page qui correspond à ce modèle sur l'interface graphique réalisée via le logiciel LABView. Sur cette page, apparait le curseur de fixation de la consigne liée au niveau choisi (à gauche). Juste à côté, nous pouvons remarquer le bouton du choix du type de contrôleur (PID ou Flou). Au milieu de la page, on peut voir l'image du processus à commander (Réservoir + Pompe + Vanne), à noter ici que le niveau du liquide (en couleur bleue) varie schématiques selon le modèle physique adopté. Tout à fait à droite, on a une illustration d'un oscilloscope graphique sue lequel évoluent les différents signaux de mesures et de commande. Et juste en bas figurent l'ensemble des règles floues utilisées pour la configuration du contrôleur Flou.

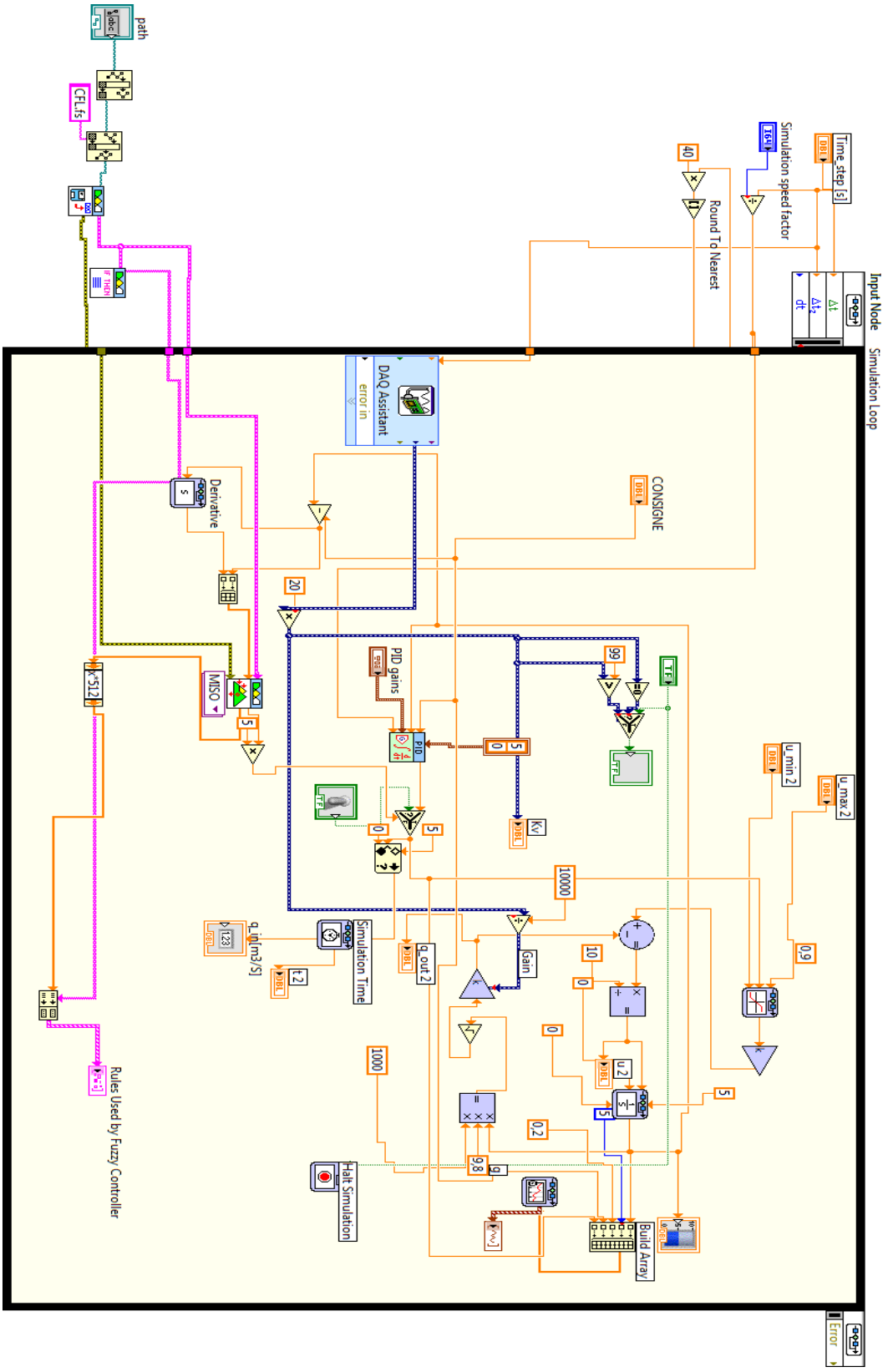


Figure IV.1 : Diagramme de commande tel qu'il a été simulé par LABView.

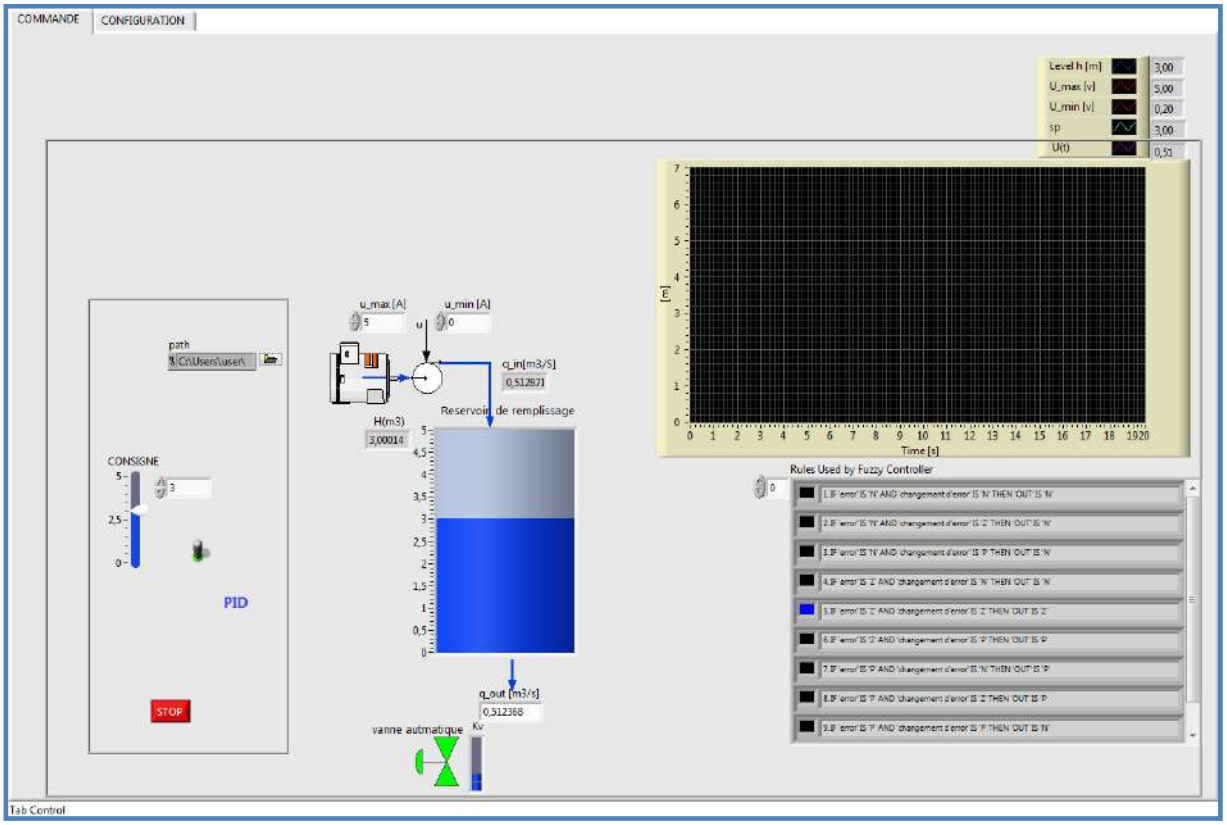


Figure IV.2 : page de commande sur l'interface graphique.

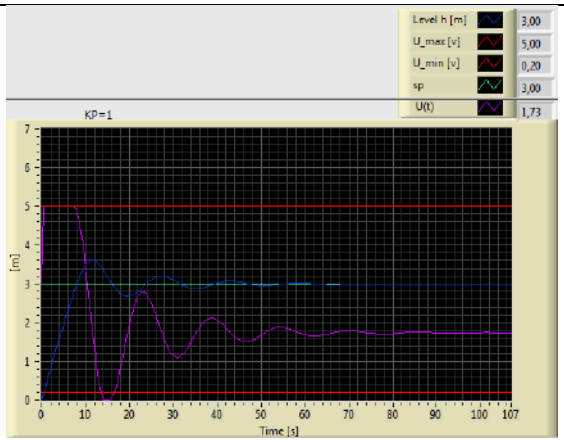
IV. 3. Phases de conception des deux contrôleurs :

Le contrôleur principal à réaliser dans le cadre de ce travail est bel et bien le contrôleur flou. Mais pour des besoins de comparaison de performances, nous étions amenés à concevoir un contrôleur PID classique contrôlant le même processus physique.

