République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Mouloud MAMMERI, Tizi-Ouzou

Faculté de Génie Electrique et d'Informatique
Département d'Automatique

Mémoire de Fin d'Etudes

En vue de l'obtention du diplôme

De Master Professionnel en Automatique
Option : Automatique et Informatique Industrielles

Thème

Etude et Développement d'une Plate Forme de Régulation de la Station de Pression PUP-4 /EV d'ElettronicaVeneta Sous Environnement LabVIEW

Proposé et dirigé par : **M. ALLAD. M**

Présenté par :
M. ACHIR Aghiles
M. MAOUCHE Naim

Membres de jury
Président : M^r^ BENSIDHOUM M. OTAHAR
M^r^ CHARIF MOUSSA
M^r^ SAIDI KHIRDINE

Soutenu le : 06 / 10 / 2013

Promotion 2013

Ce travail a été préparé au sein du Laboratoire d'Automates Programmables et de Régulation Industrielle, Département d'Automatique, Université de Tizi-Ouzou

Remerciements

En préambule à ce mémoire, nous souhaitons adresser nos remerciements les plus sincères aux personnes qui nous ont apporté leur aide et qu'ont contribué à l'élaboration de ce mémoire ainsi qu'à la réussite de cette formidable année universitaire.

Nous tenons à remercier M. ALLAD.M, qui, en tant que promoteur de mémoire, s'est toujours montré à l'écoute et très disponible tout au long de la réalisation de ce mémoire, ainsi pour l'inspiration, l'aide et le temps qui l'a bien voulu nous consacrer.

Nous remercions d'avance, les membres de jury d'accepter d'examiner notre travail. Merci également aux membres et responsables du « Laboratoire d'Automates programmables et de la Régulation Industrielle » qui ont largement contribué à la bonne ambiance de travail au quotidien.

Sans oublier de présenter tout notre respect et hommages à tous les enseignants qui nous ont transmis leur savoir durant nos cinq années d'étude.

Dédicaces

Je dédie ce travail

Je tiens en tout premier lieu à remercier mes chers parents pour m'avoir toujours encouragé à poursuivre mes études

A mes chers Frères, mes Chères Sœurs et à toute ma famille

A mon ami et binôme AGHILES

A tous mes amis chacun par son nom.

Maouche Naim

Dédicaces

Je dédie ce mémoire à tous ceux qui me sont chers.
Je tiens en tout premier lieu à remercier mes chers parents pour m'avoir toujours encouragé à poursuivre mes études et m'avoir permis d'arriver là où j'en suis, qu'ils trouvent aujourd'hui dans ce travail le témoignage de mon profond attachement.
Je dédie ce mémoire :

A mon frère ASSALAS et ma sœur TINHINANE

A mon binôme MAOUCHE NAIM

A tous mes amis.

ACHIR AGHILES

Liste d’indices

G_S: Gain statique.
θ: Constante de temps.
τ : Temps de retard.
R : Coefficient de pente.
θ_d : Constante de temps désirée.
SP : SetPoint.
PV : Process Variable.
$X(t)$: Mesure.
$W(t)$: Consigne.
$Y(t)$: Commande.
T_i: Temps d’intégration.
T_d : Temps de dérivation.
Sd: Différentiel statique.
Dd : Différentiel dynamique.
G_m : Marge de gain.
φ_m : Marge de phase.
f_{ech} : Fréquence d’échantillonnage.
f_c: Fréquence de coupure.
T_s : Période d'échantillonnage.
T: Durée de cycle.
G_r : Gain du régulateur.
K_c : Action proportionnelle.
Δ : Pas d’échantillonnage.
ω_n : Pulsation propre.
h : Facteur d’amortissement.
V_e : Tension d’entrée.
V_S : Tension de sortie.

Sommaire

Chapitre 2 : Présentation du logiciel LabVIEW

Chapitre 3 : Description matérielle

Chapitre 4 : Simulations et résultats expérimentaux

Introduction Générale

Les technologies modernes ont permis le développement des sciences tout en imposant l'exploration des domaines théoriques de plus en plus complexes. Parmi ces sciences en pleine expansion et intégrant l'apport des technologies modernes, on compte l'automatique. Le mot « Automatique » a été utilisé pour la première fois en 1914 dans un article « Essai sur l'Automatique » publié dans une revue scientifique.

De nos jours, l'automatique fait partie des sciences de l'ingénieur. Cette discipline traite de la modélisation, de l'analyse, de la commande et de la régulation des systèmes dynamiques. Elle a pour fondements théoriques les mathématiques, la théorie du signal et l'informatique.

Réguler une grandeur, c'est d'obtenir d'elle un comportement voulu, dans un environnement susceptible présentant des variations paramétriques et des perturbations.

L'objectif de la régulation est de maintenir le plus près possible une grandeur commandée, ou une grandeur réglée à une valeur prédéfinie par le cahier des charges. Répondre à des changements d'objectifs, ou à un objectif variable tel que la poursuite d'une cible, on parle d'un fonctionnement d'asservissement.

Le logiciel de programmation LabVIEW dispose d'un nombre important de fonctions graphiques, ce qui nous permet de programmer facilement des applications pour acquérir des données, de les traiter et d'afficher les résultats. Sachant que la plupart des maquettes destinées à la pédagogie ou pour la recherche par des sociétés étrangères telles que : Gunt (Allemande), TecQuipment (Anglaise), EDIBON (Espagnol), ElettronicaVeneta (Italienne)…etc, utilisent des plates-formes de commande à base de LabVIEW, sans oublier de citer la géante société Américaine National Instruments qui donne un grand intérêt à ce logiciel pour développer des applications.

Notre travail consiste à identifier un système de pression PUP-4 d'ElettronicaVeneta, implémenter deux régulateurs TOR et PID sous automate, ensuite créer une plateforme de commande avec LabVIEW pour commander la station.

Pour mener à terme notre travail, nous avons organisé notre mémoire de la façon suivante :

Le premier chapitre consiste à faire des rappels sur l'identification et la régulation des systèmes.

Le deuxième chapitre présente le logiciel LabVIEW, c'est-à-dire son contenu que nous allons l'utiliser pour la réalisation de la plate forme de commande.

Le troisième chapitre est consacré à la description matérielle. On commence par décrire la station de pression PUP-4/EV, puis on présente la carte d'acquisition Labjack-U3 et ses différents blocs de programmation sous LabVIEW. Enfin, nous avons présenté des circuits électroniques d'adaptations.

Le quatrième chapitre traite des simulations et des résultats expérimentaux dans lequel nous avons identifié notre système par une fonction de transfert. Ensuite, nous avons implémenté une régulation TOR et un régulateur PI sous MATLAB, automate S7-200 et sur une plate-forme de commande LabVIEW, puis une étude comparative a été faite.

Nous terminons notre travail par une conclusion générale discutant les résultats obtenus et donnant une perspective pour des travaux futurs.

Chapitre 1

Identification et Régulation des systèmes

1.1. Introduction

La fonction de transfert idéale d'un procédé industriel est pratiquement impossible à déterminer. Il est alors nécessaire d'utiliser un modèle qui soit le plus représentatif possible de ce procédé.

Identifier un procédé, c'est de déterminer à partir de données expérimentales, les paramètres qui caractérisent son modèle.

L'objectif de l'identification est de calculer les paramètres d'un modèle du procédé, à partir des données expérimentales, de façon à ce que le comportement du procédé est celui de modèle soient identiques, et ceci pour toutes les séquences de variables d'entrée habituellement utilisées.

La régulation des procédés industriels regroupe l'ensemble des moyens matériels et techniques mis en œuvre dans le but de :

Maintenir une grandeur physique à régler (débit, pression, température… etc.) à une valeur désirée (consigne), malgré les perturbations ou changements de consigne.

Fournir à l'opérateur des informations (fonctionnement, alarmes (visuelles ou sonores).

1.2. Modélisation et identification

1.2 .1.modélisation

L'automaticien a besoin d'un modèle pour concevoir un régulateur à mettre en œuvre afin d'atteindre les objectifs décrits dans un cahier des charges, Les modèles les plus utilisés en automatique sont les modèles de connaissances et les modèles de représentation. Ainsi, le modèle de connaissance est peu utilisé car les équations physiques régissant les processus ne sont pas toujours facile à obtenir, contrairement au modèle de représentation qu'est basé sur la connaissance expérimentale des entrées/sorties, pour élaborer ce modèle on a besoin d'utiliser les méthodes d'identification.

1.2.1.1. Modélisation pour la commande des procédés

Le développement d'un modèle pour un système physique peut être réalisé pour différentes raisons :

- Avoir une meilleure compréhension des phénomènes.
- Dimensionnement d'une installation.
- Formation des opérateurs.
- Conception du système de commande, cette dernière a pour objectif :
  - La mise au point de la stratégie de commande.
  - Conception de la loi de commande et son réglage.
  - Conception de capteurs logiciels ou estimateur d'état du système.

1.2.1.2. Différents types de modèles

Il est important de préciser que les modèles dont nous aurons en général besoin sont de type dynamique, permettant de représenter l'évolution d'un système dans le temps.

Dans la catégorie des modèles dynamiques, il est important de distinguer différents types qui ont pour objectif de décrire le système avec plus ou moins de détails.

On a trois types de modèles, modèle de connaissance, modèle de comportement et modèle intermédiaire.

A) Modèles de connaissance ‘’boite blanche’’

Ils sont élaborés à partir des lois de la physique ou de la chimie. Leur objectif principal est d'expliquer un phénomène par une relation mathématique. Les équations physiques ne sont pas toujours données par le fournisseur, elles conduisent souvent à des développements mathématiques trop complexes pour être exploitées au sens de l'automatique.

B) Modèles de comportement ‘’boite noire’’

Ce sont des modèles linéaires, dont la validité reste limitée à de petites variations autour de point de fonctionnement. Les petites variations de l'entrée autour d'un point de fonctionnement peuvent être reliées à de petites variations de la sortie par un modèle dynamique linéaire.

C) Modèles intermédiaires ‘’boite grise’’

Ils constituent un hybride entre les deux types précédents. On peut les considérer comme des modèles de connaissance simplifiés.

1.2.2. Identification des procèdes

1.2.2.1. Modèle du premier ordre [3]

La réponse d'un système du premier ordre soumis à un échelon, est donné comme suit :

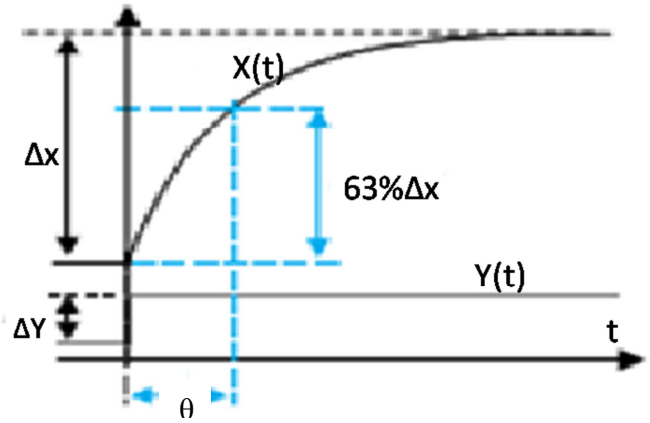


Figure **1.1** : Système du premier ordre

Le modèle du premier ordre est de la forme :

$$G(p)=\frac{Gs}{(1+\theta p)} \qquad (1.1)$$

La détermination des paramètres de modèle se fait comme suit :

- Le gain statique est mesuré directement par :

$$Gs=\frac{\Delta y}{\Delta x} \qquad (1.2)$$

- La constante de temps θ: Comme la pente à l'origine peut être difficile à déterminer avec précision, on trace conjointement la droite d'ordonnée (0.63 Δx) parallèle à l'axe des abscisses. Cette construction permet de déterminer la constante de temps θ.

1.2.2.2 Modèle du second ordre

La réponse d'un système du second ordre soumis à un échelon, est donnée comme suit:

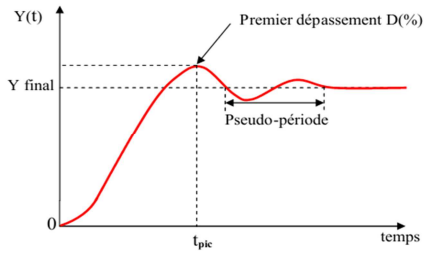


Figure **1.2.** Réponse d'un système du deuxième ordre à un échelon pure

La forme canonique du modèle du second ordre est de la forme :

$$G(p) = \frac{Gs}{1+\frac{2h}{\omega_n}P+\frac{P^2}{\omega_n^2}} \quad (1.3)$$

• Le facteur d'amortissement h se détermine à l'aide de la mesure du dépassement :

$$D\%=100e^{\frac{-\pi h}{\sqrt{1-h^2}}} \quad (1.4)$$

• La pulsation propre ω s'obtient via l'une de deux formules :

$$T_{pic} = \frac{\pi}{\omega_n\sqrt{1-h^2}} \quad (1.5)$$

$$T_{pseudo-période} = \frac{2\pi}{\omega_n\sqrt{1-h^2}} \quad (1.6)$$

1.2.2.3. Méthode de Broïda [3]

Dans le domaine industriel la plus part des cas utilise la méthode de Broïda par rapport aux autres.

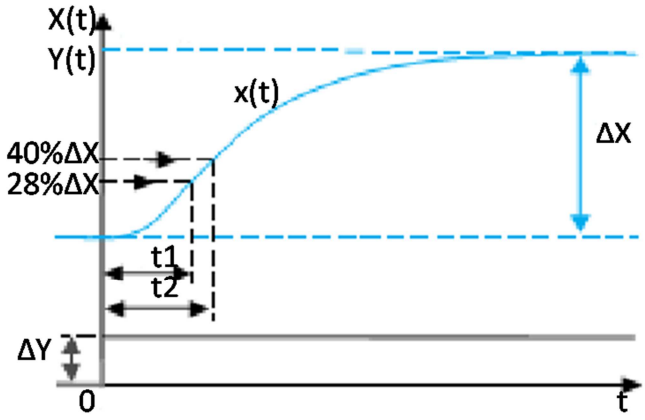


Figure **1.3** : Système sans dépassement avec un point d'inflexion

En passant d'un ordre n à un 1^{er} ordre. Il a estimé que :

- La tangente au point d'inflexion était une source d'erreur importante,
- La durée des essais pouvait être longue sur les systèmes lents avec le risque d'avoir une entrée qui varie pendant l'essai.

Le modèle proposé pour approcher le comportement du système est un premier ordre avec retard τ, sa fonction de transfert est :

$$G(p)=\frac{Gs.e^{-\tau p}}{(1+\theta P)} \quad (1.7)$$

Ses calculs montrent que pour obtenir les temps t_1 et t_2, on prend respectivement pour Y(t) les valeurs $0.28\Delta x$ et $0.40\Delta x$

- Le gain statique G_s, calcul comme suit :

$$Gs=\frac{\Delta y}{\Delta x} \quad (1.8)$$

- La constante de temps (θ) :

$$\theta = 5.5\ (t_{40\%}-t_{28\%}) \quad (1.9)$$

- Le retard (τ) :

$$\tau = (2.8\ t_{28\%} - 1.8\ t_{40\%}) \quad (1.10)$$

1.2.2.4. Modèle d'ordre supérieur ''Méthode de Strejc''

Cette méthode permet l'identification d'un processus dont la réponse à l'échelon n'a pas de dépassement

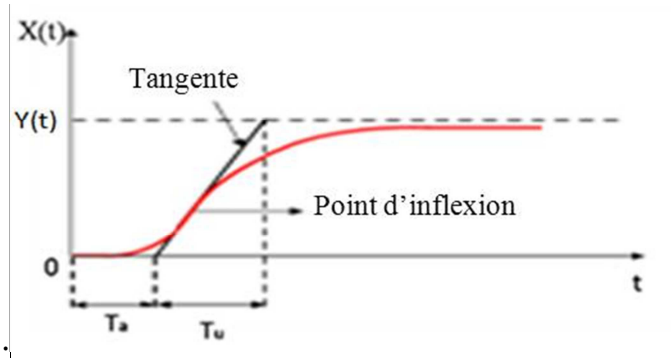


Figure **1.4** : Réponse d'un système sans dépassement

La méthode de Strejc consiste à caractériser le procédé par un modèle de la forme :

$$G(P) = \frac{Gs.e^{-\tau.p}}{(1+\theta P)^n} \quad (1.11)$$

- **Le principe de la méthode de Strejc**

La méthode de Strejc est basée sur 4 éléments essentiels pour déterminer le modèle correspondant

- Tracer la tangente au point d'inflexion, qui permet de définir les deux grandeurs (Tu) et(Ta), (Figure **1.4**)
- On calcule le rapport

$$\eta = \frac{Tu}{Ta} \quad (1.12)$$

- On cherche dans le (tableau 1.1) le rapport $(\frac{Tu}{Ta})$ immédiatement inférieur à la valeur calculée (η). Cette ligne permet d'obtenir l'ordre (n) du modèle. La constante de temps est calculée à partir de la troisième colonne (tableau **1.1**).
- Le retard est égale à :

$$\tau = T_{u,mesure} - \frac{Tu}{Ta} /_{tableau,\ Ta\ mesuré} \quad (1.13)$$

Le tableau suivant permet d'obtenir la constante du temps et l'ordre (n).

$\frac{T_u}{T_a}$	Ordre du modèle « n »	$\frac{\theta}{T_a}$
0	1	1
0.105	2	0.37
0.22	3	0.27
0.32	4	0.22
0.41	5	0.20
0.49	6	0.18
0.57	7	0.19
0.64	8	0.15
0.71	9	0.14
0.77	10	0.13

Tableau **1.1 :** Coefficients de la méthode de Strejc

1.2.2.5. Méthodes de réglage de PID

Pour un choix des paramètres bien adéquat, il est possible d'obtenir un comportement désiré en choisissant une boucle fermée, caractérisant les performances du système de régulation.

Les critères à satisfaire sont les suivants :

• La mesure doit être égale à la consigne.

- Les effets des perturbations doivent être minimisés.
- La sollicitation des actionneurs doit être raisonnable.
- Le réglage doit être maintenu, c'est-à-dire ne pas faire d'ajustements trop fréquents.

1.2.2.6. Méthodes basées sur un modèle de réponse à l'échelon

- **Méthode de Ziegler-Nichols en boucle ouverte**

Cette méthode consiste à approximer la réponse de processus en boucle ouverte à un échelon, que l'on suppose apériodique, par un modèle de type :

$$G(p)=\frac{Gs.e^{-\tau p}}{(1+\theta P)} \qquad (1.14)$$

Où :

G_s : Gain statique.

θ: Constante de temps.

τ : Temps de retard

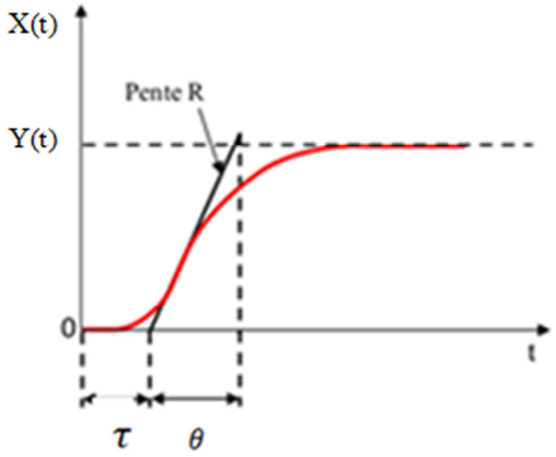


Figure **1.5 :** Réponse d'un procédé stable et apériodique à une entrée en échelon

Le coefficient de pente R est défini comme étant :

$$R=\frac{Gs}{\theta} \qquad (1.15)$$

Pour obtenir les paramètres des régulateurs P, PI ou PID, il suffit d'appliquer les relations de tableau suivant.

Ces relations ont été développées empiriquement pour donner une réponse en boucle fermée oscillante, avec dépassement initial de l'ordre 30 à 40% et avec un rapport

d'amplitude d'oscillations de 1/4 (rapport entre le dépassement de deux pics de même signe).