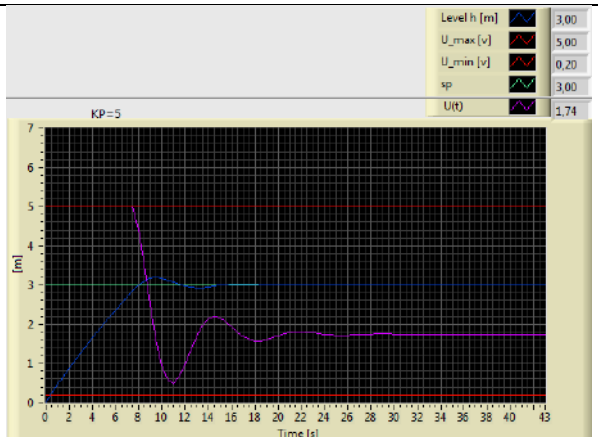
IV. 3. 1. Configuration des paramètres du contrôleur PID :

La version du logiciel utilisé pour la réalisation de ce travail (*LABView 2011*) jouit d'une bibliothèque d'outils contrôle-commande très diversifiée. Parmi les outils disponibles on note la présente des deux structures de contrôleurs PID (parallèle et série) configurables et prêtes à l'emploi. Pour ce qui de notre application, nous opté pour un contrôleur PID à structure parallèle, les paramètres de ce dernier (K_P, K_I, K_D), on été choisis par tâtonnement à travers plusieurs essais pour obtenir les meilleurs valeurs adaptés au processus en cours. Le tableau suivant, résume les différentes étapes suivies.

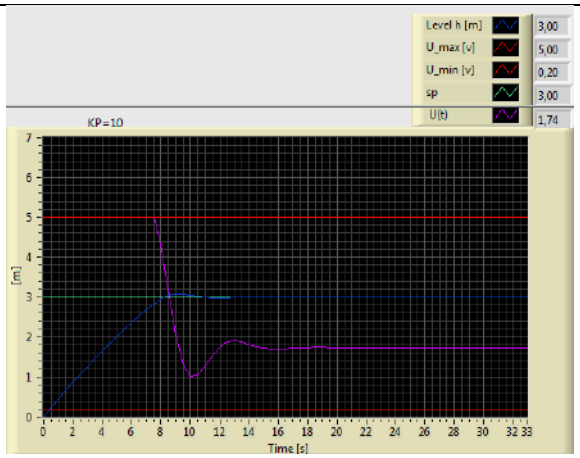
- **Tableau 1 :** Opération du choix des paramètres du contrôleur PID.


Level h [m] 3,00
U_max [v] 5,00
U_min [v] 0,20
sp 3,00
U(t) 1,73
KP=1
[m]
Time [s]

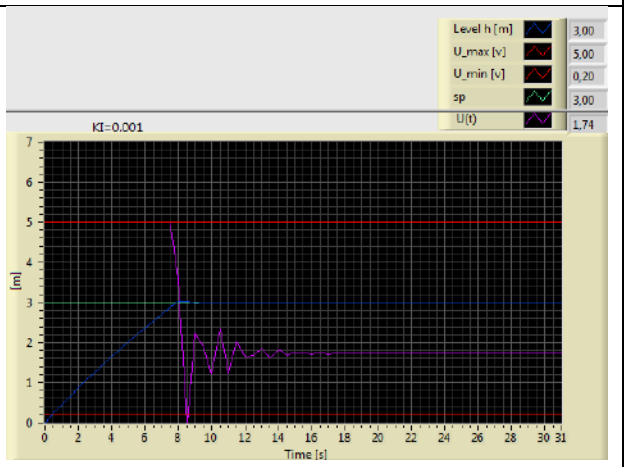
(A)


Level h [m] 3,00
U_max [v] 5,00
U_min [v] 0,20
sp 3,00
U(t) 1,74
KP=5
[m]
Time [s]

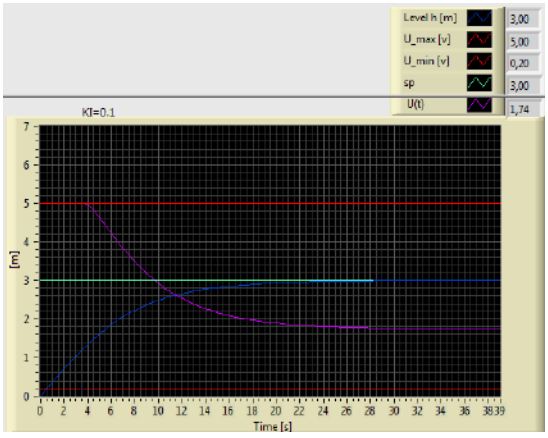
(B)


Level h [m] 3,00
U_max [v] 5,00
U_min [v] 0,20
sp 3,00
U(t) 1,74
KP=10
[m]
Time [s]

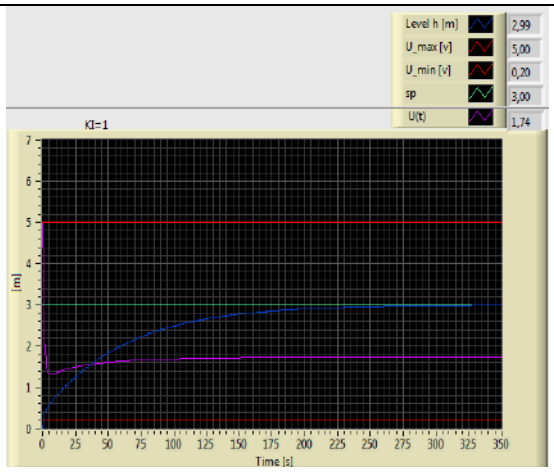
(C)


Level h [m] 3,00
U_max [v] 5,00
U_min [v] 0,20
sp 3,00
U(t) 1,74
KI=0.001
[m]
Time [s]

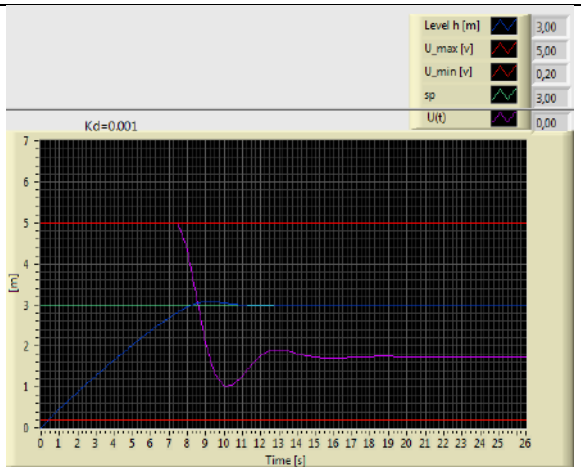
(D)


Level h [m] 3,00
U_max [v] 5,00
U_min [v] 0,20
sp 3,00
U(t) 1,74
KI=0.1
[m]
Time [s]

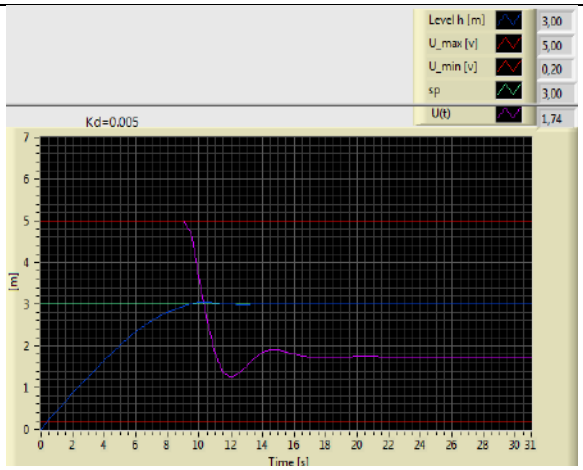
(E)


Level h [m] 2,99
U_max [v] 5,00
U_min [v] 0,20
sp 3,00
U(t) 1,74
KI=1
[m]
Time [s]

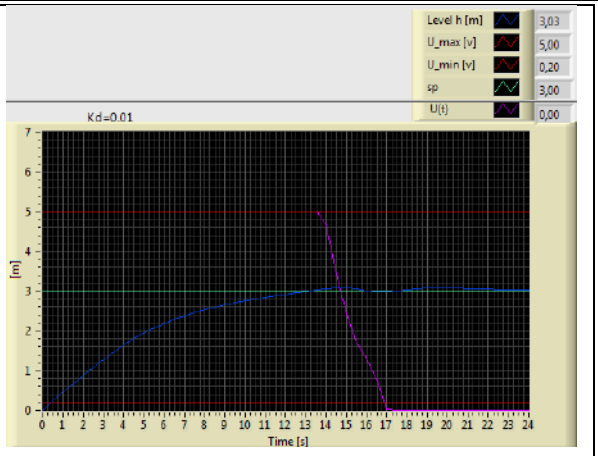
(F)



(G)



(H)



(I)

En examinant de près ce tableau, on remarque bien que :

- Lors des cas : (A, B, C) : on a varié l'action proportionnelle $\boldsymbol{K_p}$ et fixé les deux autres actions : l'action et dérivé $\boldsymbol{K}_{\mathrm{I}}$ et $\boldsymbol{K_d}$.
- Lors des cas (D, E, F) :on a varié l'action intégrale $\boldsymbol{K_I}$ et laissé les actions $\boldsymbol{K_p}$,$\boldsymbol{K_d}$ constantes.
- Lors des cas : (G, H, I) : on a varié l'action dérivé $\boldsymbol{K_d}$ et laissé les actions $\boldsymbol{K}_{\mathrm{p}}$, $\boldsymbol{K_I}$ constantes.

Donc un bon réglage des différents actions résulte d'un compromise entre la précision, rapidité et la stabilité (oscillations).

Ces tests liés aux choix de paramètres du contrôleur, ont renforcé chez nous les conclusions suivantes :

-Fonction proportionnelle donne un système plus précis, plus rapide.

-Fonction intégrateur élimine l'erreur statique (stabilité).

-Fonction dérivée accélère la correction.

Apres tous les essais on a choisi les paramètres qui donnent la réponse du système la plus précise et la plus stable (FIGURE IV.3).

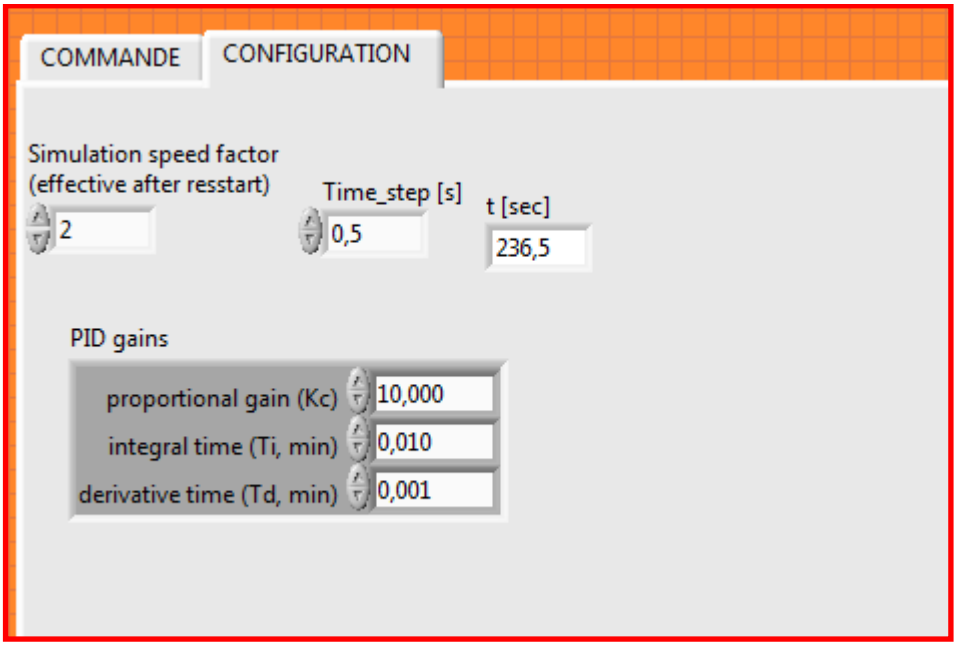


Figure IV.3 : Choix final des paramètres du contrôleur PID.

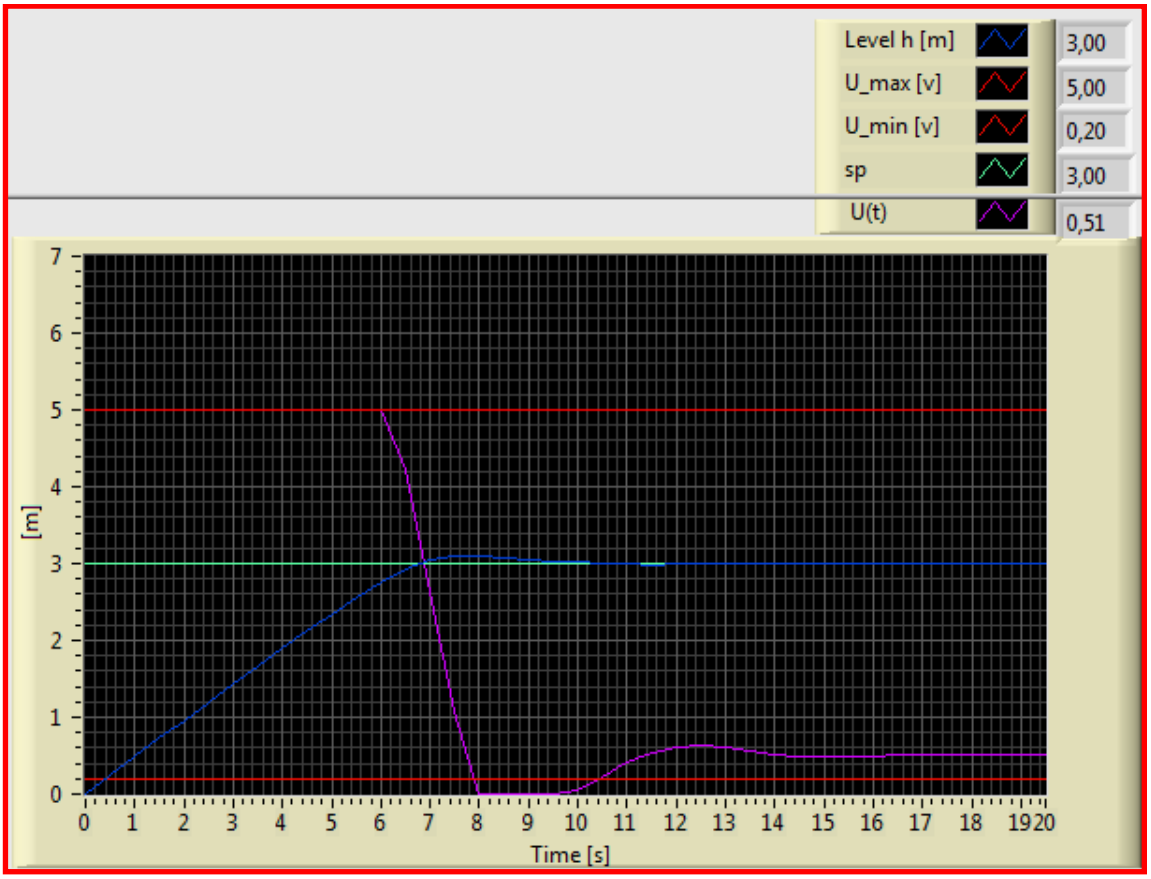


Figure IV .4 : Réponse du système après choix final des paramètres.

La figure (IV. 4) illustre la réponse finale du contrôleur obtenu. On a fixé la consigne à (SP = 3 m) et les paramètres de PID sont ceux illustrés dans la figure (IV. 3), i. e. ***Kp***=10, ***Ki***=0.01 , ***Kd***=0, 001, la période d'échantillonnage : ***Te*** =0.5s. On voit bien que le temps de réponse est aux alentours de : ***11 s***.

IV. 3.1.1-résultats de simulation avec paramètres fixes:

- **En asservissement** : La variation de la consigne.

➢ CAS A : Démarrage avec une consigne fixe.

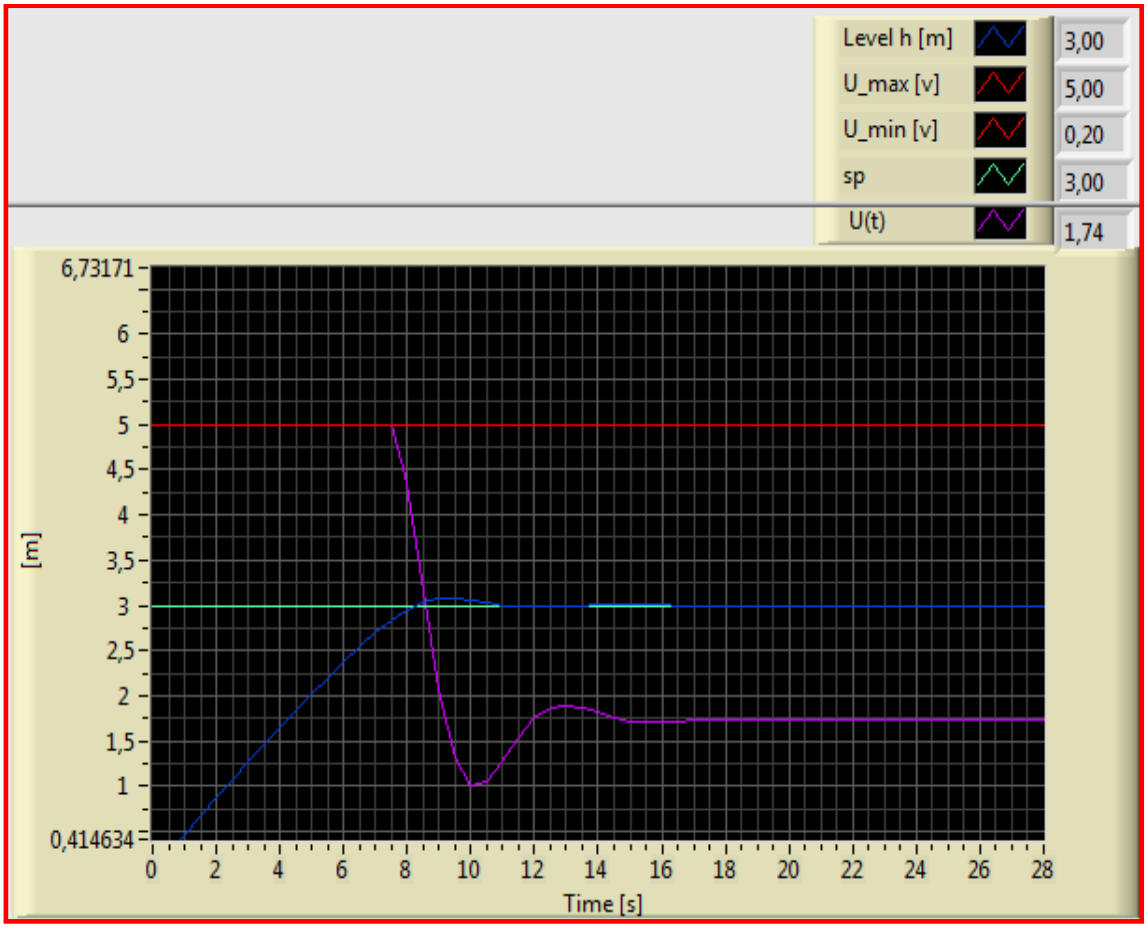


Figure IV.5 : Réponse du système à une consigne de 3 m.