Type de régulateur	Gs	Ti	Td
P	$\frac{1}{\tau . R}$	----	----
PI	$\frac{0.9}{\tau . R}$	$3.33\ \tau$	----
PID	$\frac{1.2}{\tau . R}$	2τ	0.5τ

Tableau **1.2 :** Réglage de Ziegler/Nichols à partir d'une réponse indicielle

- **Méthode du modèle de référence [3]**

En imposant un modèle du premier ordre en boucle fermée $F(p) = \frac{1}{1+\theta_d P}$ **,** la fonction de transfert du correcteur devient :C (p) = $\frac{1}{\theta_d . P .\ H(p)}$ soit C(p)=$\frac{1+\theta P}{Gs.\theta_d P}$

La fonction C (p) correspond à un régulateur PI dont les valeurs des coefficients dépendront de la structure du régulateur réel disponible

	Gr	Ti
PI parallèle	$\frac{1}{Gs}\frac{\theta}{\theta_d}$	$Gs\ \theta_d$
PI série	$\frac{1}{Gs}\frac{\theta}{\theta_d}$	θ
P	$\frac{1}{Gs}\frac{\theta}{\theta_d}$	—

θ_d: *constatnte de temps*

H(p) : fonction de transfert du procédé

F(p) : la fonction de transfert en boucle fermée

Tableau **1.3** Paramètres du régulateur

Exemple

Soit la fonction de transfert procédé: $H(p) = \frac{2}{1+30P}$, $\theta_d = 10\ Sec$ (en boucle fermée)

La fonction de transfert correspondante est $C(p) = \frac{1+30P}{2\theta_d P}$

Pour un PI série on écrit : $C(p) = G_r \frac{1+Ti.P}{Ti.P} = \frac{30}{20}.\frac{1+30p}{30p}$

Le régulateur donc doit être configuré avec :$G_r = 1.5, T_i = 30\ Sec$

1.3. Régulation

1.3.1. Définition

La régulation automatique prend une grande importance dans le domaine de l'industrie. Réguler une grandeur, c'est obtenir d'elle un comportement voulu. Pour effectuer des tâches de régulation, ce n'est pas la connaissance de nombreuses formules et méthodes de calcul qu'est importante, mais la compréhension des relations de cause à effet dans la boucle de régulation.

1.3.2. Objectifs de la régulation

- Stabiliser les systèmes instables.
- Augmenter la précision.
- Maitriser la qualité de production

1.3.3. Recommandation d'une régulation

Pour qu'une régulation soit juste, il faut :

- Qu'elle ne mette pas en péril la stabilité de processus, car une instabilité se caractérise par des oscillations excessives.
- Qu'elle assure une bonne précision, c'est-à-dire, l'écart consigne-mesure doit être le plus faible possible
- Qu'elle corrige rapidement l'influence des perturbations et le temps de réponse vis-à-vis consigne-mesure.

1.3.4. Principe de régulation

Dans la plus part des cas la régulation comprend trois maillons indispensables : l'organe de mesure, l'organe de régulation et l'organe de contrôle (figure .**1.7**). 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 agisse sur le processus. Les paramètres qui régissent le processus sont ainsi stabilisés en permanence à des niveaux souhaités.

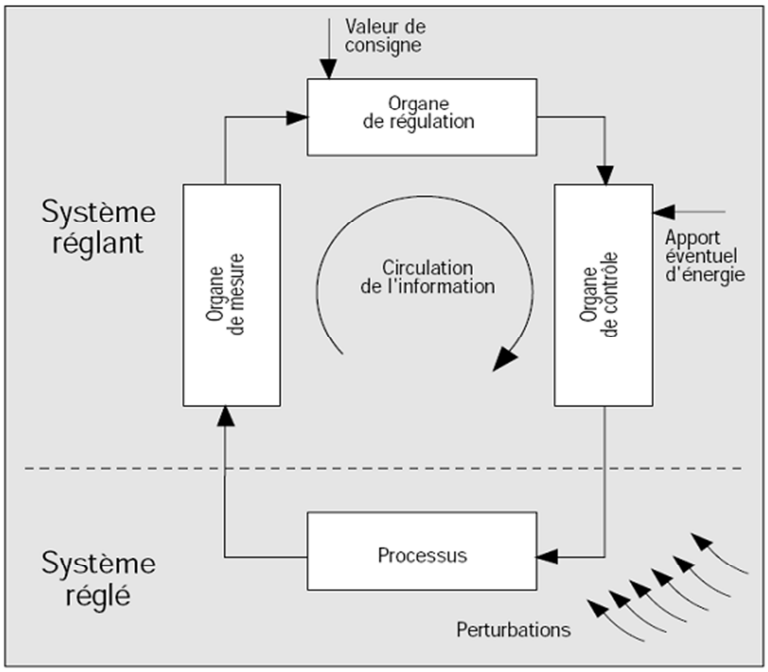


Figure **1.5** : Schéma de principe d'une régulation en boucle fermée

- **Conduite en régulation**

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 à l'aspect asservissement.

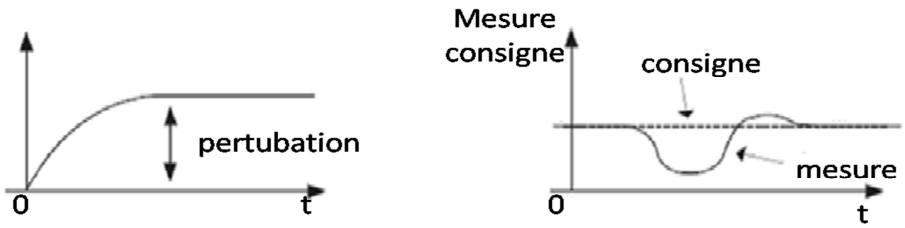


Figure **1.6** : Rejet de perturbation

- **Conduite en poursuite**

L'opérateur 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.

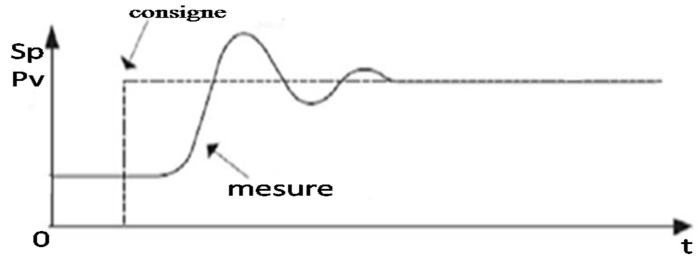


Figure **1.7** : Conduite en asservissement

1.3.5. Formes fondamentales de régulation [5]

1.3.5.1. Régulation en boucle ouverte

En boucle ouverte l'opérateur tiens le contrôle de l'organe de réglage, l'organe de contrôle dans ce type d'asservissement ne réagit pas à travers le processus sur la grandeur mesurée celle-ci n'est pas contrôlée. Une régulation boucle ouverte a besoin absolument de la loi régissant le fonctionnement du processus (veut dire il faut connaitre la corrélation entre la valeur mesurée et la grandeur réglante).

Cette singularité d'asservissement nous permette d'anticiper les phénomènes et d'obtenir un temps de réponse très court. Enfin, l'asservissement en boucle ouverte est la seul solution envisageable lorsqu'il n'y a pas de contrôle final possible.

Par contre, ces inconvénients s'impose sur la connaissance des lois régissant le fonctionnement de processus, il est très souvent qu'on le connait pas cette loi en question

Enfin, la régulation en boucle ouverte ne compense pas les facteurs perturbateurs.

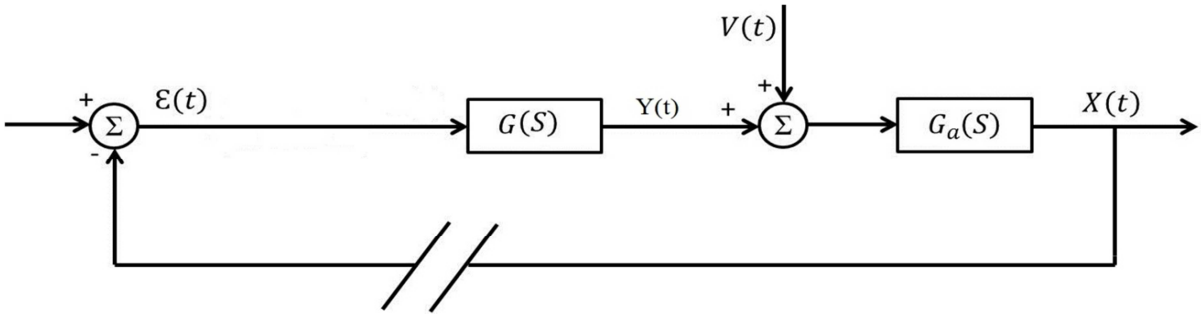


Figure **1.8** : Schéma fonctionnel en boucle ouverte

1.3.5.2. Régulation en boucle fermée

Dans une régulation en boucle fermée, une bonne partie des facteurs perturbateurs sont automatiquement compensés par la contre-réaction à travers le procédé. Autres avantages, il n'est pas nécessaire de connaître 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, en fait un mauvais choix de certains composants peut amener la boucle à entrer en oscillation.

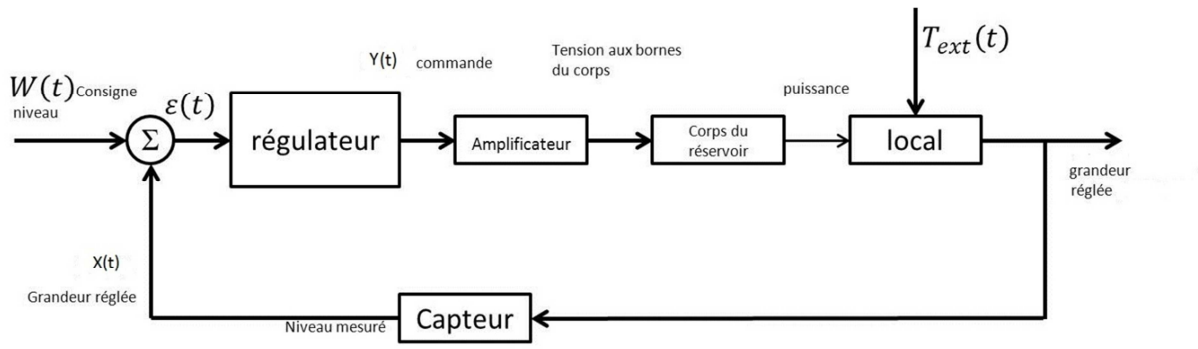


Figure **1.9** : Exemple fonctionnel en boucle fermée

1.3.5.3. Régulation cascade

Le choix d'une régulation en cascade est de minimiser les effets d'une ou de plusieurs grandeurs perturbatrices qui agissent soit sur la variable réglante, soit sur une grandeur intermédiaire se trouvant en amont de la variable à régler. Cette procédure de régulation est susceptible dans des processus à long temps de réponse (régulation de niveau).

En effet, quand une perturbation se manifeste, il est nécessaire d'attendre que son influence se ressente au niveau de l'organe de mesure placé en sortie de chaîne.

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.

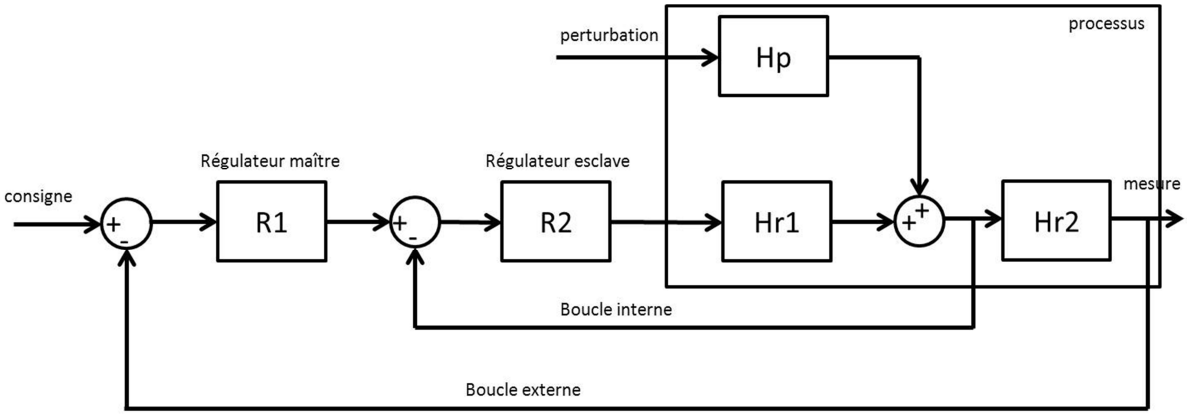


Figure **1.10** Régulation cascade

Hp : Fonction de transfert d'une perturbation

$Hr1$: Fonction de transfert systeme1

$Hr2$: Fonction de transfert systeme2

$M1$: Mesure intermédiaire

R1 : Régulateur de base (maître)

R2 : Régulateur esclave

1.3.6. Régulateur PID

Le régulateur le plus utilisé dans l'industrie est le régulateur PID (Proportionnel Intégral Dérivée), à savoir que ses trois actions permettent de contrôler le processus.

Le régulateur PID basé sur trois piliers principaux :

1. Il fournit un signal de commande vis-à-vis de la mesure x(t) par rapport à la consigne
2. Le terme de dérivateur anticipe la variation de la sortie
3. L'erreur statique $\varepsilon(t)$ éliminée grâce au terme intégrateur

1.3.6.1. Aspect fonctionnel et structurel du régulateur PID

Avec ses trois paramètres P, I et D on a :

a)Action proportionnelle

Ce régulateur est proportionnel au signal de sortie, veut dire à l'écart entre la consigne et la mesure, note $\varepsilon(t)$, l'équation de régulateur proportionnel en fonction de temps est données par :

$$u(t) = K_c \,.\, \varepsilon(t) \qquad (1.16)$$

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

Son inconvénient, si on n'arrive pas à éliminer l'erreur, à cet effet, une valeur trop élevée du gain augmente l'instabilité du système et donne lieu à des oscillations.

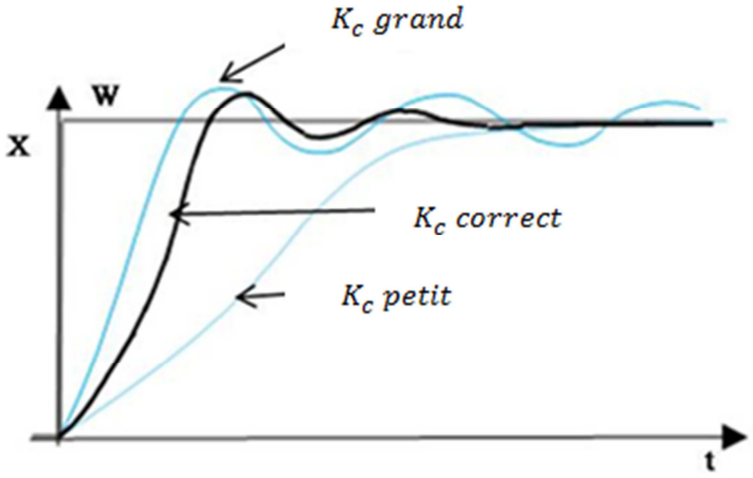


Figure **1.11 :** Influence de l'action proportionnelle

b) 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 élimine (l'erreur statique). L'action intégrale est conditionnée par le temps d'intégrale Ti.

$$U(t) = \frac{1}{T_I} \int_0^t \varepsilon(t)\, dt \qquad (1.17)$$

Sachant que, 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 accordement entre la stabilité et la rapidité.

Par contre, 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.

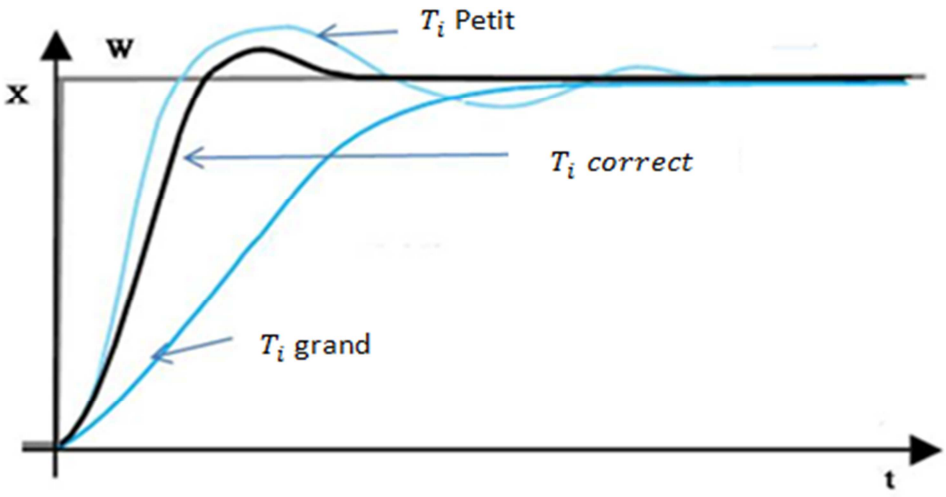


Figure **1 .12 :** Influence de l'action intégrale

c)Action dérivée

C'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 ε(t)

$$u(t) = T_d \frac{d\,\varepsilon(t)}{dt} \qquad (1.18)$$

T_d : C'est la constante de temps dérivée

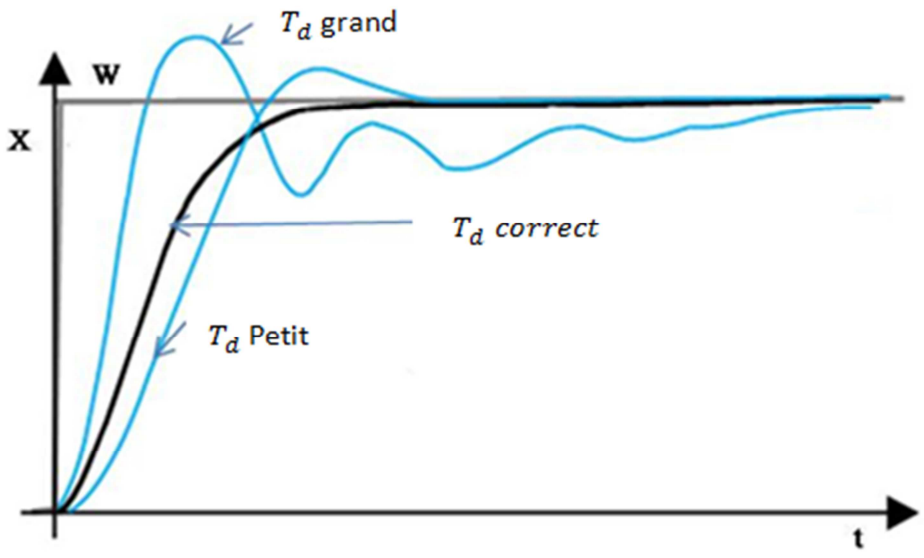


Figure **1 .13 :** Influence de l'action dérivée

Remarque : on pratique il est préférable de limiter cette action pour ne pas amplifier les bruits haute fréquence et de limiter l'amplitude des impulsions dues aux discontinuités de l'écart.

d)Principe de l'action PI [5]

Pour un régulateur intégral pur, le régime dynamique est relativement long. D'un autre côté, 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-à-dire la réaction rapide à un écart de réglage, à l'avantage du régulateur I qui est la compensation exacte de la grandeur pilote.

$$U(t)= K_c \left[\varepsilon(t) + \frac{1}{T_i}\int_0^t \varepsilon(t)dt\right] \quad (1.19)$$

Note:

avant de procéder à n'importe quelle réglage du régulateur , il est nécessaire de connaitre sa structure interne et le montage adéquat pour le processus.

e)Principe de l'action Proportionnel dérivée

En général, le régulateur ne fonctionne pas en action dérivée pure. Mais on lui associe une action proportionnelle.

Son équation est donnée comme suit:

$$U(t)= K_c\left[\varepsilon(t) + T_d \frac{d\varepsilon(t)}{dt}\right] \quad (1.20)$$

f) Principe de l'action proportionnelle intégral Dérivée

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 utilisé en milieu industriel.

Il est donné par l'équation :

$$U(t)= K_c\left[\varepsilon(t) + \frac{1}{T_i}\int_0^t \varepsilon(t)\,dt + T_d \frac{d\varepsilon(t)}{dt}\right] \quad (1.21)$$

D'une autre part la sortie d'un régulateur PID avec filtrage se calcul comme suite :

$$U(t)= K_c[\varepsilon(t) + \frac{1}{T_i}\int_0^t \varepsilon(t)\,dt + \frac{T_d}{N}\frac{d\varepsilon(t)}{dt}] \quad (1.22)$$

NB : le coefficient N correspond au gain du module dérivé filtrée, il est introduit pour que la fonction de transfert soit réalisable.