Avec :

- *Level h(m)* : niveaux d'eau.
- *U_max(v)* : La limite maximale du signal de commande *u(t).*
- *U_min(v)* : La limite minimale du signal de commande *u(t).*
- *SP(m)* : La valeur de la consigne.
- *U(t) [v]* : Signal de commande de la vitesse de la pompe.

➢ **CAS B :** Variation brusque de la consigne (de 3 à 4m).

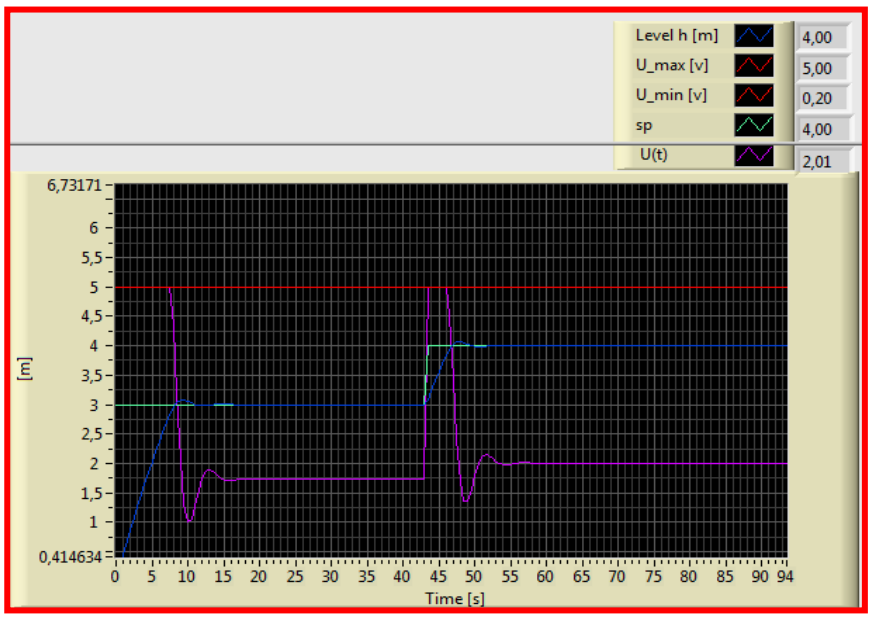


Figure IV.6 : Réponse du système après un changement de consigne de 3m à 4m.

L'operateur effectue un changement de la valeur de la consigne, ce qui correspond à une modification du point de fonctionnement du processus. Si le comportement en asservissement est correct, on démontre que la boucle de régulation réagit bien, même lorsqu'une perturbation se produit

- Le cas A présente un mode d'asservissement normale (suivie de la consigne initiale).
- Le cas B présente un changement de consigne de 3 m au 4 m,

On a remarqué que la commande change au même temps du changement de la consigne.

En régulation : Avec perturbation externe (Débit variable) ***q_in(t)*** variable.

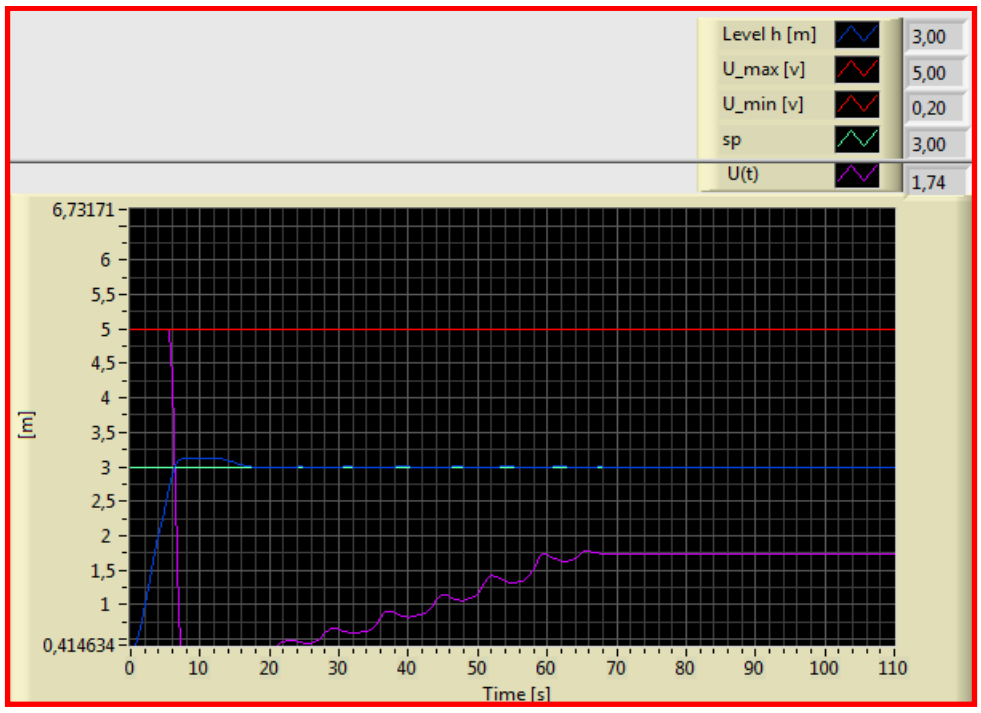


Figure IV.7 : Réponse du système après une variation du débit de la pompe.

La consigne est maintenue constante et il se produit sur le procédé une modification (ou une variation) d'une des entrées perturbatrices. L'aspect régulation est considéré comme le plus important dans le milieu industriel, car les valeurs des consignes sont souvent fixes. Néanmoins, pour tester les performances et la qualité d'une boucle de régulation, on s'intéresse souvent à l'aspect Asservissement.

- Lorsque on aboutie à un état stable on change le debit de la pompe de 50%(Débit q_out=0,86[m3/S] à 100% q_out=1,73[m3/S], on effectue donc une perturbation sur le système.
- On remarque que la commande augmente pour corriger cette perturbation et garder le même niveau correspondant à la consigne.

IV. 3. 2. Configuration des paramètres du contrôleur Flou :

Ce qui a été dit lors de la conception du contrôleur PID est aussi valable pour le cas du contrôleur Flou. La version du logiciel utilisé pour la réalisation de ce travail (LABView 2011) jouit d'une bibliothèque d'outils contrôle-commande très diversifiée. Parmi les outils disponibles on note aussi la présence d'une structure de contrôleurs Flou configurables et prêtes à l'emploi. Les paramètres configurables dans le cas du contrôleur flou sont : La structure externe du contrôleur, le nombre d'entrées est de sortie, le nombre des paramètres flous, la forme et le nombre des fonctions d'appartenance, le nombre et la structure des règles d'inférence flous et la stratégie de défuzzification.

a) **.** Choix de la structure du contrôleur :

Pour notre cas nous avons choisi la structure du contrôleur proposée par ***Mamdani***, qui est un contrôleur à deux entrées : ***l'erreur et la variation de l'erreur***. La sortie du contrôleur est la commande ***u(t)***. Le nombre d'ensembles flous (NEF) pour chaque variable peut prendre une des valeurs possibles suivantes {3, 5, 7, 9} seulement. En effet, dans la littérature la plupart des conceptions de contrôleurs flous utilisent cette plage de valeurs. Nous avons choisi 03 ensembles flous : Les ensembles flous correspondant seront : ***N***, ***Z, P*** indexés de 1 à 3.

b. Choix de la stratégie de fuzzification :

Il s'agit du choix des formes des fonctions d'appartenance, et leur répartition sur l'univers de discours. Dans notre cas, nous avons choisi de travailler sur des univers de discours

normalisé [-6, 6] pour les variables d'entrée et [-3, 3] pour la commande, partitionnés en trois classes (sous ensembles flous) pour les entrées et pour la sortie. Les formes utilisées en commande floue sont la forme triangulaire, gaussienne, trapézoïdale, …etc. La figure ci-dessous montre les partitions floues de l'univers de discours qui ont été utilisées dans notre cas.

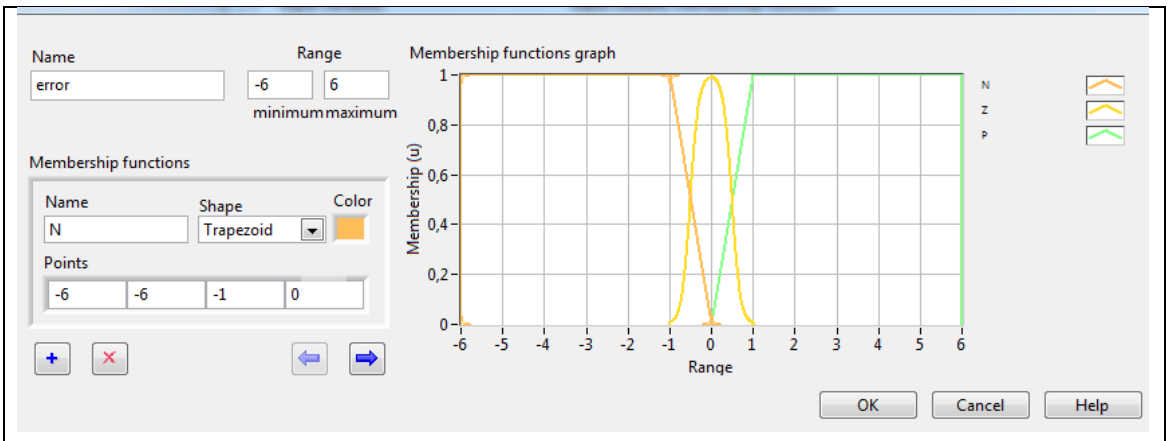


Figure IV.8 : Fonction d'appartenance de l'erreur.

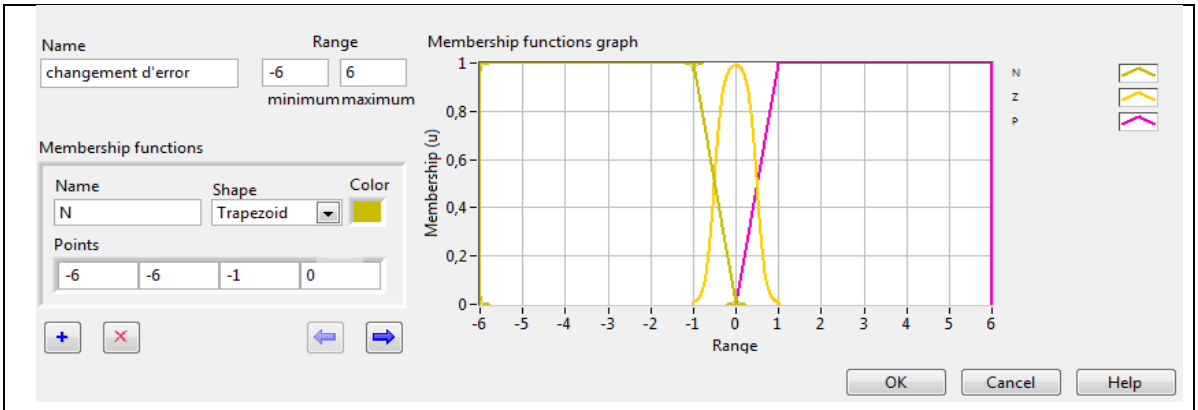


Figure IV.9 : Fonction d'appartenance pour la variation de l'erreur.

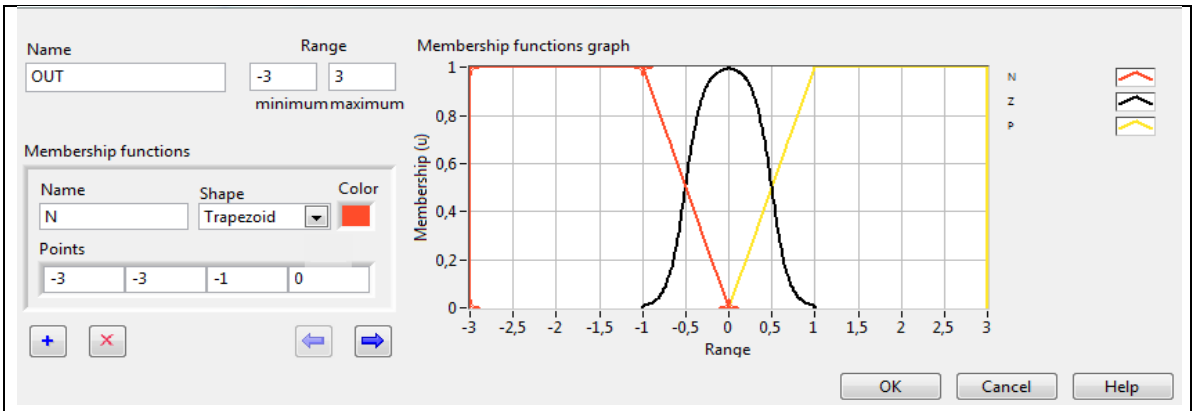


Figure IV.10 : Fonction d'appartenance pour variation de la commande.

c) Etablissement des règles d'inférence :

Sur la base de la description du système à régler avec des variables linguistiques et de la définition des fonctions d'appartenance pour les variables d'entrée et de sortie, on peut établir les règles d'inférence.

Nous avons utilisé une base de règle que l'on trouve dans la plupart des applications où la dynamique du système étudié présente une certaine symétrie autour d'un certain état stable. La table (IV. 2) représente la table des règles retenue dans notre application.

Tableau IV.2 : Base de règles floues.

E/EC	N	Z	P
N	*N*	*N*	*Z*
Z	*N*	*Z*	*P*
P	*Z*	*P*	*P*

d) Choix de la méthode d'inférence :

Les opérateurs *min* et *max* tels qu'ils sont définis par "***Zadeh***" sont utilisés pour inférer les règles, et l'opérateur *max* pour l'agrégation des règles.

e) Choix de la stratégie de défuzzification :

La méthode la plus utilisée en commande flou est celle du centre de gravité, mais comme elle est très coûteuse en terme de temps de calcul, et le but de notre travail était de minimiser le temps de calcul, nous avons opté pour la méthode des centre des aires, qui est dérivée de la méthode du centre de gravité mais qui nécessitent peu de calculs.

IV.2.2-résultats de simulation.

- ✓ En asservissement : Variation de la consigne SP.

- ➢ CAS A : Démarrage du contrôleur avec une consigne fixe (3m).

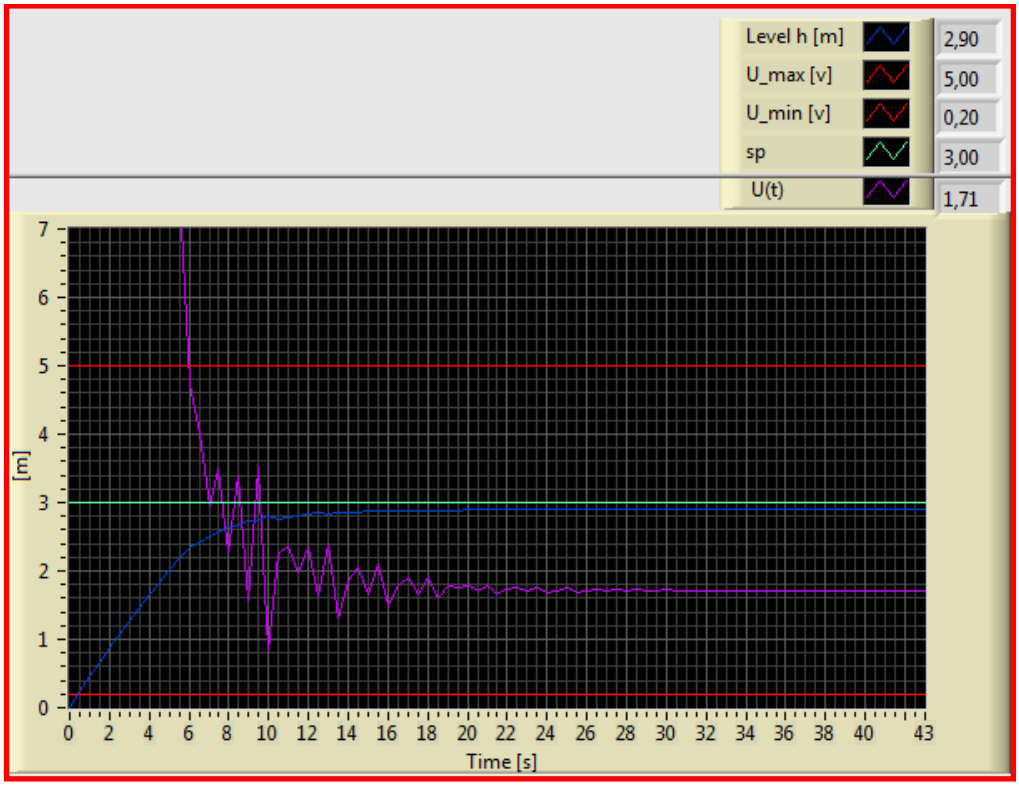


Figure IV.11: Réponse du système à une consigne de 3 m.

- **CAS B :** Variation brusque de la consigne de 3 à 4m.

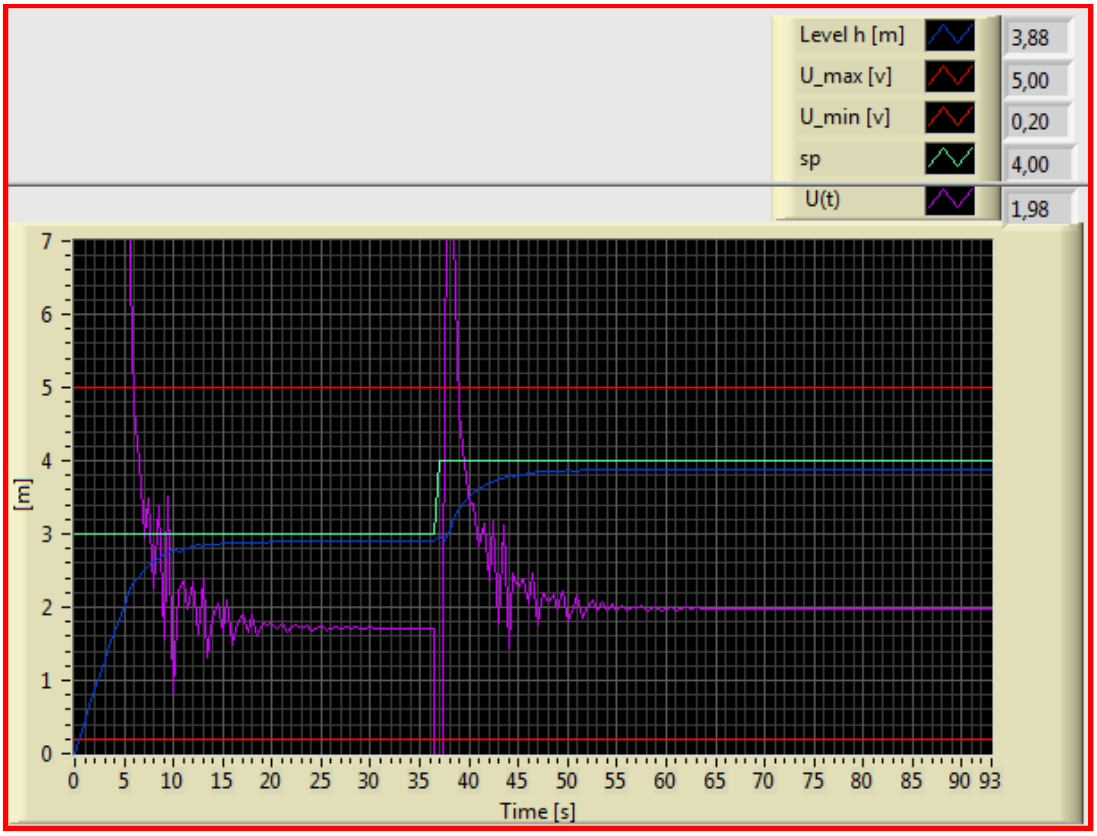


Figure IV.12 : réponse du système après un changement de consigne de 3m à 4 m.