1.3.6.2. Différentes structures PID [1]

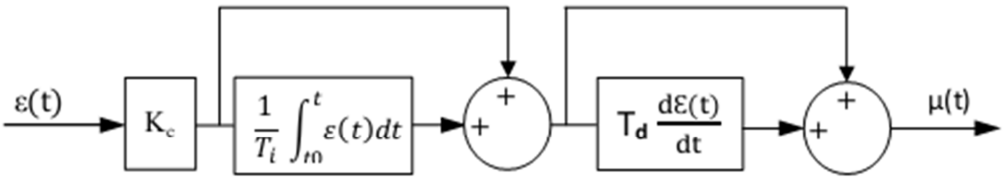


Figure **1.14 :** Structure série du régulateur PID

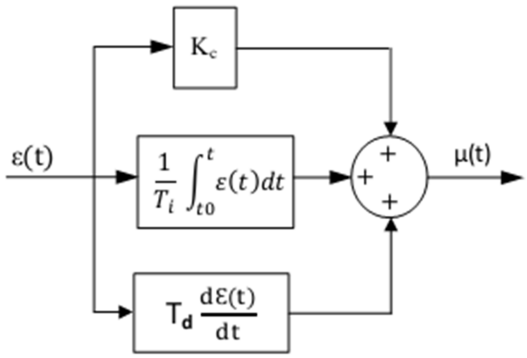


Figure **1.15 :** Structure parallèle du régulateur PID

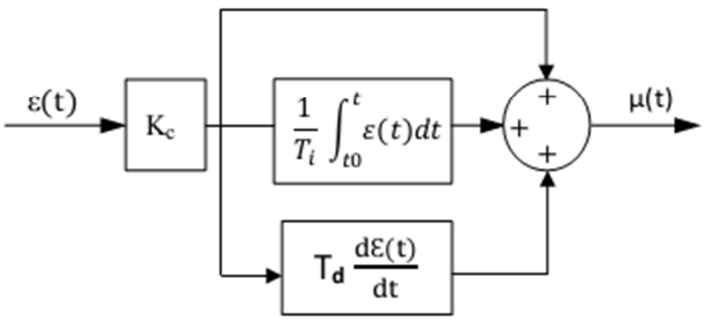


Figure **1.16 :** Structure mixte du régulateur PID

1.3.7. Régulation TOR (Tout Ou Rien)

Comme son nom indique, la régulation TOR se caractérise par son action sur l'organe de réglage qui ne peut être que fonctionner à 100% ou arrêter à 0%. L'action de régulateur peut se présente comme un contact ouvert ou fermé, aussi un signal de 0V ou bien 24V pour commander une électrovanne.

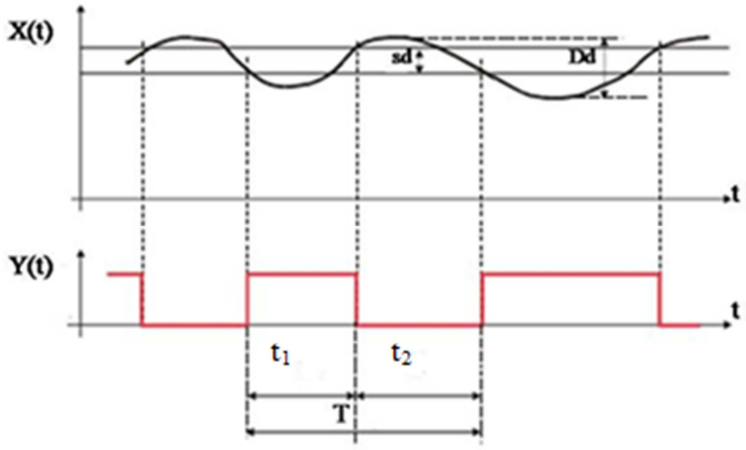


Figure **1.17 :** Régulation TOR

t_1 : Durée d'enclenchement

t_2 : Durée de coupure

T : Durée de cycle

Sd : Différentiel statique

Dd : Différentiel dynamique

1.4. Stabilité [2]

Il ne suffit pas qu'un système soit stable, il faut qu'il soit suffisamment stable. En effet, l'évaluation de la fonction de transfert d'un système n'est pas toujours parfaite (petites constantes de temps ou légers temps morts négligés, hypothèses simplificatrices, incertitude sur les paramètres ou les mesures lors des identifications.

Les deux critères pour vérifier la stabilité : la marge de phase et le gain pour cela un système et stable pour $G_m > 0$ et $\varphi_m > 0$.

Valeur courantes des marges :

8dB< G_m <15dB

40°< φ_m <60°

- **Procédés naturellement stables**

Un procédé est dit naturellement stable si à une variation finie de la grandeur réglante Y correspond à une variation finie de la grandeur réglée X.

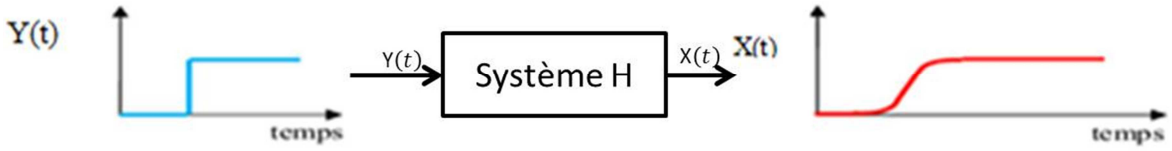


Figure **1.18 :** Système naturellement stable

- **Procédés naturellement instables**

Un procédé est dit naturellement instable si à une variation finie de la grandeur réglante Y, correspond une variation continue de la grandeur réglée X.

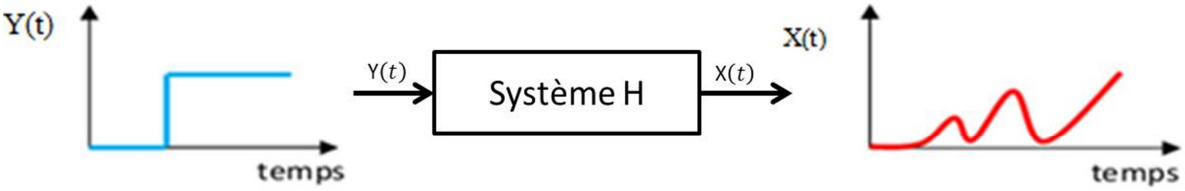


Figure **1.19 :** Système naturellement instable

1.5. Régulation numérique [6]

Le principe de la commande numérique est identique à celui de la commande analogique. La structure d'un système bouclé reste la même dans les deux cas, mais les signaux nécessaires pour le traitement de l'information change, à cet effet on introduit des dispositifs électroniques qui permettent de transformer les signaux analogiques au numérique et vice versa.

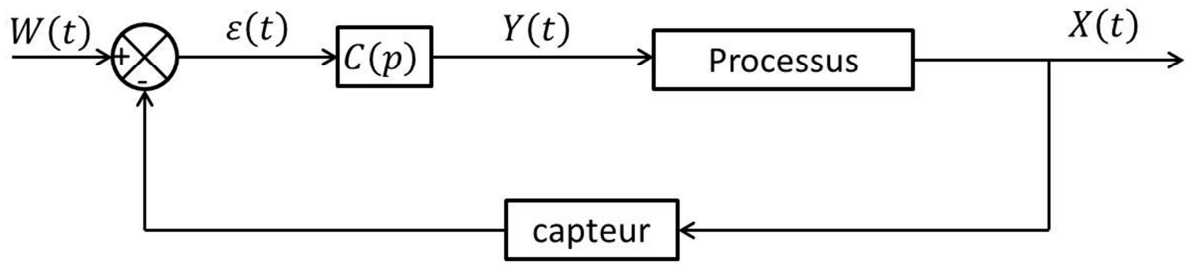


Figure**1.20** : Schéma de régulation d'une chaîne d'asservissement analogique

Dans une chaîne de commande analogique le correcteur C(p) (PID, PI, PD…) produit le signal de commande U(t), cette dernière peut être produite par un ordinateur comme le montre la figure suivante :

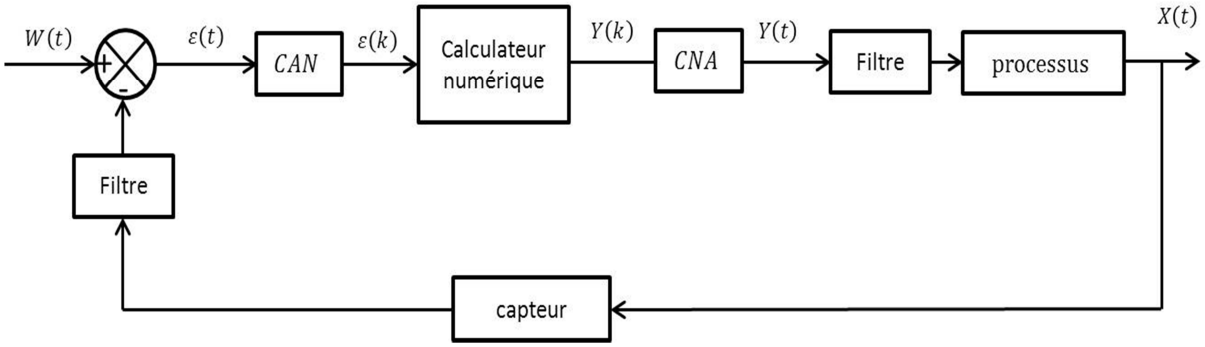


Figure **1.21 :** Schéma d'une correction numérique

A la sortie de l'ordinateur on doit avoir un CNA (convertisseur numérique analogique)

Le correcteur dans ce cas est réalisé sous forme d'un programme dans l'ordinateur .le système est alors vu comme un système numérique de fonction de transfert $H(z) = \frac{X(z)}{W(z)}$

Le correcteur est alors représenté sous forme d'équation de récurrence qui lie Y(k) à ε(k)

1.5.1 Choix du pas d'échantillonnage

La fréquence d'échantillonnage $fe = \frac{1}{\Delta}$ est un paramètre essentiel a régler : elle correspond au nombre d'échantillons du signal que l'ordinateur peut prélever pendant une seconde d'acquisition.

Plus la fréquence d'échantillonnage est élevée et plus celui-ci sera capable de capter des phénomènes rapides. A titre d'exemple, une fréquence d'échantillonnage de 2 G éch. /s pourra prélever un échantillon toutes les 500 ps : c'est la période d'échantillonnage (1/fréquence d'échantillonnage).

Δ ne doit pas être trop petit sinon l'ordinateur consacre inutilement trop de temps aux systèmes qu'il pilote, en le corrigeant par petits coups souvent répétés, et il ne doit pas être trop grand sinon l'ordinateur ne recevra pas certaines informations importantes ce qui nous conduit à élaborer la loi de Shannon.

1 .5.2. Limites théoriques de Shannon. Limites pratiques

Selon le théorème de Shannon, pour restituer fidèlement le signal après numérisation, il faut que la fréquence d'échantillonnage soit au moins égale au double de la fréquence maximale

contenue dans le signal à numériser $f_{ech} > 2f_{max}$. Autrement dit, il faut prélever au moins deux échantillons pendant une période du signal pour pouvoir le restituer fidèlement, ce qui est un minimum en théorie. Mais, en pratique pour satisfaire à la condition de Shannon avec une bonne marge de sécurité on prendra :

$\frac{T_h}{25} < \Delta < \frac{T_h}{5}$ Soit $5f_h < \frac{1}{\Delta} < 25f_h$ (1.23)

1 .5.3.Filtrage des signaux

En toute rigueur chaque fois qu'un signal est échantillonné il faudrait faire précéder le convertisseur numérique d'un filtre analogique anti-repliement éliminant physiquement toutes les composantes du signal. En général la fréquence de coupure du filtre est de l'ordre de f_h

1.5.4. Application aux boucles d'asservissement

Lorsque, après bouclage on obtient une fonction de transfert du 1er ou 2eme ordre il est aisé de définir une plage convenable pour la valeur du pas d'échantillonnage Δ.

b) Système 1er ordre

$$H(p) = \frac{1}{1 + \theta p}$$

$f_h = f_c = \frac{1}{2\,\pi\,\theta}$, donc $T_h = 2\pi\theta \simeq 6.3\,\theta$

En remplaçant T_h dans (1.23) $\frac{6{,}3\theta}{25} < \Delta < \frac{6.3\theta}{5}$

En arrondissant les valeurs on obtient $0{,}25 < \frac{\Delta}{\theta} < 1{,}25$ (1.24)

On prend $\frac{\Delta}{\theta}$ voisin de 1

c) Système du 2eme ordre

$$H(p) = \frac{1}{1+2\,h\frac{p}{\omega_n}+\left(\frac{p}{\omega_n}\right)^2} \quad (1.25)$$

$f_h = f_c = \frac{\omega_n}{2\pi}$, donc $T_h = \frac{2\pi}{\omega_n}$

En remplaçant dans (1.23) $\frac{2\pi}{25\omega_n} < \Delta < \frac{2\pi}{5\omega_n}$

En arrondissant les valeurs, on obtient $0{,}25 < \Delta\omega_n < 1{,}25$ (1.26)

On prend $\Delta\omega_n$ voisin de 1

1.5.5. Programme numérique d'un PID sous l'automate S7-200

Le PID sous l'automate comprend une association mixte (série - parallèle) du gain de l'automate (K_c) et des constantes de temps intégrales (T_i) et dérivées (T_d). Ainsi, la loi de régulation PID utilisée par l'automate a la forme suivante :

$$U(i) = K_C.\varepsilon(i) + \frac{T_s}{T_i}\sum_{j=1}^{i}\varepsilon(j) + \frac{T_d}{T_s}[\varepsilon(i) - \varepsilon(i-1)] \quad (1.27)$$

K_c: Gain proportionnel de l'automate.

T_i : Constante de temps intégrale.

T_d : Constante de temps dérivée.

T_s : Période d'échantillonnage.

$\varepsilon(i) =$ écart ($\varepsilon(i)$ = consigne – mesure).

1.6. Sélection du régulateur

Il se peut que dans certains installation de régulation, d'utiliser qu'une ou deux actions. On sélectionne le type de régulateur en déterminant la valeur des paramètres comme suite :

Si on a besoin d'éliminer l'action intégrale on doit indiquer la valeur infinie pour le temps d'intégration.

Si on a besoin d'éliminer l'action dérivée on doit indiquer la valeur 0 pour le temps de dérivation.

Si on a besoin d'éliminer l'action proportionnelle on doit indiquer la valeur 0 pour le gain KC. Comme Kc est un coefficient dans les équations intégrale et dérivée, mettre le gain 0 entraîne l'utilisation de la valeur 1 comme gain dans le calcul des actions intégrale et dérivée.

1.7 Systèmes de contrôle

Avec des systèmes performants, on trouve toujours trois niveaux :

1. Système de conduite
2. Système de protection
3. Système de sécurité

Le système de conduite (niveau1) comprend essentiellement l'instrumentation de contrôle du processus : capteur, régulateurs, programmateurs, vanne automatique.

Ce niveau assure le bon fonctionnement du processus d'une façon permanente et donne une meilleure communication avec les niveaux en amonts, où par séquence programmées initialisable par le manipulateur.

Le niveau 2 comprend une instrumentation bien fixée d'un niveau bien déterminé.

Ce dernier assure la protection du processus selon une fonction non commandée par l'operateur c'est-à-dire non initialisable sur des paramètres critique du processus.

Le niveau3 c'est la phase finale de protection la plus importante que celle des deux niveaux précédents, il comprend des dispositifs fonctionnant sans énergie auxiliaire (soupapes, disque de rupture, …etc.).

1.8. Conclusion

Dans ce chapitre nous avons vu que, identifier un système dynamique réel revient à caractériser son modèle . Ainsi, une bonne régulation doit assurer avec efficacité la mise au point des boucles de régulations.

Les notions techniques d'identification et de réglage vues dans ce chapitre seront appliquées pour l'étude de la station de pression d'ElettronicaVeneta PUP-4/EV dans le chapitre prochain.

Chapitre 2

Présentation du logiciel LabVIEW

2.1. Introduction

Le langage de programmation LabVIEW (Laboratory Virtual Instrument Engineering Workbench) est un environnement de programmation à caractère universel bien adapté pour la mesure, les tests, l'instrumentation et l'automatisation. C'est un programme dont le but est de contrôler et commander des processus allant du simple capteur ou de l'actionneur, jusqu'à une chaîne de fabrication complète.

Comme nous allons le voir dans la suite, LabVIEW dispose d'un nombre important de fonctions graphiques qui permettent facilement d'acquérir des données, de les traiter et d'afficher les résultats.

2.2. Définition [8]

LabVIEW est un logiciel de programmation en langage graphique qui utilise des icônes au lieu des lignes de texte dans d'autres langages pour créer des applications.

Contrairement aux langages de programmation textuels où se sont les instructions qui déterminent l'ordre d'exécution du programme, LabVIEW utilise la programmation par flux de données : c'est le flux de données transitant par les nœuds sur le diagramme qui détermine l'ordre d'exécution des VIs (Virtual instruments) et des fonctions. On parle d'instruments virtuels car leur apparence et leur fonctionnement sont semblables à ceux d'instruments réels, tels que les oscilloscopes et les multimètres.

Le LabVIEW est un outil d'acquisition, d'analyse et de présentation de données comme il est illustré dans le tableau suivant :

Acquisition	Analyse	Présentation
Contrôle d'instruments -GPIB IEEE 488 -RS 232 -VXI	Traitement numérique -Génération de signaux -Filtrage, fenêtrage -Analyse fréquentielle	Affichage de données -Interface interactive -Graphiques et courbes
Acquisition de données -E/S analogiques -E/S numériques	Traitement statistique -Régression, lissage -Moyenne, écart type	Stockage des données -archivage -impression

Tableau **2.1 :** Fonctionnalités de LabVIEW.

GPIB: general purpose interface bus

VXI : bus informatique industriel destiné à l'instrumentation

2 .3. Constitution d'un VI

Les VIs se composent de trois éléments principaux : face avant, diagramme et icône, connecteur

2.3 .1. Face avant (Front Panel)

La face avant est l'interface utilisateur du programme du VI, elle réceptionne les données d'entrées par l'utilisateur et affiche celles fournies en sortie par le programme. Cette face peut contenir différents types d'instrument virtuel tels que des boutons poussoirs, des interrupteurs, des graphes et d'autre sorte de commande et d'indicateurs.

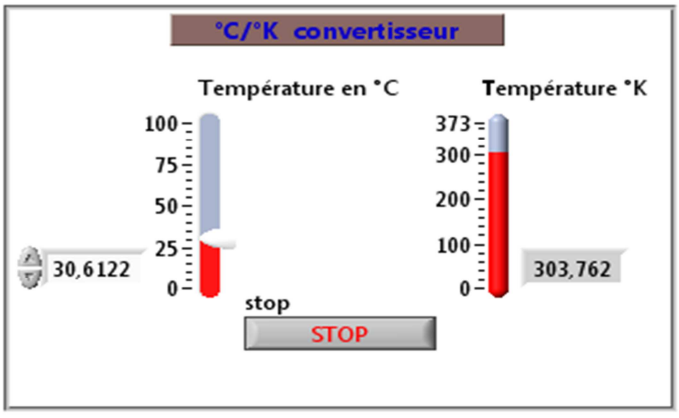


Figure **2.1 :** Exemple de la face avant d'un VI

2.3 .2. Diagramme (Block Diagram)

Le diagramme contient le code source graphique des instruments virtuels. On programme le VI pour contrôler et remplir les fonctions sur les entrées et sorties crées dans la face avant.

Le diagramme peut contenir des fonctions et des structures issues des bibliothèques de VIs intégrées à LabVIEW. Il peut aussi contenir des terminaux associées à des commandes et à des indicateurs crées dans la face avant.

Exemple :

Figure **2.2 :** Diagramme de conversion de température

2.3.3. Icône/connecteurs

- **Icône :** est une représentation graphique d'un VI qui permet de l'identifier au sein d'un autre VI : appeler un sous-programme pour l'utiliser dans le programme principal.
- **Connecteur** : ensemble de terminaux correspondant aux commandes et aux indicateurs du VI qui sont accessibles.