✓ En régulation : Variation du débit de remplissage du réservoir.

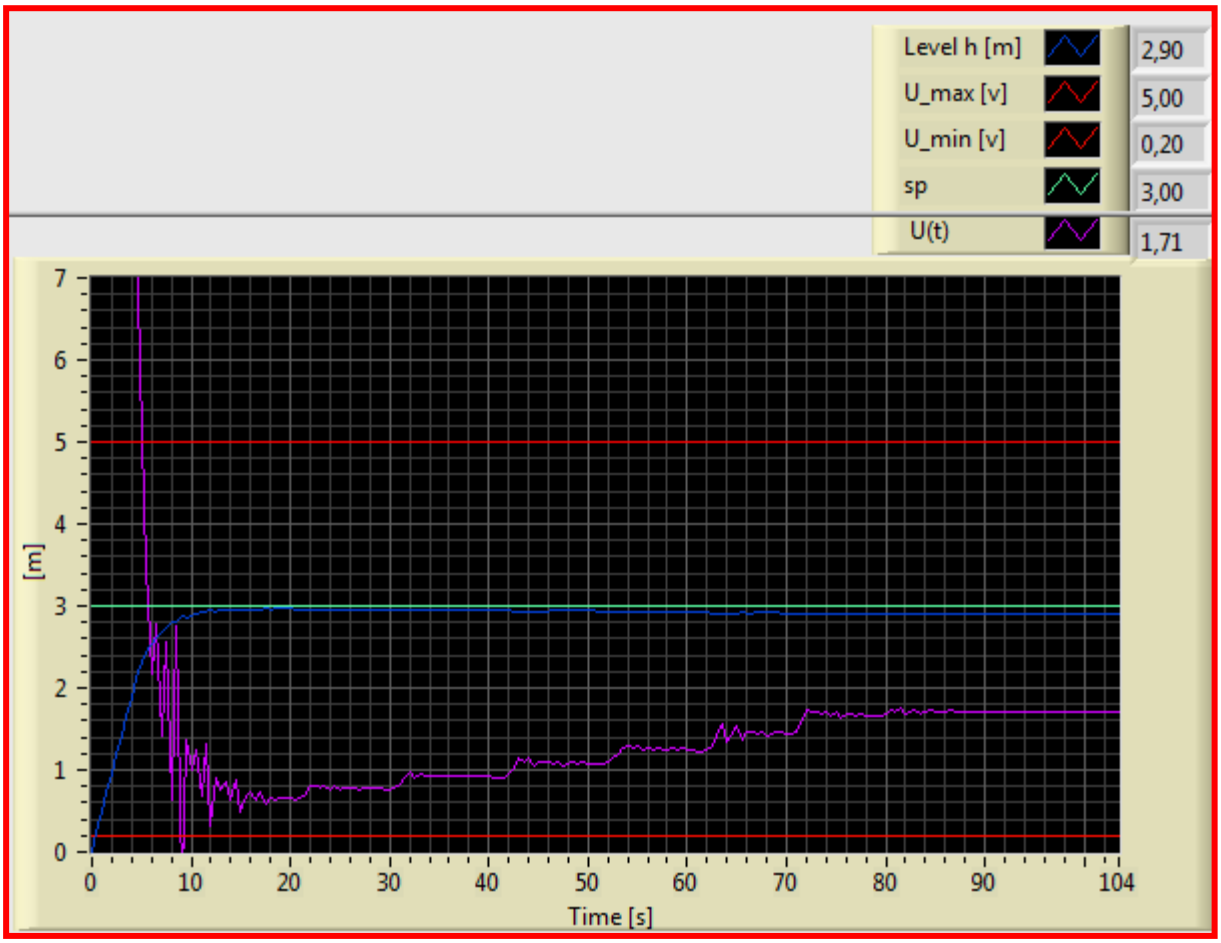


Figure IV.13 : réponse du système après variation de débit de la pompe de remplissage.

A travers cette dernière courbe, on remarque que le contrôleur flou est plus rapide que le contrôleur PID du fait que, dans les même conditions, le niveau atteint sont état d'équilibre dans un temps plus faible (***même pas 10 s***) que celui dans le cas du contrôleur PID qui dépasse les 11s avant d'attendre sont état d'équilibre.

IV. 4 Conclusion :

Si on fait une comparaison entre les deux commandes (Logique floue et PID classique), on se rendra facilement compté que la commande floue corrige mieux la perturbation introduite dans le système et en plus elle répond plus rapidement que la commande PID.

CONCLUSION GENERALE

Le but de notre travail était d'étudier, concevoir et de mettre au point un système de régulation permettant le contrôle continu du niveau d'un fluide dans enceinte avec appoint. Pour arriver à concrétiser un tel système, nous avons en premier lieu modélisé notre système par les équations différentielles régissant son fonctionnement dynamique. Par la suite, nous avons utilisé le logiciel LABView pour simuler le système pris en considération.

L'opération de la régulation du processus (le niveau d'un fluide dans une enceinte donnée) a été faite par le biais de deux contrôleurs, l'un d'eux est basé sur le système d'inférence flou tandis que l'autre était un contrôleur PID Classique.

Nous nous sommes également investis dans le domaine de l'acquisition de données dans le contexte d'un processus industriel. Pour cela, nous avons utilisé une carte d'interface très développées fabriquée par la firme (National Instruments) et ce pour acquérir quelques paramètres liés au processus et même de commander probablement quelques actionneurs.

Les résultats obtenus lors de la phase des tests, nous ont permis de confirmer l'avantage d'une commande floue par rapport à une commande classique PID, notamment lorsque le procédé est non linéaire, ou présentait un retard important. Dans ce cas on peut dire que la commande floue constituera un complément précieux à la commande classique PID. On tient également à signaler que pour des échelons croissants de consigne, les résultats de simulation ont montré que le PID perd ses performances en particulier en termes de dépassement ou précision dynamique alors le régulateur flou reste robuste.

Nous mentionnons aussi par cette fin, que la réalisation de notre projet de fin d'étude au sein du Centre de Recherche Nucléaire de Birine, nous a été très bénéfique sur plusieurs plans a savoir : On a pu développer un esprit de travail d'équipe et de consultation continue, chose qui a accéléré l'aboutissement de notre travail. On a appris à lire et à tirer profit, des documents techniques et des data-sheets de plusieurs équipements et composants électroniques. On s'est également familiarisé avec d'autres outils de développements (que ça soit hard ou soft), qui nous ont été jusqu'ici méconnus. Parmi les concepts et les notions qu'on a pu acquérir, on note bien : La commande floue qui constitue une nouvelle de commande des procédés industriels et qui utilise des techniques s'appuyant sur le savoir-faire humain, exprimées en mots et en phrases ordinaires, plutôt que sur des équations et des données numériques bien précis.

La tâche ainsi réalisée, n'est qu'une partie d'un travail énorme qui consiste en la rénovation de l'instrumentation de mesure et les techniques de commande autour du réacteur Es-salam. Elle sera suivie par une implémentation pratique réelle sur site avec les capteurs adéquats.

Bibliographie

[1] John J. D'azzo Constantine H. « *Linear Control System Analysis and Design: Fifth Edition, Revised and Expan ded* » ISBN: 0824740386, Marcel Dekker Inc Edition 2003.

[2] LabVIEW$_{TM}$ PID and Fuzzy Logic Toolkit User Manual, National Instruments Worldwide Technical Support and Product Information, ISBN : 372192D-01,June 2009.

[3] ZADEH (L.A.) – Fuzzy sets. Inform. Control. 8, p. 338-353 (1965).

[4]MAMDANI (E.H.). – Application of fuzzy algorithms for control of a simple dynamic plant. Proc. IEEE, 212, p. 1585-1588 (1974).

[5]SUGENO (M.) (éditeur). – Industrial applications of fuzzy control. North Holland Pub. Comp. (1992).

[6] Mireille BARRA, Yves LÉCLUSE, Jean-Pierre BARRAT « Techniques DE l'Ingénieur ' expertise technique et scientifique de référencer-7428 ,Exemple d"application de la logique floue ,commande de la température d"un four pilote. (10/07/1993) .

[7] DERLEANS(P) , BARRAT(J .P) et LECLUSE(Y),-commande par calculateur thermique R.A.P.A.2.P-65-67(1989).