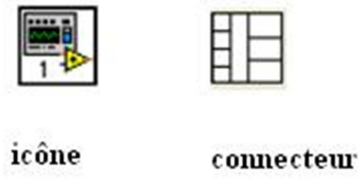


Figure **2.3 :** Exemple d'une icône et un connecteur

2.3.4. Palettes

Pour écrire un programme sur LabVIEW, on a besoin des Palettes qui nous offrent la possibilité de modifier la face avant et le digramme de LabVIEW. On trouve trois types de palettes :

2.3.4.1. Palette d'outil

Elle contient différents éléments qui permettent d'agir sur les objets de la face avant et diagramme comme l'illustre la (figure**2.4)**

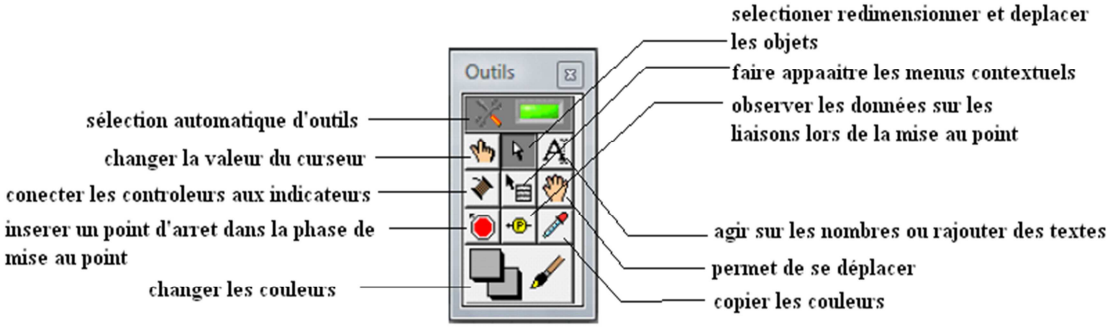


Figure **2.4 :** Palette d'outil

2.3.4.2. Palette de commande

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

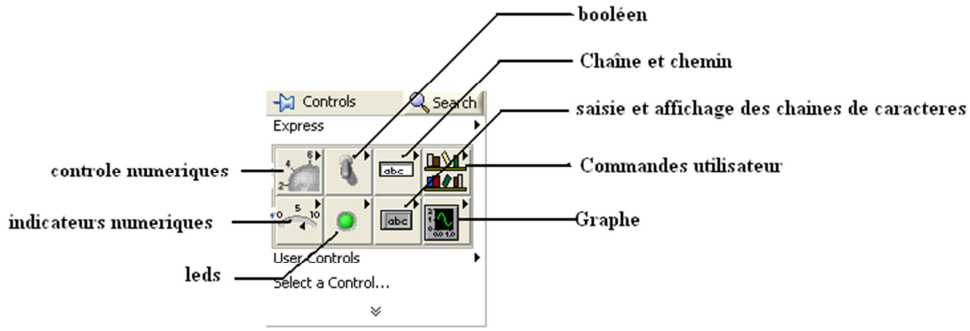


Figure **2.5 :** Exemple de palette de commandes

2.3.4.3. Palette de fonctions

Elle est disponible uniquement sur le diagramme, contient des éléments graphiques nécessaires pour la programmation.

Chaque fonction est représentée par une icône, et les fonctions sous LabVIEW sont regroupées par catégorie : structures, tableau, cluster et, entrées/sorties sur fichier, booléen, chaîne de caractère, comparaison,… etc.

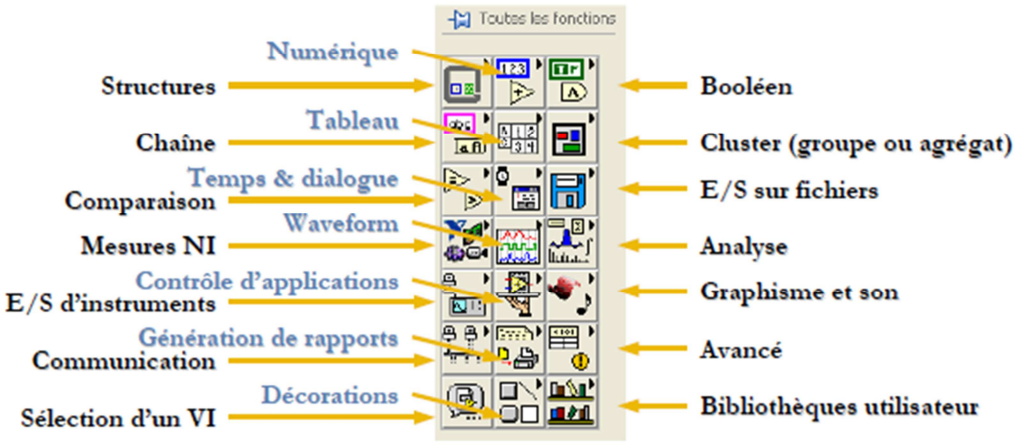


Figure **2.6 :** Palette de fonctions

2.4. Structures de données

2.4.1. Variables

Comme dans la majorité des langages de programmation, LabVIEW possède tous les types de variables existantes (entiers signés, non-signés, flottants, booléens, chaînes de caractères, matrices, et clusters ….).

Afin de les identifier plus facilement dans les diagrammes, chaque type de variables possède une couleur.

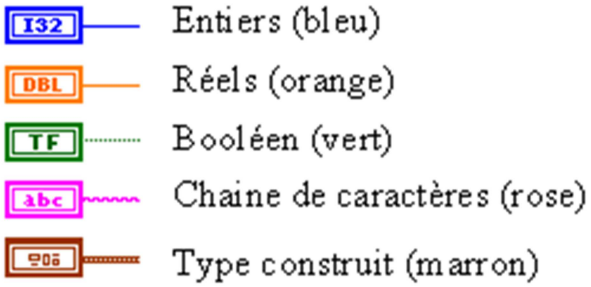


Figure **2.7 :** Variables de LabVIEW

2.4.2. Tableaux sous LabVIEW

Les tableaux est une notion obligatoire dans tous les langages de programmation.

La bibliothèque tableau dans LabVIEW est riche en fonctions prédéfinit tels que l'initialisation, l'indexation, l'inversement, …etc.

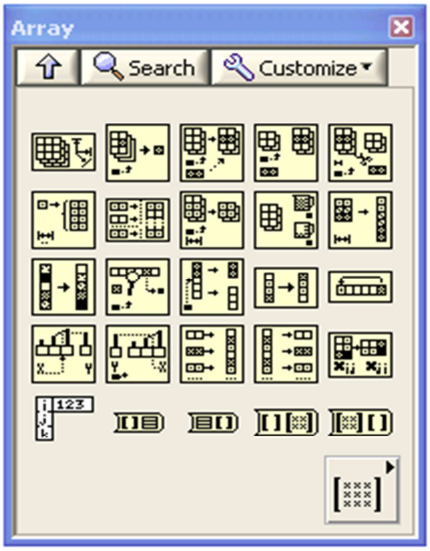


Figure **2.8 :** Bibliothèque tableau dans Labview

2 .4.3. Clusters

Le cluster correspond à un type d'enregistrement (nommé *record* ou *struct* dans d'autres langages de programmation). Il s'agit de grouper dans une seule connexion, différentes données pouvant être de type différentes.

Cela est similaire au câblage de fils dans une même gaine.

Exemple:

LabVIEW utilise 3 valeurs pour caractériser une erreur :

- un booléen pour informer si y'a erreur ou non.
- un entier *code* donnant le numéro d'erreur.
- une chaîne de caractère contenant généralement le nom du VI dans lequel

l'erreur s'est produite.

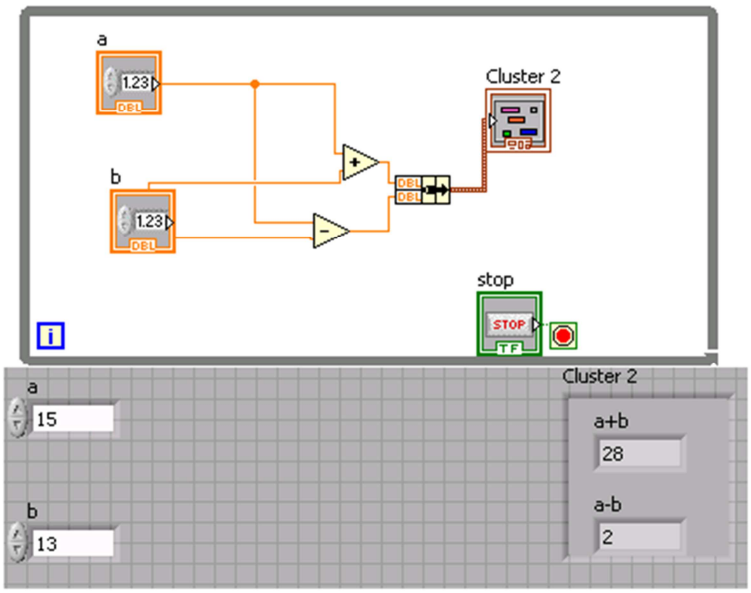


Figure **2.9 :** Exemple d'utilisation du cluster

2.4.4. Waveform

Le type Waveform est un type correspondant à un cluster possédant un champ date (absolue ou relative) t_0, un champ flottant représentant l'intervalle de temps dt séparant chaque valeur, et un tableau de valeurs réelles Y.

Le Waveform est une structure qui facilite l'utilisation des données issues des captures et générations des signaux.

La sous-palette Waveform donne accès aux fonctions suivantes :

- Composantes d'une waveform (x, Δx, [y])
- Construire une waveform
- Opération sur les waveform
- E/S sur fichiers de waveform
- Mesure sur waveform
- Génération de waveform

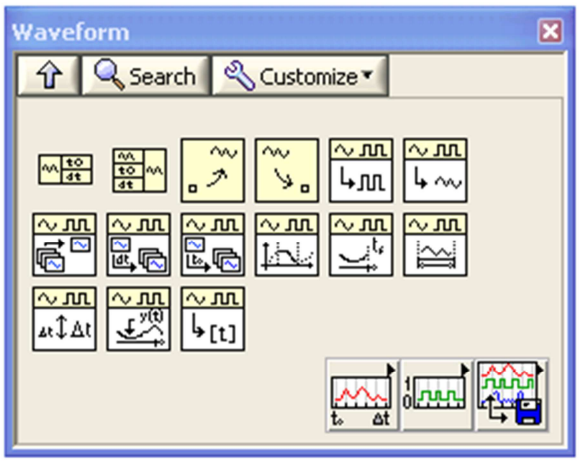


Figure **2.10** : Sous palette waveform

2.5. Structures de programmation dans LabVIEW [7]

Les structures de programmation sont des représentations graphiques de boucles et de conditions dans les langages de programmation textuels, utiliser pour répéter des blocs de code et pour exécuter le code de manière conditionnelle ou dans un ordre spécifique. Ces structures se divisent en trois types, elles permettent d'exprimer tous les algorithmes.

2.5.1 Structure séquence

La structure de séquence est un moyen pour imposer l'ordre d'exécution des taches

Sur le plans graphique, son cadre ressemble à une diapositive, les diapositives sont placées les unes derrières les autres ; l'exécution commence par le code contenu dans la première (n°0) puis continue dans l'ordre 1, 2, …etc.

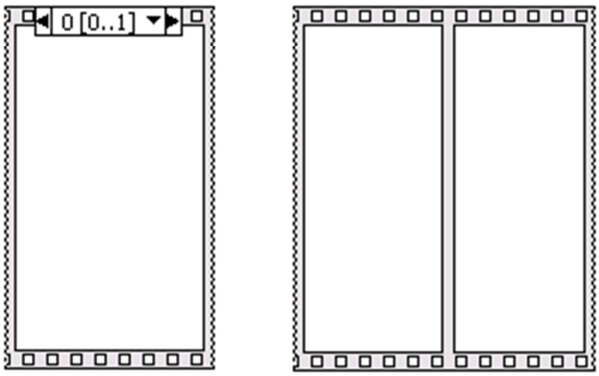


Figure **2.11 :** Structure séquence

2.5.2. Structure Condition

La structure conditionnelle peut correspondre à un *if … then … else* dans les langages textuels. LabVIEW représente l'alternative à l'aide de plusieurs diagrammes alternatifs

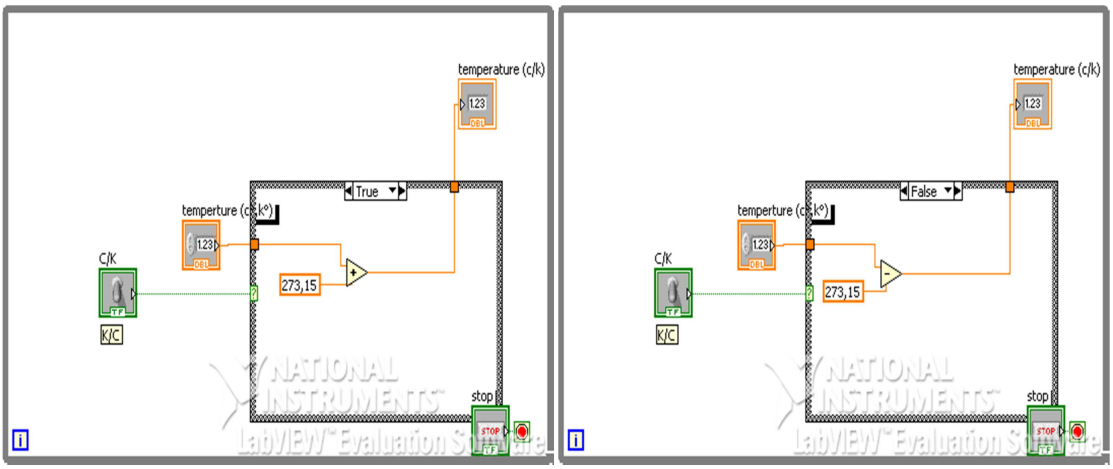


Figure **2.12 :** Exemple de choix (C° /K° ou K°/C°)

2 .5.3 Structures d'itération

- **Boucle for**

La boucle for Possède un compteur d'itération et s'exécute N fois (N paramétrable).

Pour N = 5, i = 0, 1, 2, 3, 4.

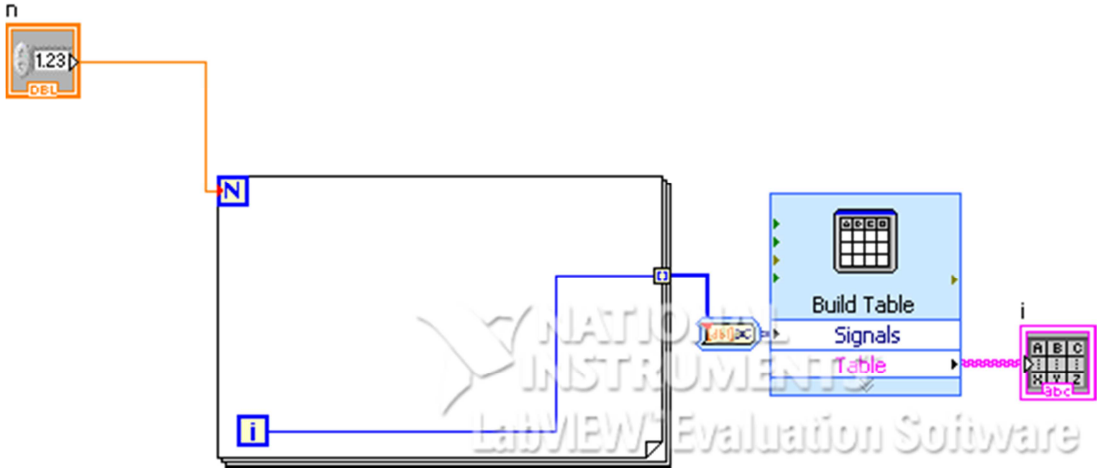


Figure **2.13** : Boucle for

- **La boucle while**

La boucle while (tant que) permet de se répéter N fois (N nombre connu).

A l'intérieur de la boucle while 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 du stop soit égale à 1.

Figure **2.14 :** Exemple d'utilisation de la boucle while

2.6. Traitements de données dans Labview [9]

- **Les fonctions prédéfinies**

Comme tous les autres langages de programmation LabVIEW possèdent des fonctions prédéfinies qui permettent de traiter les différents types de données.

Ainsi, on trouve des fonctions liée aux variables numériques (entiers, réels et complexes), aux variables booléennes, aux chaînes de caractères et aux tableaux. Ajoutant à cela des fonctions liées à la comparaison.

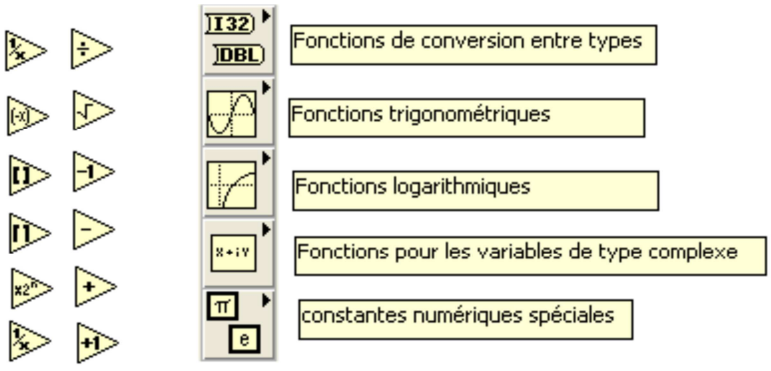


Figure **2.15 :** Quelques fonctions prédéfinies

- **Boîtes à outils mathématique**

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

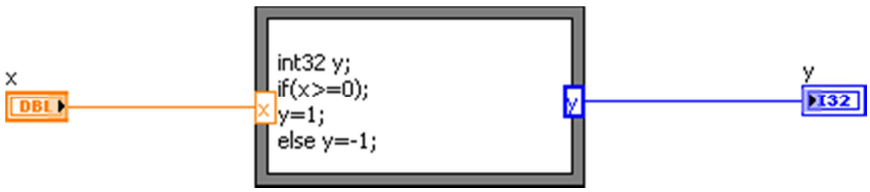


Figure **2.16 :** Exemple de fonction mathématique

- **La bibliothèque de contrôle PID**

Il 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

$$C(P) = \frac{Y(p)}{\varepsilon(p)} = K_P + \frac{1}{PT_i} + p\,T_d \qquad (2.1)$$

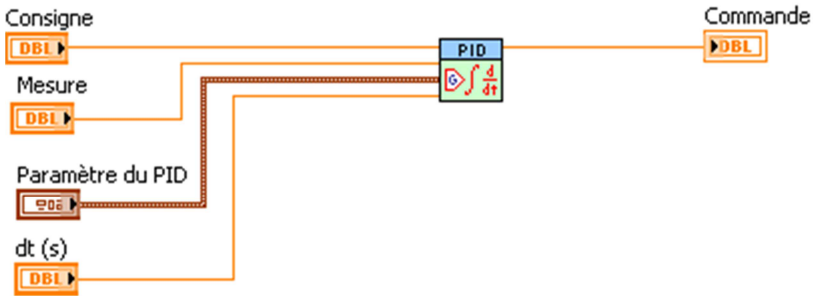


Figure **2.16** : Régulateur PID

2.7. Graphes dans LabVIEW

Les graphes sont accessibles par contrôle /Graphe indicator. Les graphes inscrivent les points en donnant une abscisse initiale et un incrément à chaque nouveau point.

LabVIEW propose 3 principaux types de graphes :

- **Graphe**

Un graphe nécessite un tableau de points ou un waveform par courbe affichée.