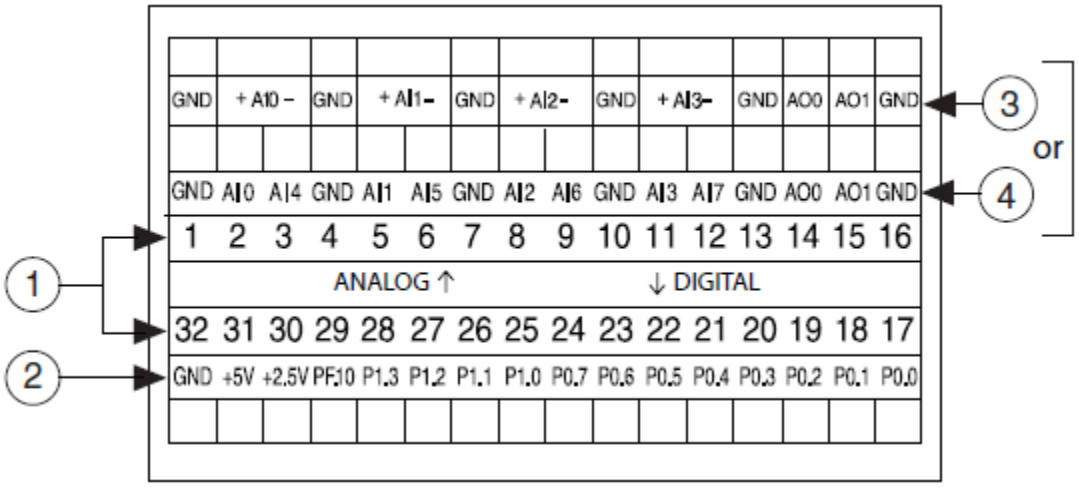
[8]BARRAT(M).DORLEANSE(p) VILLENEUVE(G) et LECLUSE(Y). partial state model refe . Rance contrôle of an industriel oven proc. Instr. MERCH ENGRS 206.P.145-149 (1992).

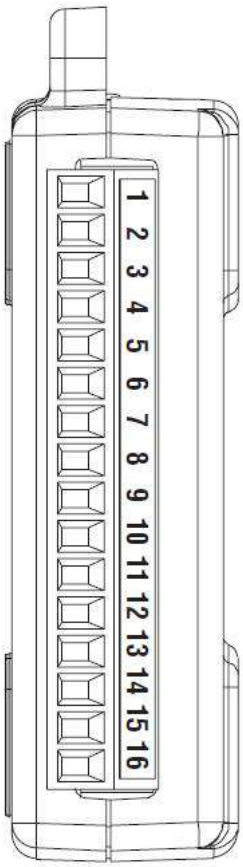
[9] Guide d'utilisateur du logiciel labview.

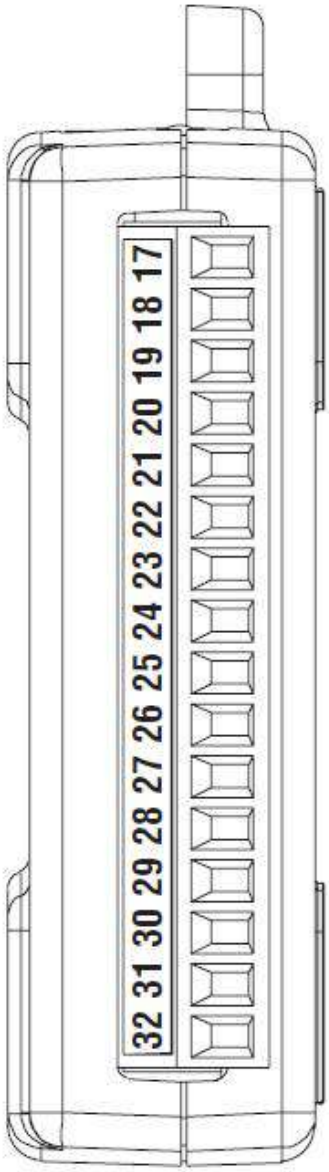
- **Annexes**

Carte d'acquisition

Documentation technique d'un boitier NI USB-6008



Module	Terminal	Signal, Single-Ended Mode	Signal, Differential Mode
	1	GND	GND
	2	AI 0	AI 0+
	3	AI 4	AI 0–
	4	GND	GND
	5	AI 1	AI 1+
	6	AI 5	AI 1–
	7	GND	GND
	8	AI 2	AI 2+
	9	AI 6	AI 2–
	10	GND	GND
	11	AI 3	AI 3+
	12	AI 7	AI 3–
	13	GND	GND
	14	AO 0	AO 0
	15	AO 1	AO 1
	16	GND	GND

Module	Terminal	Signal
	17	P0.0
	18	P0.1
	19	P0.2
	20	P0.3
	21	P0.4
	22	P0.5
	23	P0 6
	24	P0.7
	25	P1.0
	26	P1.1
	27	P1.2
	28	P1.3
	29	PFI 0
	30	+2.5 V
	31	+5 V
	32	GND

Analog Output

Analog outputs	2
Output resolution	12 bits
Maximum update rate	150 Hz, software-timed
Output range	0 to +5 V
Output impedance	50 Ω
Output current drive	5 mA
Power-on state	0 V
Slew rate	1 V/µs
Short circuit current	50 mA
Absolute accuracy (no load)	7 mV typical, 36.4 mV maximum at full scale

E/S numériques

Lignes d'E/S numériques

P0.<0..7>.....8 lignes

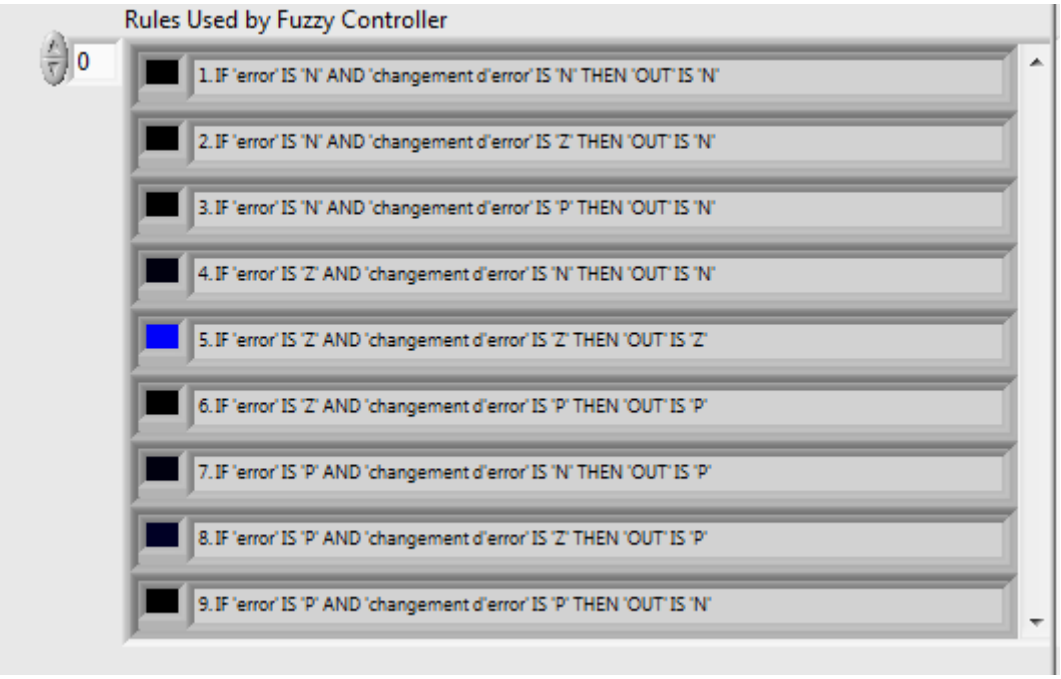
P1.<0..3>.....4 lignes

Contrôle de la direction.....Chaque voie est programmable individuellement comme entrée ou sortie

Type de pilotage en sortie1

NI USB-6008 .....Collecteur ouvert .

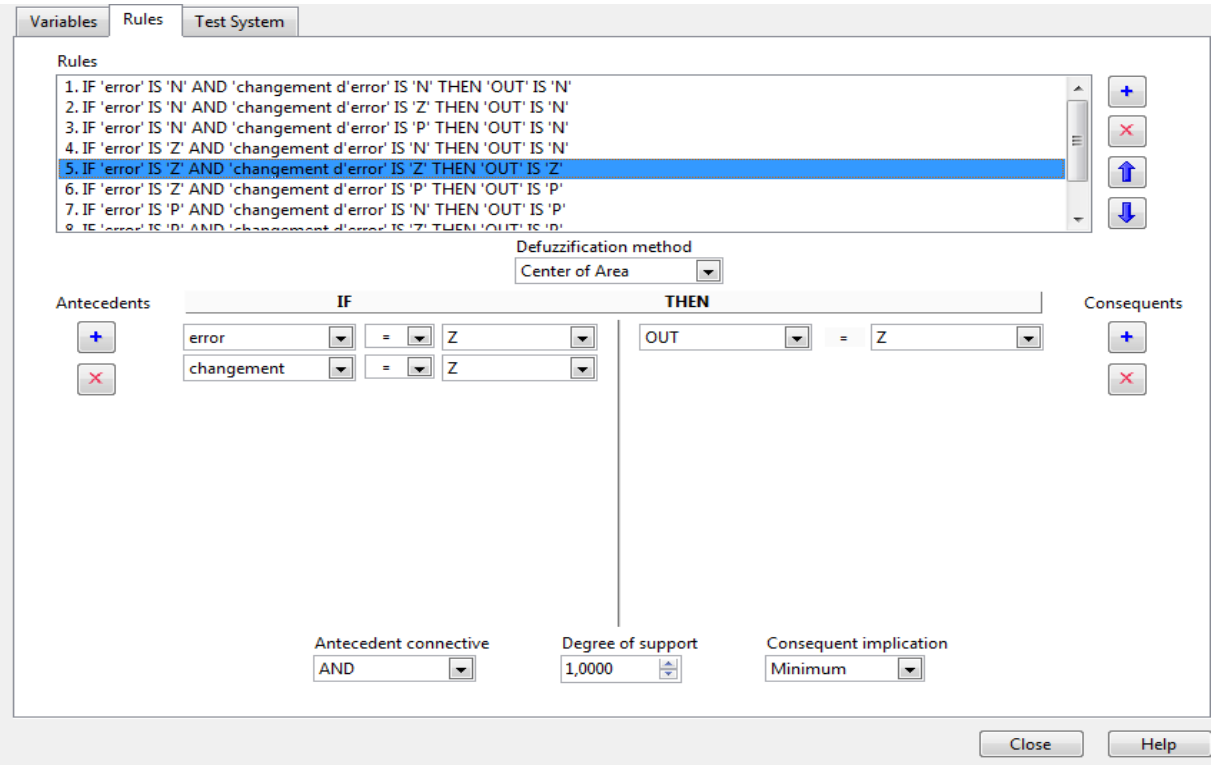
Table de commande par logique flou :