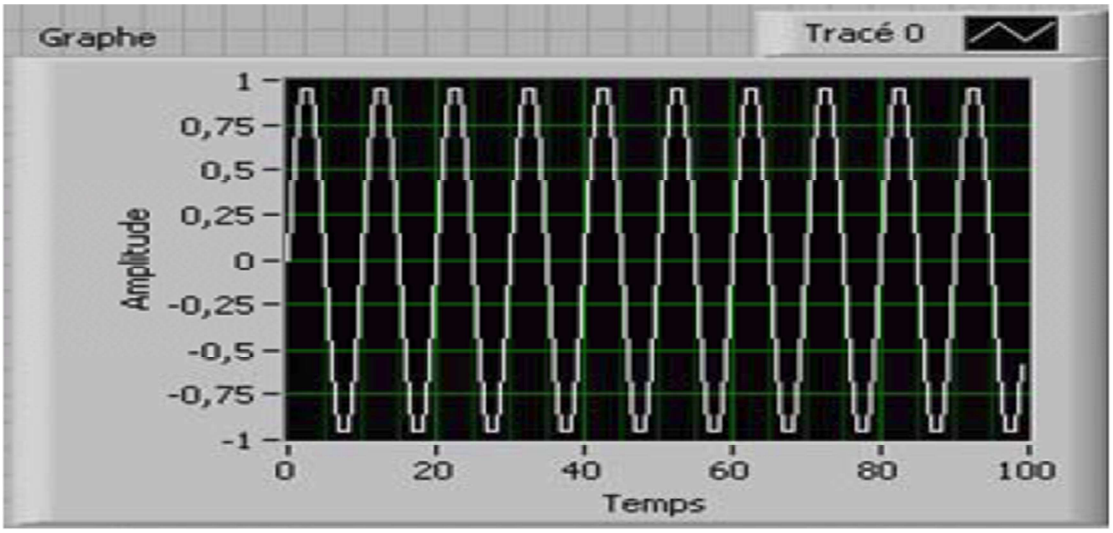


Figure **2.17 :** Graphe

- **Graphe déroulant**

Le graphe déroulant est une sortie 2D représentant une courbe (ou plusieurs courbes) dont les points sont donnés point par point.

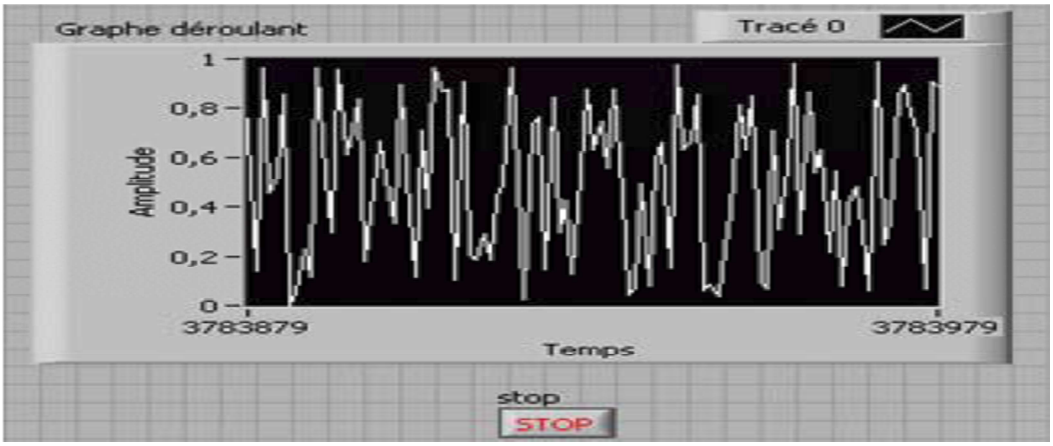


Figure **2.18 :** Graphe déroulant

- **Graphe XY**

Le graphe XY permet de traçer des courbes paramétriques, en annulant la base de temps. Pour ce faire, il faut envoyer un cluster contenant les points pour X, et les points pour Y sur chaque composante du cluster en entrée du graphe

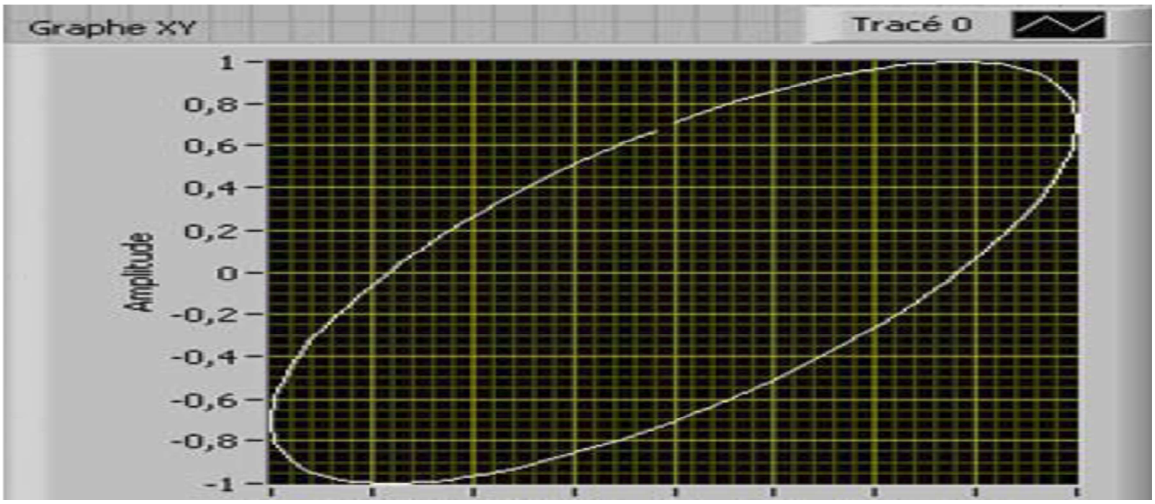


Figure **2.19 :** Graphe XY

2.8 . Acquisition de données

Pour agir efficacement sur un processus physique, chimique ou biologique, naturel ou industriel, il est important au préalable de bien le connaitre. La chaîne d'acquisition fournit au décideur (homme, machine) ces informations permettant d'orienter son action et de valider ses décisions.

La chaîne d'acquisition permet d'extraire des informations simples et nécessaires pour une représentation valable et utile. Ces informations caractérisent des grandeurs physiques telles que : niveau, pression, débit, température,… etc.

La grandeur mesurée est représentée sous forme d'un signal électrique représentatif et exploitable.

2.8.1. Acquisition de données par LabVIEW

La palette d'acquisition de données contient les VIs pour contrôler les cartes DAQ de National Instruments (figure **2.20**). Ces cartes sont souvent multifonctions : conversion analogique-numérique et numérique-analogique, entrée/sortie numérique et compteur/timer. Il est indispensable de bien connaître les fonctionnalités de la carte utilisée pour pouvoir la programmer correctement.

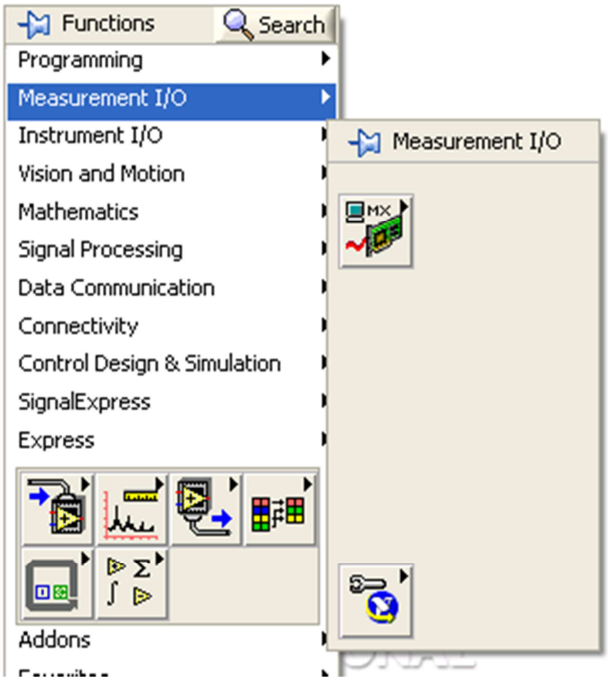


Figure **2.20 :** Palette d'acquisition

2.8.2. Configuration matérielle

National instruments propose un utilitaire d'exploration et de configuration de ses matériels, nommé Measurement & Automation. Lorsqu'il est lancé, il détecte les périphériques qu'il connaît et les place dans la section Périphériques et Interfaces. On peut alors les tester, de façon manuelle, la ressource matérielle et sa connexion au phénomène extérieur.

2.8.3. Acquisition des données DAQmx(Data Acquisition)

Les VIs d'acquisition sont situées dans la palette measurement I/O (figure **2.21**).

L'acquisition de données par DAQmx repose sur la notion de tâche. Une tâche est un paramètre de contrôle : un gain, une mise à l'échelle, un timing, un déclenchement..., tout ce qui caractérise une acquisition. Tous les VIs traitant d'acquisition DAQmx sont polymorphes: le même VI crée une tâche de lecture de tension, de sortie d'impulsion, écriture sur une sortie analogique.

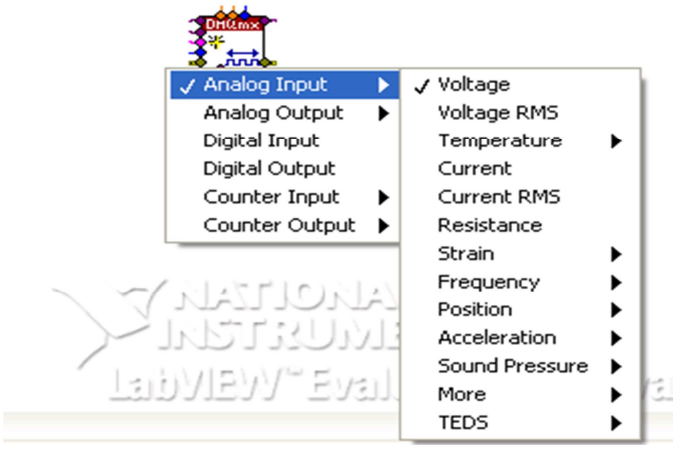


Figure **2 .21 :** Différentes tâches du DAQmx

La configuration des différents VIs est assez complexe, il est préférable d'utiliser un VI Express DAQ assistant pour engendrer automatiquement le code contrôlant l'acquisition.

2 .9. Conclusion

Dans ce chapitre nous avons présenté d'une facon générale le contenu du logiciel LabVIEW qu'on le trouve très vaste en fonctions prédéfinies pour programmer toutes applications voulues .

Ce chapitre nous a permis d'explorer l'environnement graphique de LabVIEW et de connaitres les défferents blocs que nous utiliserons plus tard pour programmer notre plate forme de contrôle de la station de pression.

Chapitre 3

Description matérielle

3.1. Introduction

Quantitativement, la pression est la seconde grandeur physique mesurée industriellement après la température. Du vide poussé de quelques Pascals absolus aux très grandes pressions de plusieurs milliers de bar, les techniques de mesures développées pour les applications sont très variées.

3.2. Description de l'unité

L'unité de pression PUP-4 (figure **3.1**) est constituée d'un réservoir d'air, un compresseur actionné par un moteur électrique qui fournit de l'air servant à atteindre une pression souhaitée et la maintenir.

L'actionneur de l'unité est une vanne proportionnelle à commande électrique située sur le côté de refoulement.

Un transducteur de pression qui fournit le signal de rétroaction, il se trouve sur un côté du réservoir.

L'unité comporte un manomètre pour la lecture de pression, elle comporte aussi une vanne manuelle (perturbatrice). Enfin une valve de pression maximale (soupape de sécurité) située en série sur le coté de refoulement qui sert à éviter les pressions dangereuses à l'intérieur du réservoir et le blocage du compresseur.

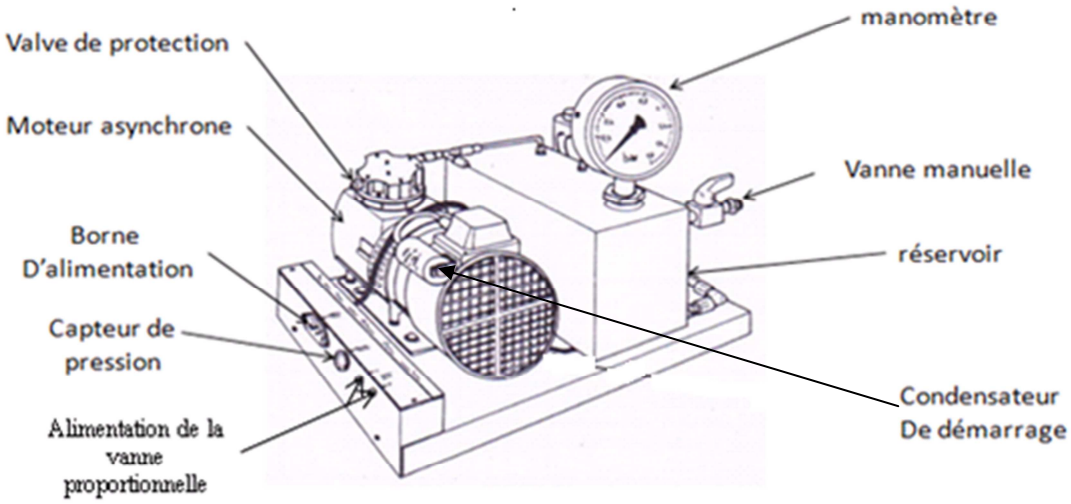


Figure **3.1 :** Processus de pression PUP-4

3.3. Contrôle automatique de la vanne proportionnelle

Le contrôle automatique de pression est très répondue en industrie, sa structure de base est identique à celle des autres automatismes, sauf que la vanne proportionnelle qui est l'actionneur à contrôler nécessite un amplificateur de puissance.

On peut appliquer les différents régulateurs qu'on connaît déjà :

- On/Off, régulation Tout Ou Rien.
- Proportionnelle.
- Proportionnelle et intégrale.
- Proportionnelle, intégrale et dérivée.

3.4. Transducteur de pression (capteur de pression)

En physique on définit la pression comme le rapport (Force /Surface), dans le système international l'unité de mesure de la pression est le pascal (Pa) qui correspond à $1N/m^2$.

En général dans l'industrie, on trouve d'autres unités de pression : Bar, PSI, ……..

3.4.1. Méthodes de mesure de pression

- **pression absolue :** Pression mesurée par rapport au vide parfait.
- **Pression relative :** Pression mesurée par rapport à la pression atmosphérique.
- **Pression différentielle :** La différence de pression mesurée entre deux sources de pression.

3.4.2. Transducteur de pression semi-conducteur

Le principe de base de ce type de transducteur est la piézorésistivité d'un support de silicium : Propriété qui permet de changer sa résistance quand il se déforme.

Dans un diaphragme au silicium on a 4 résistances reliées à un pont de Wheatstone (figure **3.2.**), ce diaphragme est soudé à un anneau en verre servant de support.

Le pont est alimenté, sur une diagonale, par une tension constante, l'autre diagonale fournit une tension variable proportionnelle à la pression agissant sur ce diaphragme.

Le transducteur employé est du type différentiel, si on laisse libre une entrée on obtient un transducteur relatif à la pression atmosphérique.

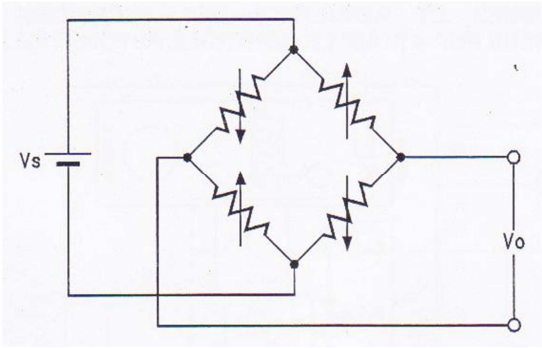


Figure **3.2 :** Pont de Wheatstone

3.4.3. Exemple de calcule avec le pont de Wheatstone [2]

On désire mesurer une température variable entre 0° et 100° à l'aide d'une sonde a résistance reliée à un pont de Wheatstone alimenté par un courant constant comme le montre la (figure **3.3)**

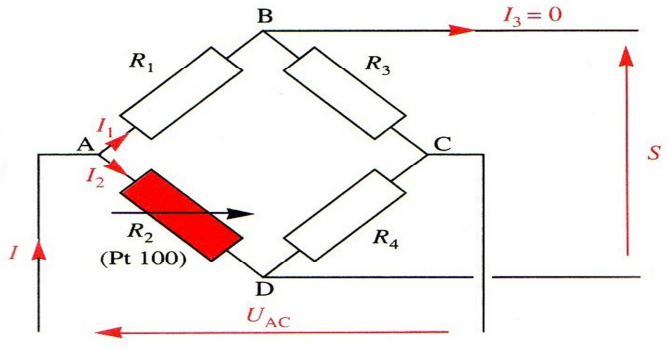


Figure **3.3 :** Pont de Wheatstone

La résistance de la sonde est donnée par la formule suivante :

$R_\theta = R_0(1 + 3{,}91.10^{-3}\,\theta - 6.10^{-7}\theta^2)$ (3.1)

En désignant par :

R_θ : Valeur de la résistance R_2 de la sonde à θ°C.

R_0 : La valeur de la résistance à0°C, $R_0 = 100\Omega$.

A l'aide du pont de Wheatstone, on réalise un transmetteur délivrant un signal de 0 mV à 50 mV lorsque la température varie entre 0°et 100°.

- On commence par déterminer la valeur commune des résistances R_1, R_2, R_3, R_4 :

À θ=0°C, S doit être égale à 0 mV et $I_3 = 0A$.

En équilibre si : $R_1.R_4 = R_2.R_3$, (R_2=100 Ω à 0°)

Donc $R_1 = R_3 = R_4 = 100\Omega$.

- Ensuite, on détermine l'intensité du courant d'alimentation I :

S = U_{BC}+U_{CD}=U_{BC}-U_{DC} =R_3I_1-R_4I_2 (3.2)

U_{AC}=$(R_1+R_3)\,I_1$=$(R_2 + R_4)I_2$ donc I_1=$\frac{I_2(R_2+R_4)}{R_1+R_3}$ (3.3)

D'après la loi des nœuds appliquée au point A : $I_2 = I - I_1$

- On remplace dans (3.3) : I_1=$\frac{(I-I_1)(R_2+R_4)}{R_1+R_3} = \frac{I(R_2+R_4)}{R_1+R_2+R_3+R_4}$ (3.4)

La relation (3.3) permet aussi d'obtenir $I_2 = \frac{I(R_1+R_3)}{R_1+R_2+R_3+R_4}$ (3.5)

- On remplace les relations (3.4) et (3.5) dans (3.2) on obtient :

S=$R_3\frac{I(R_2+R_4)}{R_1+R_2+R_3+R_4}$ -$R_4\frac{I(R_1+R_3)}{R_1+R_2+R_3+R_4} = \frac{R_3(R_2+R_3)-R_4(R_1+R_3)}{R_1+R_2+R_3+R_4}I$ (3.6)

Donc, $I = \frac{R_1+R_2+R_3+R_4}{R_3(R_2+R_3)-R_4(R_1+R_3)}S$

Pour θ=100°C, S=50mV

- On calcule R_2 en remplaçant θ dans la relation (3.1)

R_2=138,5Ω, $R_1 = R_3 = R_4$=100Ω

Application numérique : $I = 5{,}7mA$

- Pour une valeur de θ=50°C on calcule R_2 avec la relation (3.1) et on remplace dans la relation (3.6).

R_2 =119,4Ω et S=26,3 mV

3.4.4. Montage à 2 fils

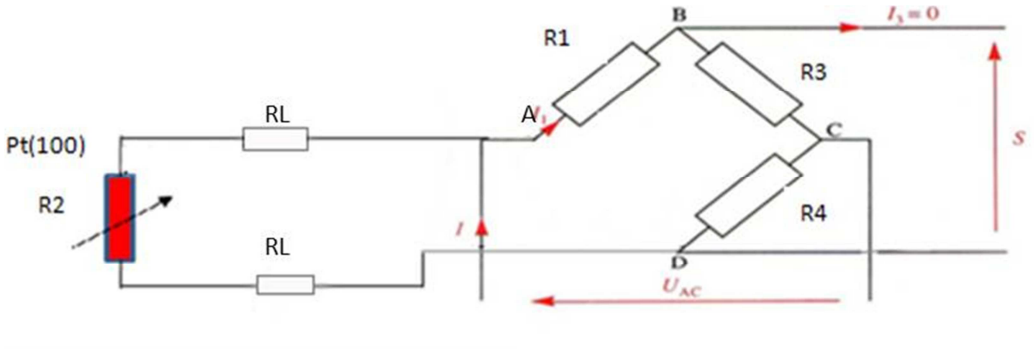


Figure **3 .4 :** Montage à deux fils

La relation de la sortie S est définie comme suit :

$$S = I\frac{R_3(R_2+R_4)-R_4(R_1+R_3)}{R_1+R_2+R_3+R_4} \quad (3.7)$$

En tenant compte des conditions d'installation, la résistance dans la branche AD devient égale à $R_2 + 2R_L$ et $R_1 = R_3 = R_4 = R = 100\Omega$.

$$S = 100I\frac{R_2-100+2R_L}{300+R_2+2R_L} \quad (3.8)$$

3.4.5: Montage 3 fils

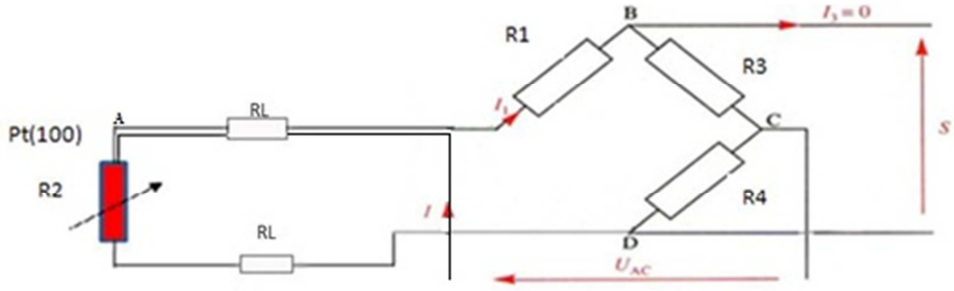


Figure **3.5 :** Montage à trois fils

On note que le 3^{eme} fil permet de déplacer le sommet A et ainsi de repartir les résistances de ligne entre les branches AB et AD.

Ces deux branches présentent des résistances égales: $R_1 + R_L$ et $R_2 + R_L$.

La nouvelle expression de S est :

$$\mathrm{S} = 100\,\mathrm{I}\,\frac{R_2 - 100}{300 + R_2 + 2R_L}$$

Le montage 3 fils est employé au niveau industriel, car dans le montage a 2 fils le terme $2R_L$ au numérateur devient non négligeable en particulier pour les températures proches de 0°C où ce terme est faible.

3.4.6. Transducteur de pression de la station PUP-4

Le transducteur est fabriqué par la société Micro-Switch Honey Well

Les caractéristiques les plus importantes de ce transducteur sont :

Champ de mesure	0 à 30 PSI (0 à 2 bars)
F.S.O	195 Mv
Linéarité	0,25% FSO
Temps de réponse	1 ms

Tableau **3.1** : caractéristique du transducteur

La (figure **3.4**) montre en détail les différents composants du transducteur utilisé dans la station

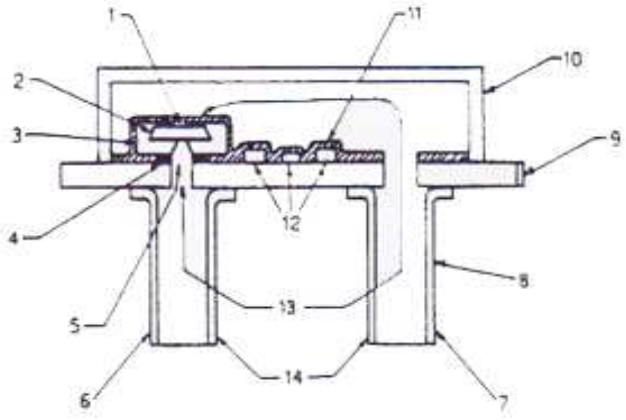


Figure **3.6** : Transducteur RC/EV

1 Elément capteur

2 Cavité du capteur

3 Cristal du silicium

4 Point de relâchement

5 Entrée alternative (transducteur absolu ou transducteur différentiel)

6,7 Deux orifices

8 Placage de laiton (alliage de cuivre et de zinc)

9 Substrat de céramique

10 Revêtement de céramique

11 Couche de pyralène

12 Composant du circuit

13 Pression

14 Orifice double (différentielle)

3.5. Conditionnement du signal du capteur

Pour faire l'acquisition du signal de sortie du transducteur, il faut d'abord le conditionner pour le rendre lisible par les cartes d'acquisitions. La plage de conditionnement disponible est (0~ 10V). La (figure **3.7**) montre le schéma électronique du conditionneur :

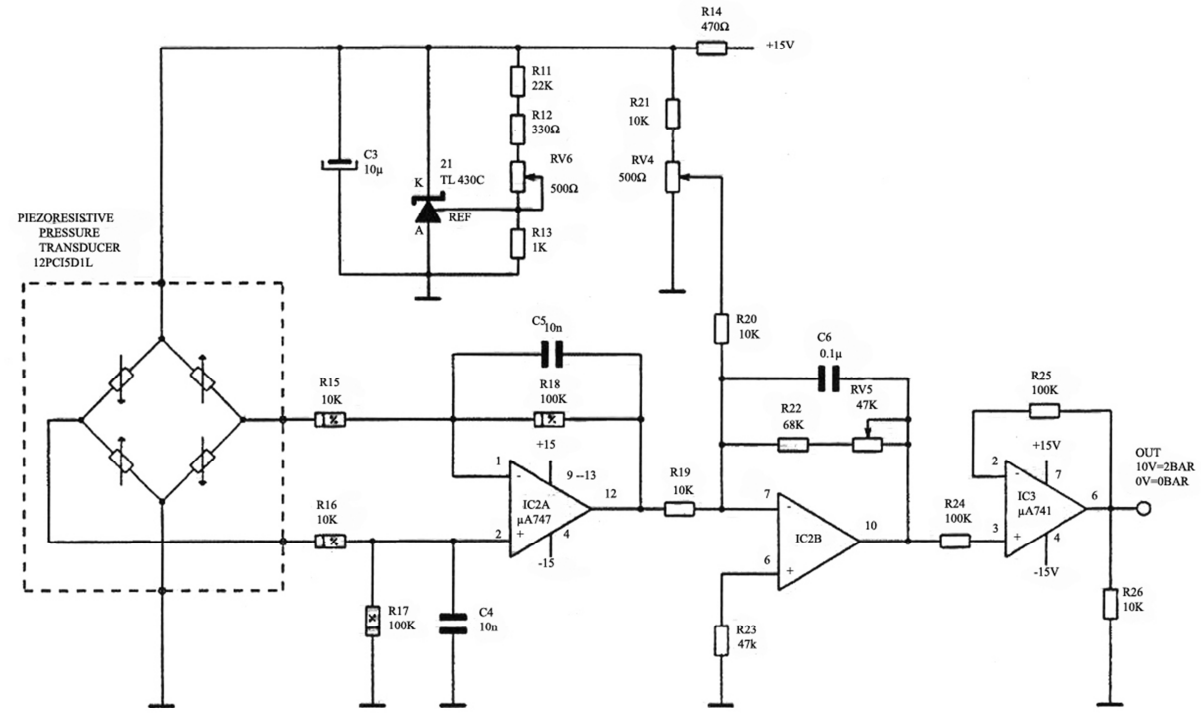


Figure **3.7 :** Schéma électronique du conditionneur

3.6. Amplificateur de puissance de la vanne proportionnelle

L'organe réglant (vanne proportionnelle) est l'élément qui assure la régulation de pression de la station. Cette vanne est alimentée par(0~ 24V) .

Pour fournir cette tension il faut réaliser un amplificateur de puissance comme le montre la (figure **3.8**) (0~ 10V), sortie (0~ 24V) .

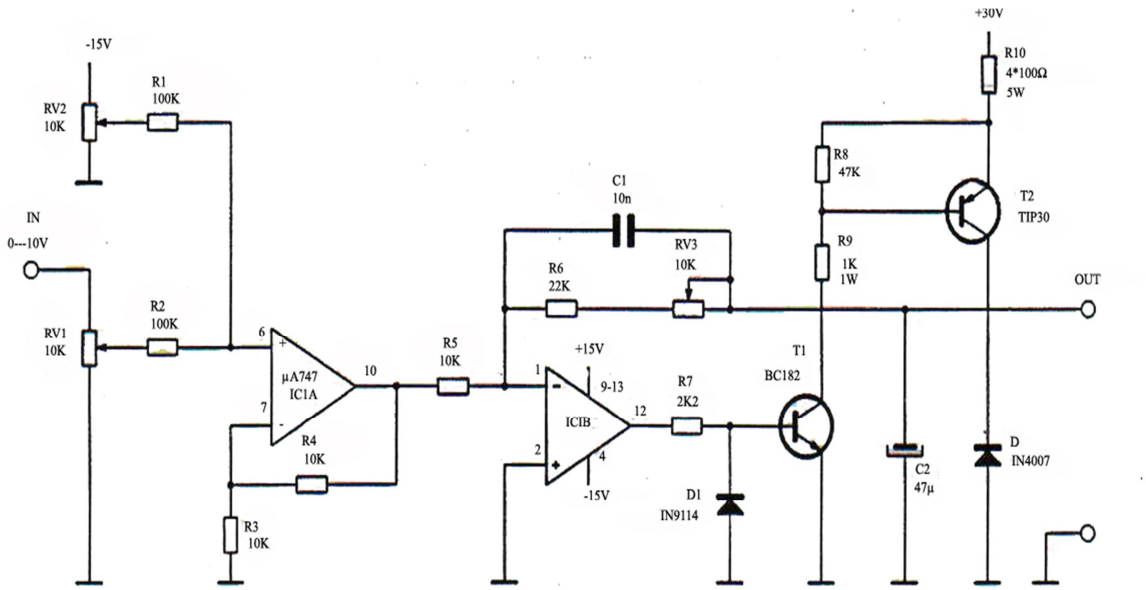


Figure **3.8 :** Schéma électronique de l'amplificateur de puissance

L'amplificateur et le conditionneur sont rassemblés sous un seul boîtier comme le montre la (figure **3.7**).

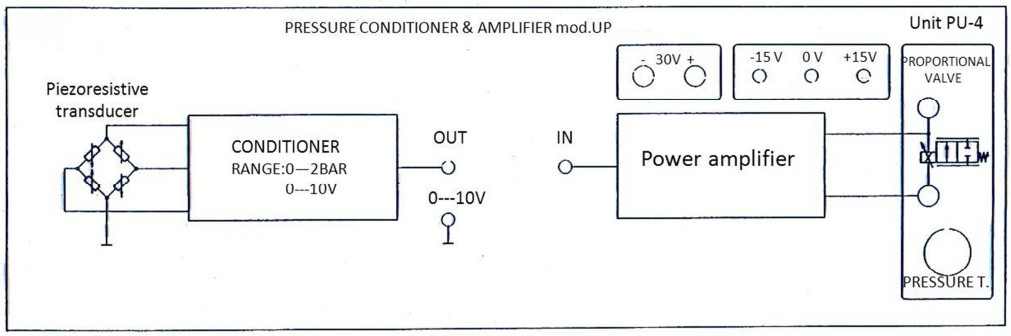


Figure **3.9** : Bloc conditionneur et amplificateur de puissance

3.7. Carte d'acquisition Labjack

Pour commander notre système avec un ordinateur, on fait appel aux cartes d'acquisitions qui assurent la communication entre l'ordinateur et la station PUP-4.

La carte "Labjack U3-LV" est un dispositif d'acquisition de données à connexion USB. Doté de 16 entrées/sorties. Il se prêtera à de multiples applications. La (figure **3.8**) montre un Labjack U3_LV (low voltage).

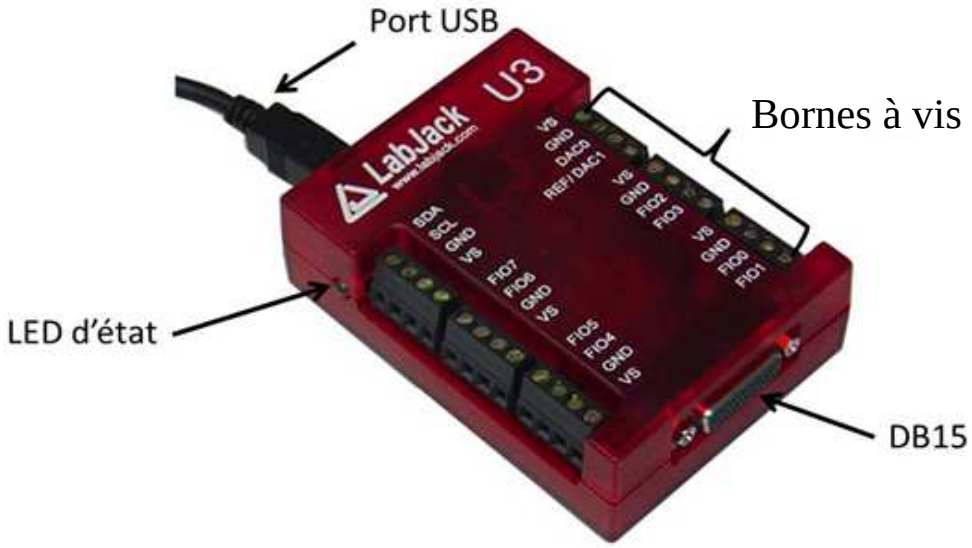


Figure **3.10** : Carte d'acquisition Labjack

3.7.1. Description de la carte Labjack U3_LV

La carte se compose :

- **Port USB(universel serial bus)** : qui assure l'alimentation et la communication.
- **LED d'état** : Led verte située sur le bord gauche de l'appareil, elle clignote lors de la réinitialisation de l'appareil puis reste allumée d'une façon permanente.
- **GND et SGND (Ground) :** Les bornes GND et SGND disponible sur les bornes à vis et le DB15 offre une masse commune qui est reliée à la masse de l'ordinateur.
- **VS (voltage supply)** : les bornes de VS sont conçues comme des sorties d'alimentation d'une valeur de 5V (tension fournie par le câble USB).
- **FIO (flexible input/output), EIO** : Les ports FIO et EIO du Labjack peuvent être configurés individuellement comme entrée numérique, sortie numérique ou une entrée analogique.

Les 8 premières lignes (FIO_0 ~ FIO_7) sont disponibles sur les bornes à vis, les autres lignes (EIO_0 ~ EIO_7) sont disponibles sur le DB15.

- **AIN (analog input)**: On peut configurer sur le Labjack jusqu'à 16 entrées analogiques (FIO_0 ~ FIO_7) et (EIO_0 ~ EIO_7). La plage de variation du signal d'entrée analogique est de (0 ~ 2.44V) avec une résolution de 12 bits soit une précision de 0.58mV.
- **DAC (data acquisition converter) :** Le Labjack U3 dispose de 2 sorties analogiques (DAC0 et DAC1) qui sont disponibles sur les bornes à vis. Chaque sortie analogique peut délivrer une tension comprise entre (0 ~ 5V) volts avec 10 bits de résolution soit une précision de 4.88 mV.

- **Digital I / O :** on peut configurer jusqu'à 20 canaux d'entrées/sorties numériques. Si une borne est configurée en sortie il délivre soit 0V comme 0 logique, ou 3.3V comme 1 logique.
- **DB15 (data bus)** : le connecteur DB 15 fait sortir 12 entrée/sortie numériques on peut l'utiliser comme un bus d'extension ou les 8 EIO sont des lignes de données et le 4 CIO sont les lignes de commandes.

3.7.2. Descriptions des différents blocs de programmation du Labjack

- **List all :** Renvoie tous les périphériques connectés d'un DeviceType et Connection Type données.

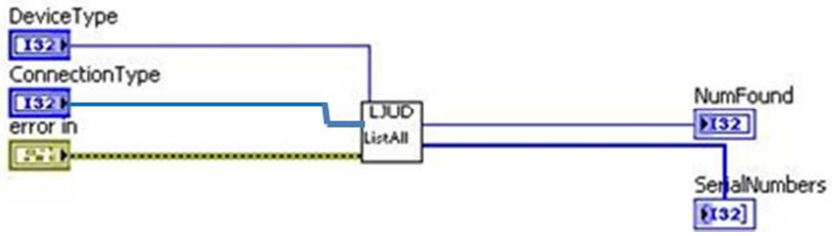


Figure **3.11 :** Bloc listAll

Device Type : Le type de Labjack à rechercher.

Cennection Type : Le type de connexion utilisé des Labjack à rechercher.

NumFound : Le nombre de Labjack trouvé.

SerialNumbers : Table des numéros de séries des Labjack trouvés.

Error In : Renvoi le code de l'erreur produite si il y'a eu lieu, 0 sinon.

- **LJUD OPENS** : Utilisé pour appeler un Labjack, cependant, si ce dernier est ouvert il reste jusqu'à ce que l'application se termine.

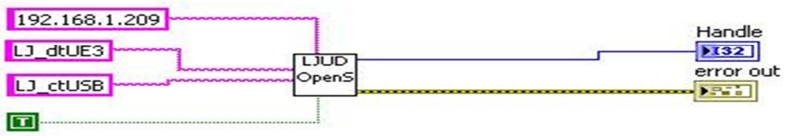


Figure **3.12 :** Bloc opens

Device Type et Connection Type : chaînes de caractère pour introduire le type de Labjack et le type de connexion utilisé (USB uniquement pour le U3).

Adress : Adresse IP si on utilise Ethernet ou le numéro de série du Labjack.

First Found : Commande boolienne si elle est égale à 1 Adress et Connection type sont ignorés, et le premier Labjack connecté s'ouvre.

Handle : Code renvoyé par l'Opens et transmit a d'autre bloc pour identifier le Labjack ouvert.

Error Out : Renvoi le code de l'erreur produite si il y'a eu lieu, 0 sinon .

- **LJUD EGETS :** Utiliser pour configurer l'un des bornes du Labjack soit en entrée analogique, entrée numérique ou sortie numérique.

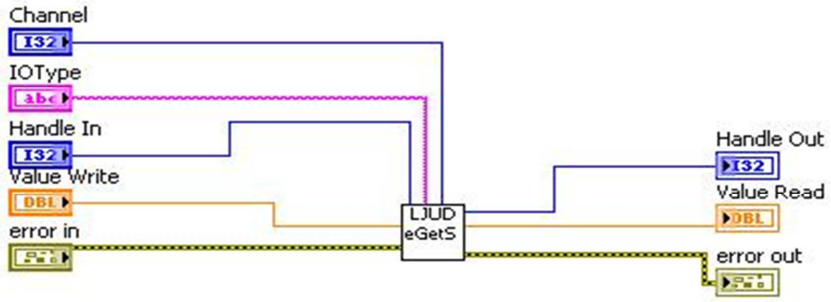


Figure **3.13** : Bloc EGets

Chanel : Le numéro de la borne à configurer (FIO_0 ~ FIO_7)

IOType : Le type de commande

Type de demande	Commande
Entrée analogique	LJ_ioGET_AIN
Sortie analogique	LJ_ioPUT_DAC
Entrée numérique	LJ_ioGET_DIGITAL_BIT
Sortie numérique	LJ_ioPUT_DIGITAL_BIT

Tableau **3.2 :** Liste des commandes

Pour les sortie, analogique, on peut configurer que les deux bornes DAC0 et DAC 1.

Handle In et handle Out : handle in, reçoit le code approprié du Labjack envoyé par le bloc LJUD opens. handle out, renvoie ce code pour d'autre blocs.

Value Write, value Read : Si on configure la borne en entrée, on utilise Read qui nous permet d'afficher la valeur de la mesure, sinon on utilise Write pour faire sortir une valeur désirée.

Error In, error Out : Error in reçoit les erreurs des précédents blocs. Error out renvoie tous les erreurs.

- **Add request** : Ajouter un élément à la liste des commandes à effectuer.
- **Go one :** Après avoir utilisé add request pour faire une liste de commande, on appel Go one pour les exécuter.
- **Get result :** Affiche les résultats des requêtes exécutées par GO one**.**
- **Erreur to String :** Convertit l'erreur code en une chaine de caractère
- **E AIN, E DAC, E DI, E DO :** Des fonctions simples, au lieu d'utiliser le bloc E Gets avec une commande spécifiée, ces blocs remplacent les différentes commandes comme l'illustre le tableau ci-dessous :

Bloc	Fonction
E AIN	Entrée analogique
E DAC	Sortie analogique
E DI	Entrée numérique
E DO	Sortie numérique

Tableau **3.3 :** Blocs des commandes simples

Ces blocs nous servira pour programmer une entrée analogique pour récupérer le signal de mesure, une sortie analogique pour envoyer le signal de commande, et une sortie numérique pour agir sur une vanne tout ou rien.