- On utilise dans ce bloc l'ensemble des règles et les sous ensembles flous provenant de la fuzzification pour calculer les sous ensembles flous relatifs à la commande.
- SI *e* (erreur) est Négative ET *Δe* (d_erreur) est Négative ALORS la commande *Δu* est Négative

On a clique dans la page de commande TOOLS__control design and simulation__ fazzy Designer_ system puis charger le système

On obtient le schéma sous dessous


Variables
Rules
Test System
Rules
1. IF 'error' IS 'N' AND 'changement d'error' IS 'N' THEN 'OUT' IS 'N'
2. IF 'error' IS 'N' AND 'changement d'error' IS 'Z' THEN 'OUT' IS 'N'
3. IF 'error' IS 'N' AND 'changement d'error' IS 'P' THEN 'OUT' IS 'N'
4. IF 'error' IS 'Z' AND 'changement d'error' IS 'N' THEN 'OUT' IS 'N'
5. IF 'error' IS 'Z' AND 'changement d'error' IS 'Z' THEN 'OUT' IS 'Z'
6. IF 'error' IS 'Z' AND 'changement d'error' IS 'P' THEN 'OUT' IS 'P'
7. IF 'error' IS 'P' AND 'changement d'error' IS 'N' THEN 'OUT' IS 'P'
Defuzzification method
Center of Area
Antecedents
IF
THEN
Consequents
error = Z
changement = Z
OUT = Z
Antecedent connective
AND
Degree of support
1,0000
Consequent implication
Minimum
Close
Help

Apres l'état précédant on a cliqué sur test system de meme schéma on obtient

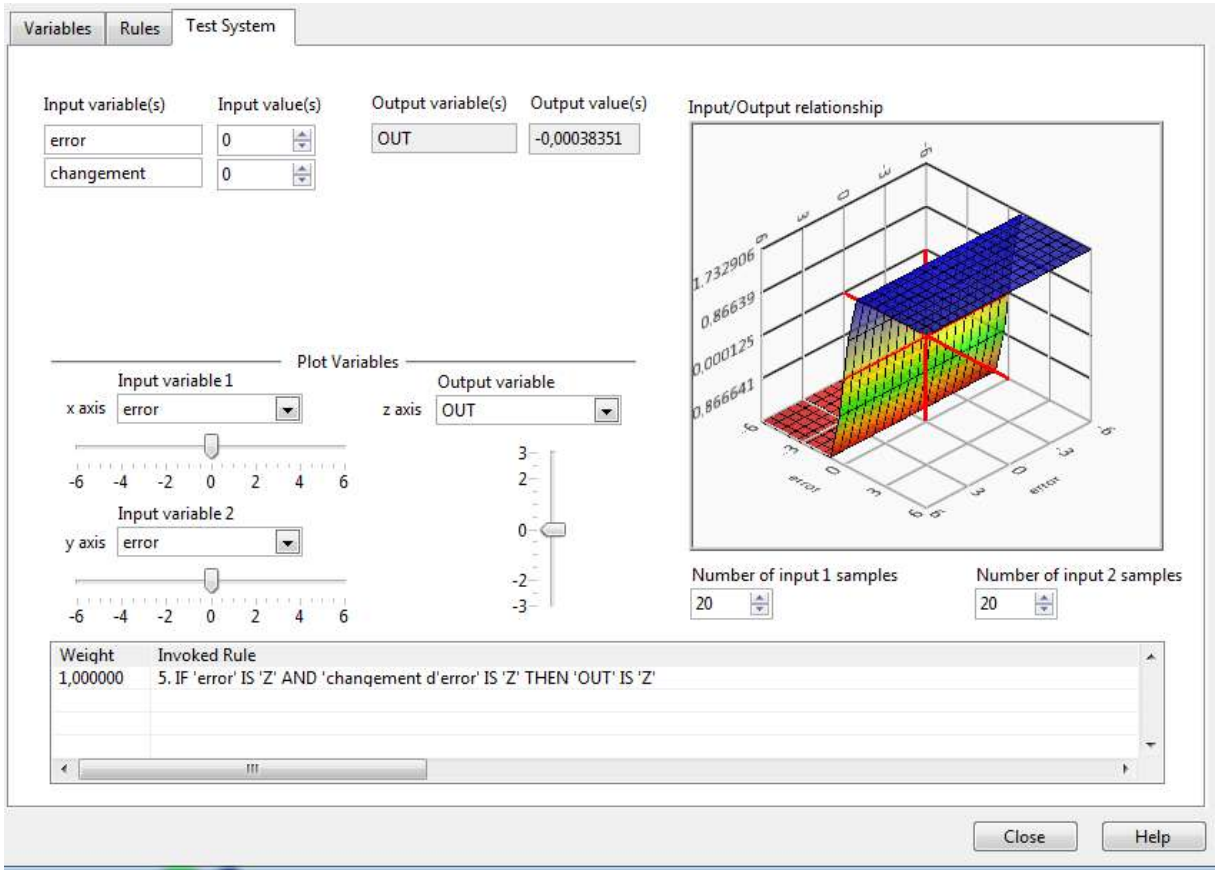


Schéma de test de système

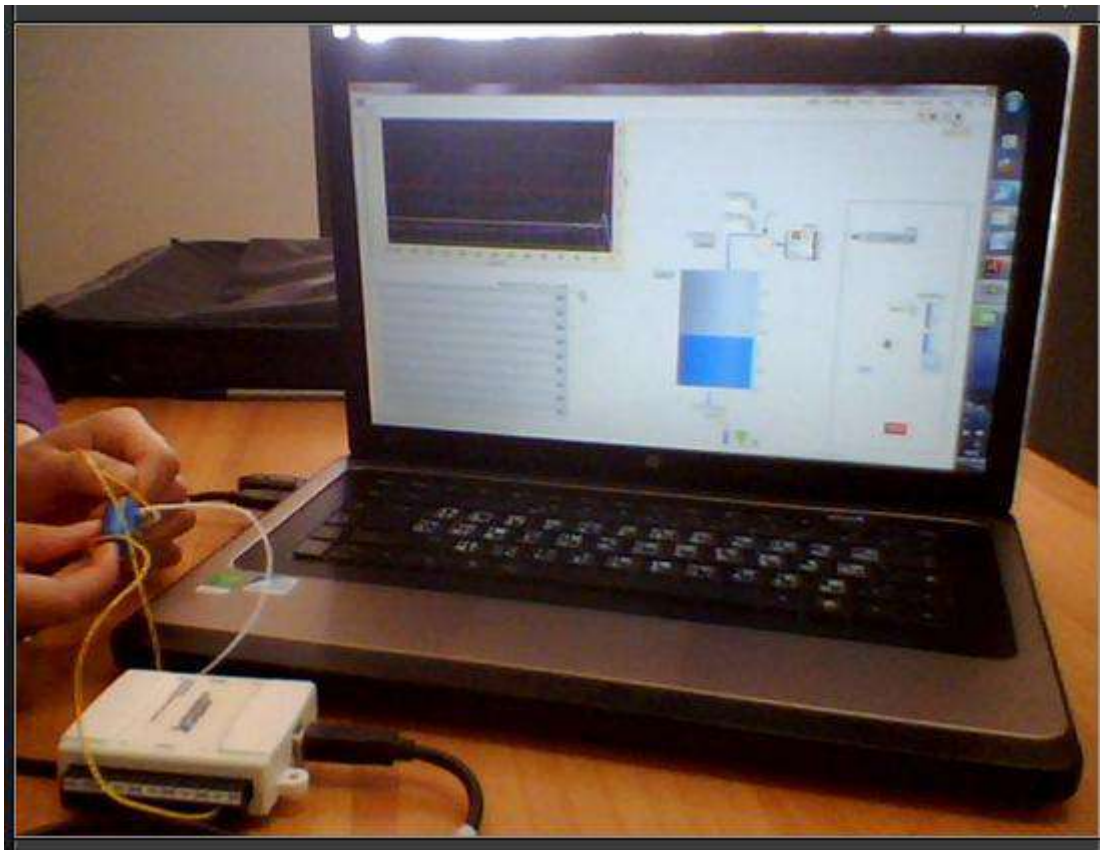


Schéma démontre tous les outilles qui utilisent pour l'expérimentale.

ANNEXES