On rappelle que le signal d'entrée du Labjack à une plage de variation de (0V ~ 2.44V) et le signal de sortie est de (0V ~ 5V), mais, la plage de conditionnement du capteur de pression de la station est de (0V ~ 10V) et la tension utile de la vanne proportionnelle est de (0V ~ 24V). Donc, pour adapter ses signaux il faut les conditionner, pour cela on réalise 2 circuits d'adaptations à base des amplificateurs opérationnels.

3.8. Amplificateur opérationnel [10]

L'amplificateur opérationnel est un circuit intégré, utilisé pour réaliser des opérations d'amplification, addition, soustraction, intégrale, dérivée et de comparaison.

Un amplificateur opérationnel comporte essentiellement deux entrées e+ et e- et une sortie Vs ainsi que deux bornes d'alimentation V+ et V- . La figure **(3.14)** montre un amplificateur opérationnel.

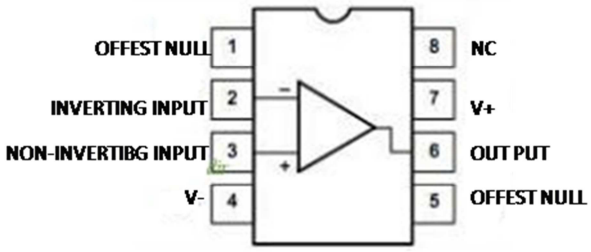


Figure **3.14** : Amplificateur opérationnel

3.8.1. Circuit (0V ~ 10V)/ (0V ~ 2.44V)

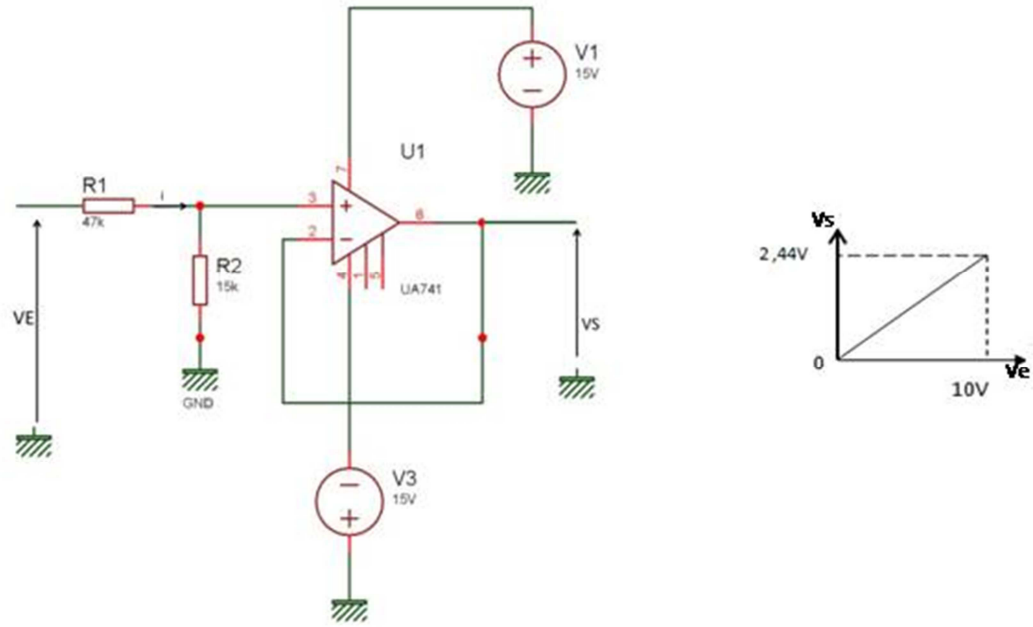


Figure **3.15 :** Montage (0V~ 10V) / (0V~ 2,44V)

$$Vs - R_2 i = 0, \text{ donc } i = \frac{VS}{R_2} \tag{3.9}$$

$$Ve - R_1 i - R_2 i = 0 \tag{3.10}$$

$$Ve - R_1 \frac{Vs}{R_2} - R_2 \frac{Vs}{R_2} = 0$$

$$Ve = \frac{R_1 + R_2}{R_2} Vs$$

$$\frac{Vs}{Ve} = \frac{R_2}{R_1 + R_2} \tag{3.11}$$

Dans notre cas on a Ve_{max}=10V et Vs_{max}=2.44V donc, on doit avoir un gain qui est égale a $\frac{2.44}{10} = 0.24$

$\frac{R_2}{R_1+R_2} = 0.24$. Si on choisit $R_1 = 47K\Omega$ et $R_2 = 15K\Omega$, on trouve $\frac{15}{15+47} = 0.24$

3.8.2. Circuit (0V~ 5V) / (0V~ 10V)

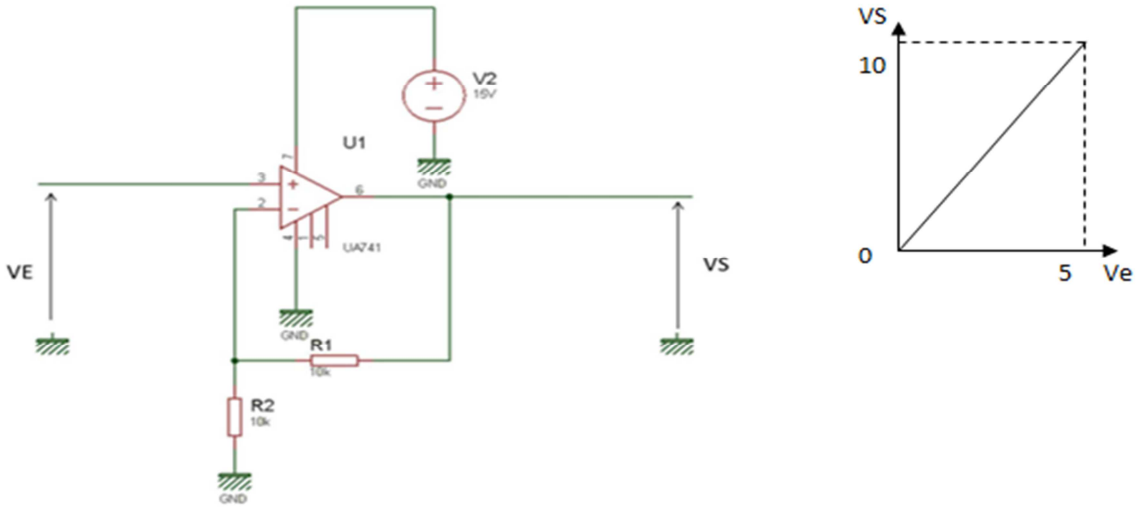


Figure **3.16 :** Montage (0V~ 5V) /(0V~ 10V)

On a : e+=e-

e+=Ve

e-= $Vs\frac{R_2}{R_1+R_2}$ (diviseur de tension)

e+=e- donc $Ve = Vs\frac{R_2}{R_1+R_2}$

$\frac{Vs}{Ve} = \frac{R_1+R_2}{R_2}$ (3.12)

Dans ce circuit on a Ve_{max}=5V et Vs_{max}=10V donc, on doit avoir un gain qui est égale à 2.

On choisit $R_1 = R_2 = 10K\Omega$.

3.8.3. Circuit (0V- 5V) / (0V- 10V)

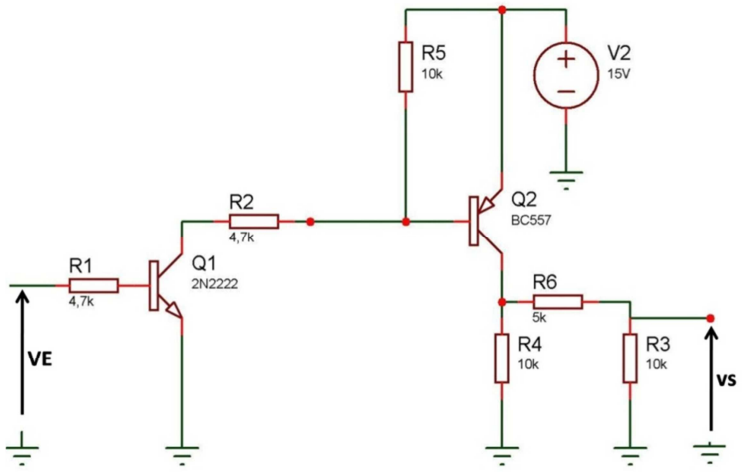


Figure **3.17 :** Montage (0V -5V)/ (0V -10V)

3.8.4 : boucle de régulation PID

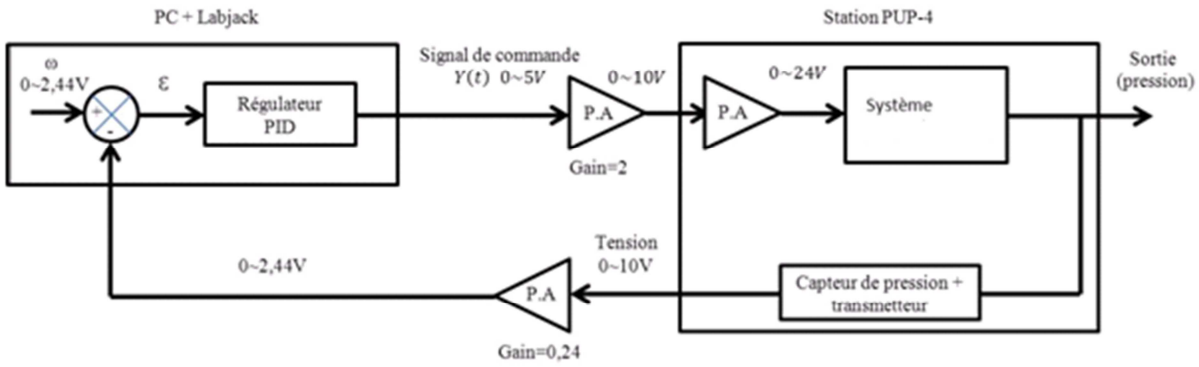


Figure **3.18 :** boucle de régulation PID

PA : Amplificateur de puissance (Power Amplifier).

3.8.5 : boucle de régulation TOR

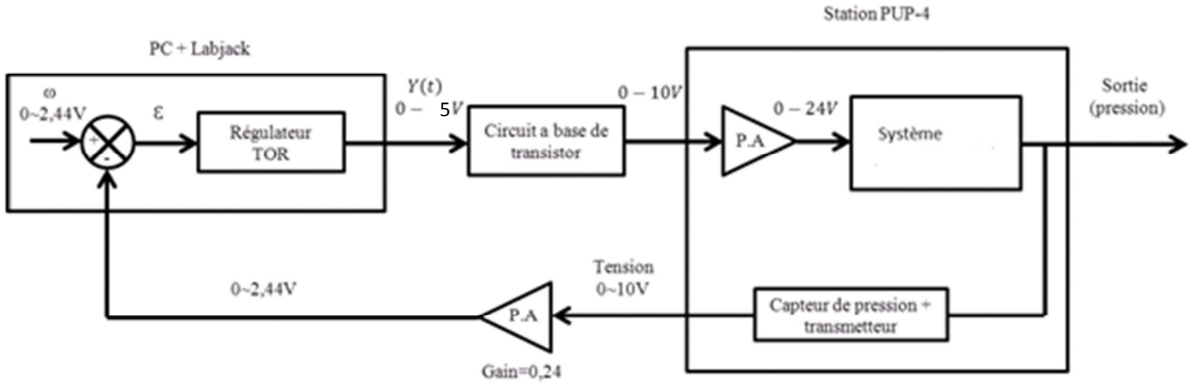


Figure **3.19 :** Boucle de régulation TOR

3.9. Conclusion

Dans ce chapitre nous avons présenté la station de pression PUP-4 qui sera l'objectif de notre étude. Puis on a décrit la carte Labjack et ses différents blocs de programmation sous LabVIEW. Mais, pour exploiter la station et la carte Labjack et réaliser une boucle de régulation de pression il faut adapter tous les signaux, ce qui nous a amené à exploiter des composants électroniques pour réaliser les circuits d'adaptation.

.

Chapitre 4

Simulations et Résultats Expérimentaux

4.1. Introduction

Ce chapitre concerne la partie application de notre travail, où on va exploiter les connaissances acquises dans les chapitres précédents pour identifier et implémenter des régulateurs (Tout ou Rien et PI) pour la station de pression PUP-4 d'ElettronicaVaneta. Cette étude a pour but, de réguler la pression à une valeur bien déterminée. Notre objectif est de trouver, la meilleure commande qui répondre à un cahier des charges imposé.

4.2. Caractéristique statique du processus

Pour déterminer la zone linéaire, il faut tracer la caractéristique statique (entrée/sortie) du processus. Dans notre cas l'entrée est la tension délivrée par une alimentation continue réglable de (0V~10V) appliquée sur une vanne régulatrice, la sortie est la tension délivrée par le capteur de pression piézorésistif. Les résultats obtenus sont représentés dans le (tableau **4.1)**

Tension d'entrée U_e(V)	1	2 ,5	3	3,5	4,5	5	6	7	8	8,5	9	9,5
Pression (bar)	0	0	0	0	0	0 ,15	0,25	0,48	1 ,05	1,28	1,55	1,8
Pression (V)	0,19	0,22	0,25	0 ,28	0,37	0,46	0,95	3	5,24	6,37	7,71	8.5

Tableau **4.1** :Caractéristique statique

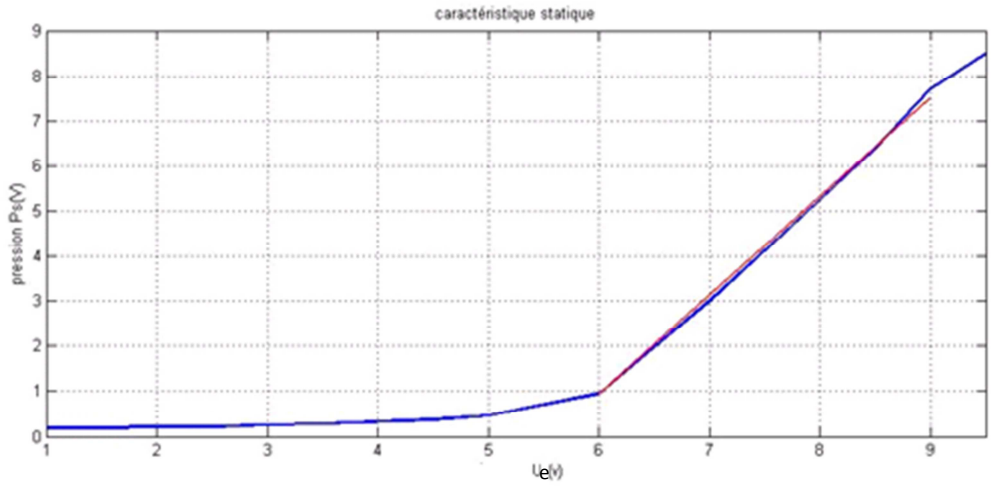


Figure **4.1** : Caracteristique statique

La courbe montre que le système est non linéaire (contient des zones linéaires).
On choisit la zone linéaire entre 6 et 9V.

4.3. Identification des paramètres du modèle

Une fois que nous avons choisis une zone linéaire, On va choisir ensuite un point de fonctionnement dans cette zone, par exemple 6V et on fait une augmentation de 2% à 5% de cette tension, dans notre cas on l'augmente jusqu'à 6.89V comme le montre la (figure **4.2**).

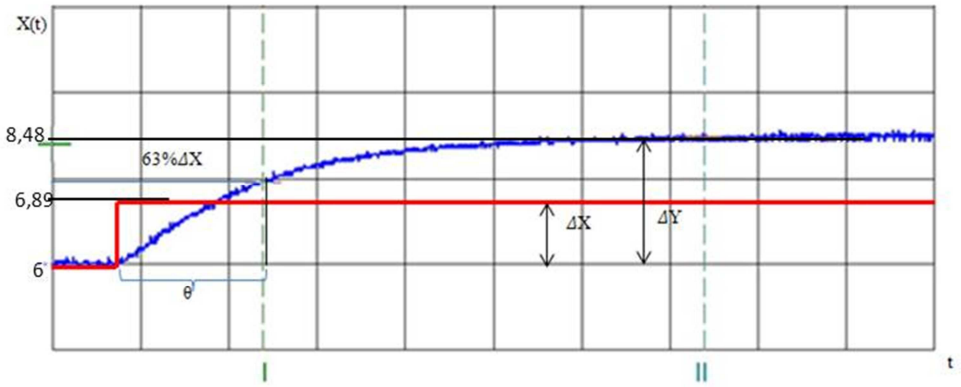


Figure **4.2** : Identification en boucle ouverte

La courbe obtenue est la réponse d'un système stable de 1er ordre sans retard, pour obtenir sa fonction de transfert, on utilise la méthode d'identification du model de 1er ordre présentée au (chapitre 1).

$$G(p) = \frac{Gs}{(1+\theta p)} \quad . \tag{4.1}$$

$$Gs = \frac{\Delta y}{\Delta x} = \frac{6.89-6}{8.48-6} = \frac{0.89}{2.48} = 0{,}36 \; . \tag{4.2}$$

$\theta = 3Sec.$

Donc le modèle obtenu est :

$$G(p)=\frac{0,36}{(1+3p)}. \quad (4.3)$$

4.4. Paramètres du régulateur PID

Nous avons vu au chapitre 1 de nombreuses méthodes pour chercher les paramètres du régulateur PID.

- **Méthode du modèle de référence**

On fixe $\theta_d = 6s$ (constante de temps désiré en boucle fermée)

Pour un PI parallèle nous avons

$$G_r = \frac{1}{G_S}\frac{\theta}{\theta_d} \quad (4.4)$$

$$T_i = \theta_d . G_S \quad (4.5)$$

Avec cette méthode on trouve

$G_r = 1.38$, $T_i = 0.03\ min$ (2.16 Sec).

4.5. Simulation

a) Réponse à un échelon (réponse indicielle)

- **Simulation Matlab**

Pour vérifier les performances de la loi de commande PI on réalise le schéma de simulation suivant :

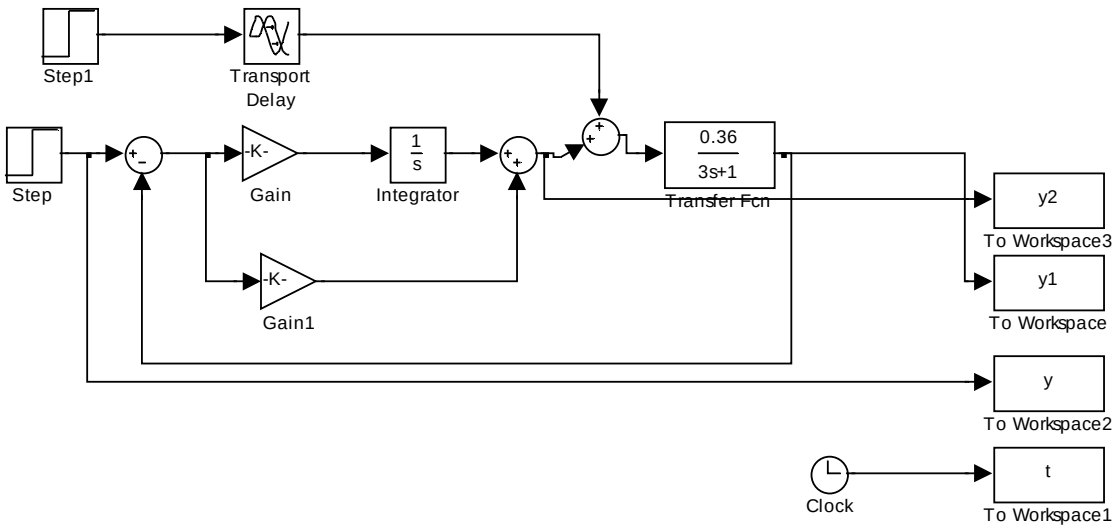


Figure **4.3 :** Schéma Simulink du régulateur PI en boucle fermée

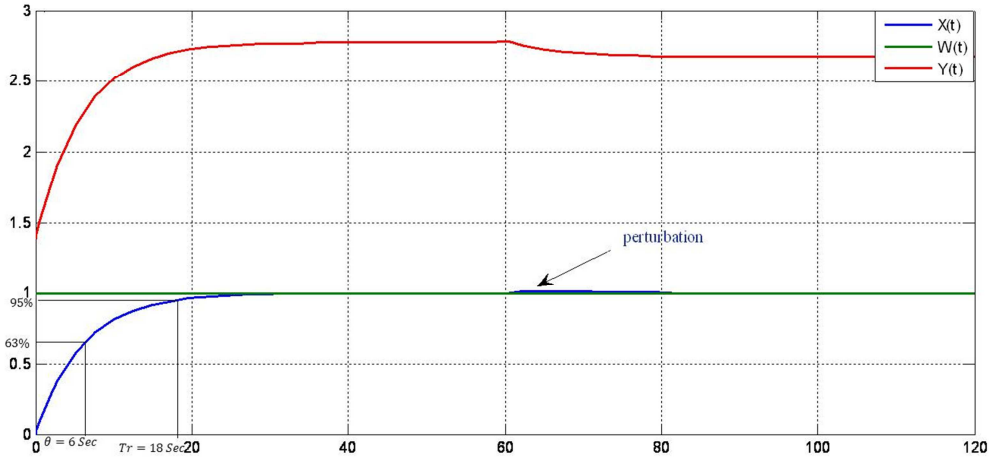


Figure **4.4** : Réponse du système en boucle fermée

On remarque que la réponse à un temps de réponse de 18 Sec approximativement et sans dépassement.

Nous avons appliqué une perturbation continue à l'instant t=60 Sec. On observe que le régulateur rejette cette perturbation après 10 Sec.

b) Stabilité

- **Tracé de Bode**

Il est utilisé afin de visualiser rapidement la marge de gain, la marge de phase et sert à étudier la stabilité des systèmes.

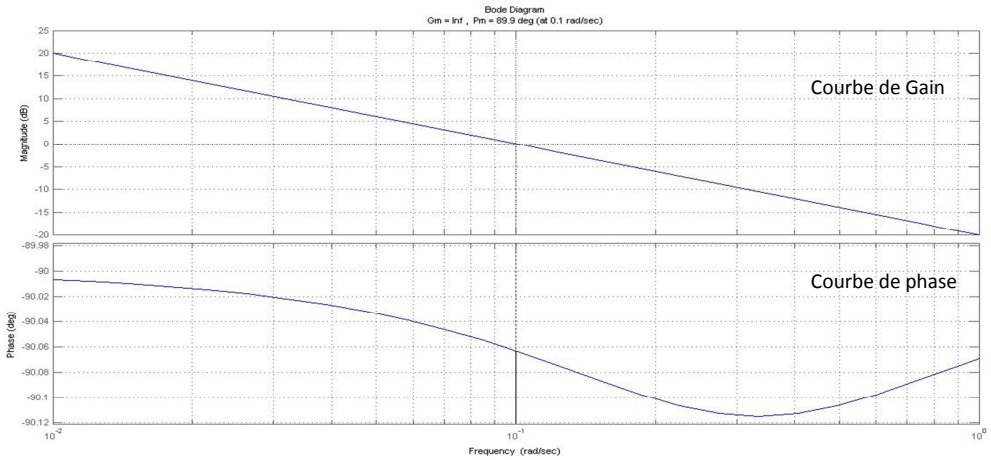


Figure **4.5** : Diagramme de Bode

φ_m=89,9°

G_m=inf

On conclut que le système est stable

4.6. Plateforme de commande

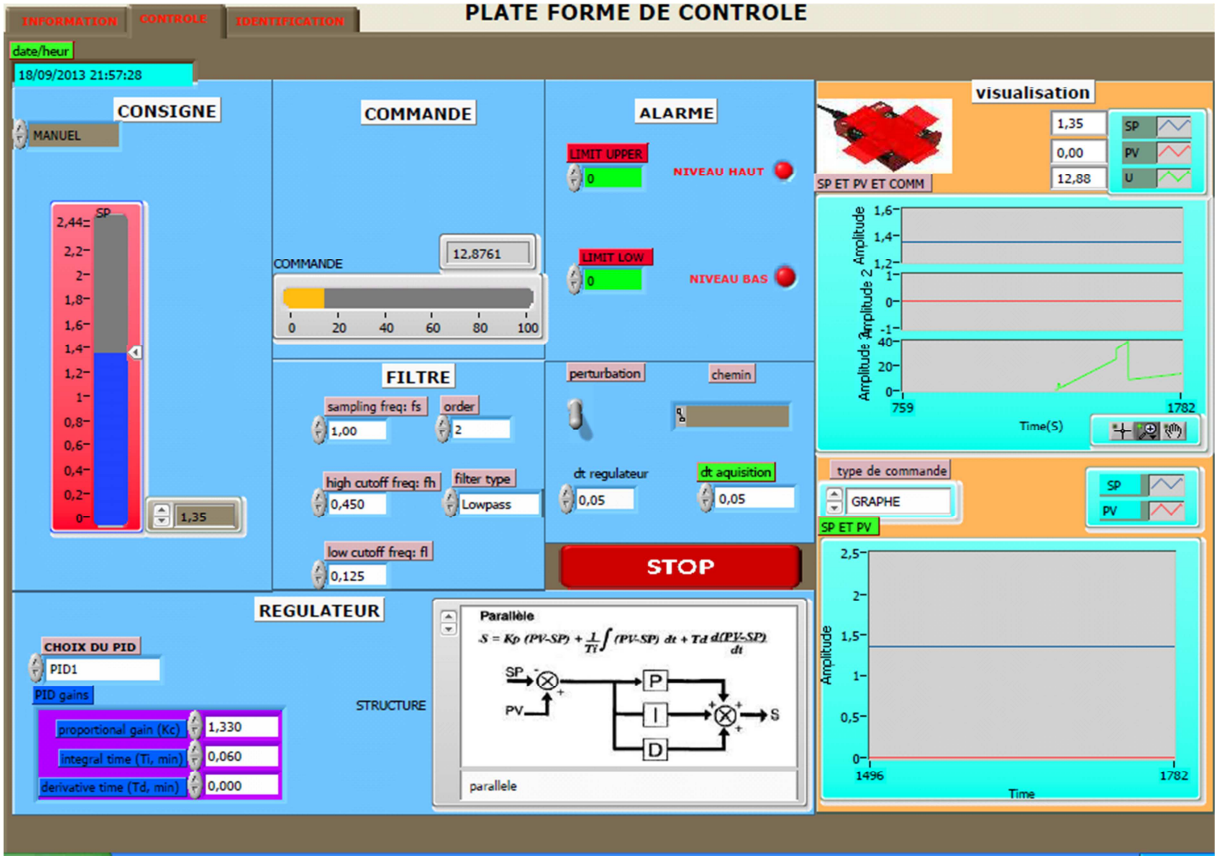


Figure **4.6 :** Interface de commande

La plateforme de commande nous l'avons structuré en 6 parties essentielles d'une façon à faire apparaître toutes les commandes et les mesures possible.

- Partie consigne: L'opérateur peut choisir entre, donner une consigne manuelle ou, programmer une poursuite bouclée.
- Partie Commande: Elle consiste à afficher l'évolution de la commande en temps réel.
- Partie régulateur : Dans cette partie on offre la possibilité de choisir le type du régulateur (PID, TOR). Si on choisit PID on peut changer sa structure (parallèle, série, mixte), et faire varier ses paramètres.

- Partie visualisation: Cette partie regroupe les différents graphes, une fenêtre pour visualiser les signaux (consigne, mesure, commande) séparément, une autre pour visualiser l'évolution de la mesure par rapport à la consigne (très utile pour la régulation).
- Partie Filtrage : Spécifier un filtre et donner ses paramètres.
- Partie alarme : Alarme réglable pour prévenir l'operateur.

4.6.1. Programme Labjack sous LabVIEW

- **Programme pour l'acquisition de données**

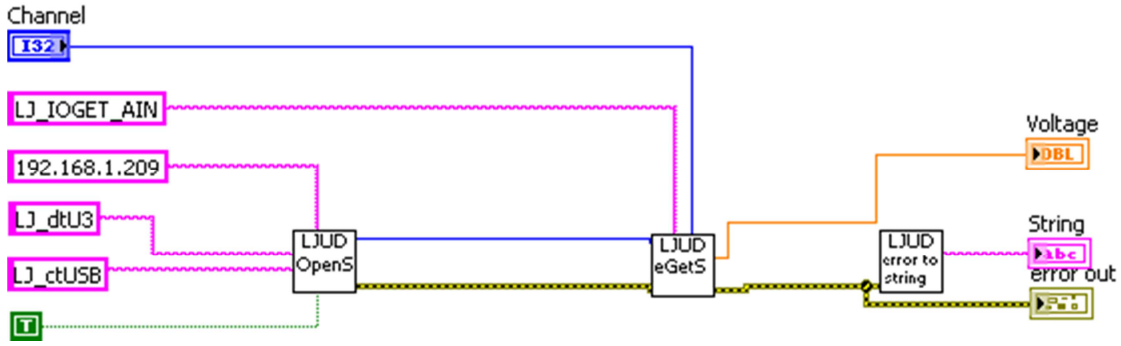


Figure **4.7 :** Programme pour acquisition

Le bloc LJUD Opens permet d'appeler le LabJack U3.

On a mit la commande LJ_ioGET_AIN pour configurer en entrée analogique un port dont le numéro est donné sur l'entrée channel .

- **Programme pour la sortie analogique**

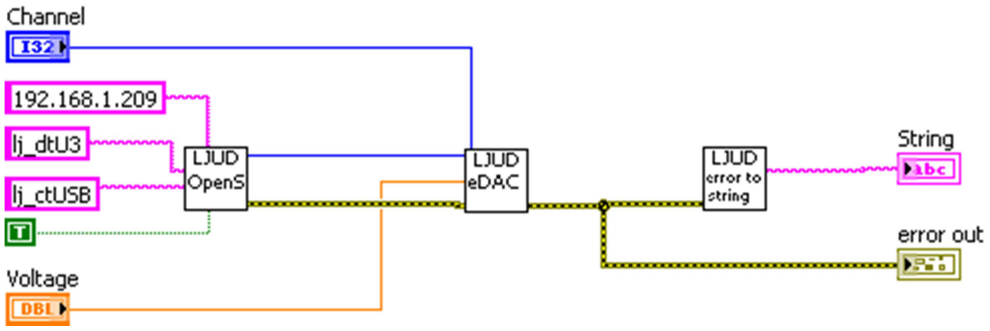


Figure **4.8 :** Programme sortie analogique

Pour la sortie analogique on utilise directement le bloc eDAC.

Channel nous permet de sélectionner DAC 0 ou DAC 1.

- **Programme pour la sortie numérique**

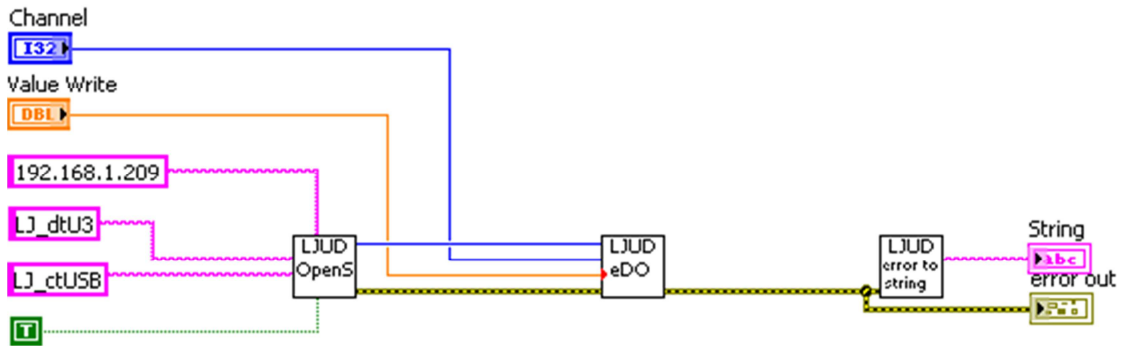


Figure **4.9 :** Programme pour sortie numérique

Le boc LJUD eDO nous permet de faire sortir un signal numérique de (0-5V) sur l'un des ports spécifiés sur chanel

4.6.2. Programme régulation TOR sur LabVIEW

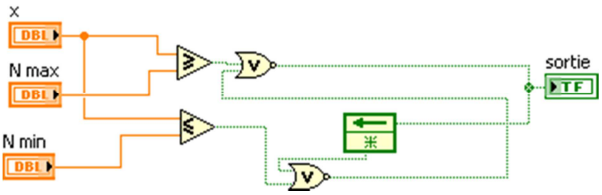


Figure **4.10** : programme tout ou rien LabVIEW

Avec ce programme, la sortie boolien est mise à 1 si X ≤ Nmin et mise à 0 si X ≥ Nmax.

4.6.3. Programmation d'un PI parallèle sous LabVIEW

Pour programmer le PID sous LabVIEW nous avons traduit l'équation de la commande donnée au chapitre 1.

$$U(t)= K_c\varepsilon(t) + \frac{1}{T_i}\int_0^t \varepsilon(t)\,dt + T_d\,\frac{d\varepsilon(t)}{dt} \qquad (4.6)$$

Les deux blocs dérivateur et intégrateur fonctionnent point par point, tel que la durée que mettent ces derniers pour passer d'un point à l'autre est donnée par dt (période d'échantillonnage).

Pour réaliser le PI, il suffit d'éliminer l'action dérivée en donnant 0 à T_d.

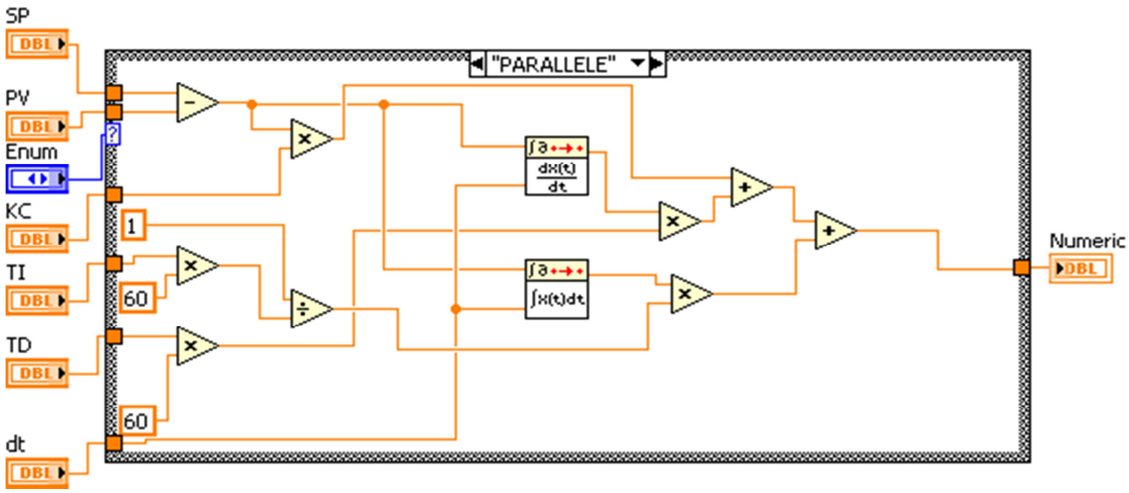


Figure **4.11** : PID parallèle avec LabVIEW

4.6.4. Réalisation d'une poursuite avec LabVIEW

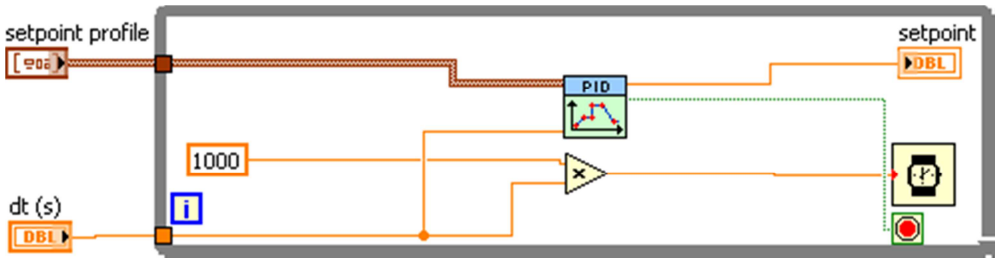


Figure **4.12 :** Programme de la poursuite avec LabVIEW

Dans la palette PID de LabVIEW, on trouve le bloc Setpoint profile qui génère des consignes différentes dans le temps.

On écrit notre poursuite dans le cluster d'entrée sous forme de deux variables (la valeur de la consigne (V), le temps que reste cette consigne(Sec)).

4.6.5. Archivage des données

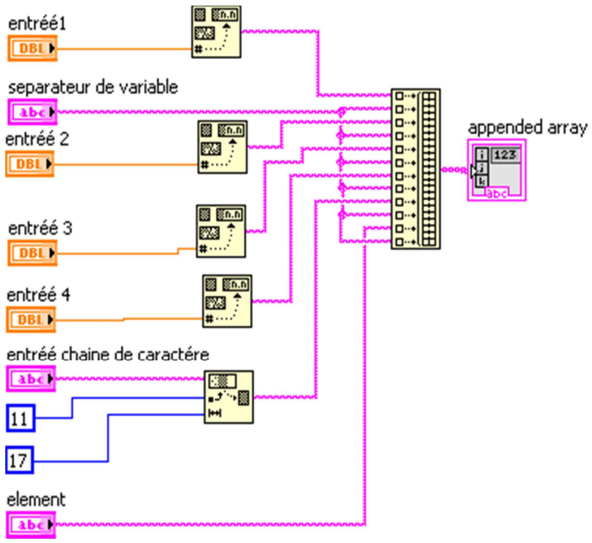


Figure **4.13 :** Programme pour l'archivage

Ce programme converti des entiers à des chaînes de caractère pour créer un fichier d'archivage.

4.6.6. Programmation d'une alarme

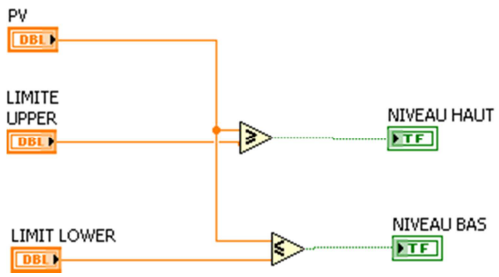


Figure **4.14 :** programme d'une alarme

On compare PV (signal de mesure) à une limite basse et à une limite haute. On récupère l'information sous forme d'un boolien qu'on va l'utiliser pour faire clignoter des Leds ou faire activer des effets sonores.

4.7. Programmes automate S7-200

4.7.1. Régulation TOR

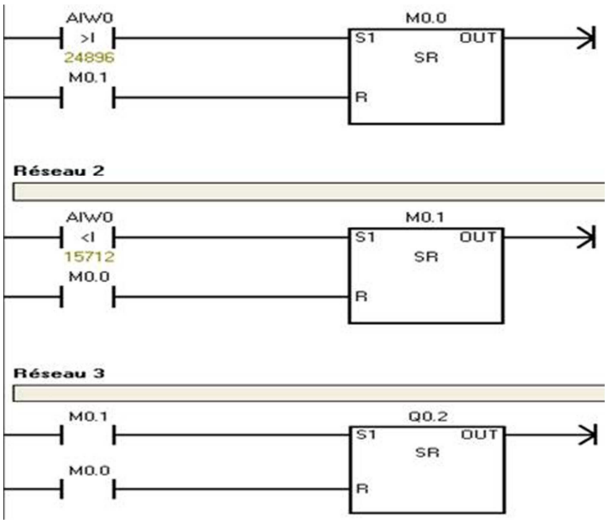


Figure **4.15 :** Programme régulation TOR automate

L'entrée analogique AIW0 est comparée à deux valeurs 24896 qui correspond à 7.78V et 15712 qui correspond à 4.91V.

- Si AIW0 $\geq$ 24896, M0.0 est activé, M0.1 est désactivé et Q 0.2 est désactivé, donc l'automate délivre 0V sur la sortie Q 0.2.
- Si AIW0 $\leq$ 15712, M0.1 est activé, M0.0 est désactivé et Q 0.2 est activé, donc l'automate délivre 24V sur la sortie Q0.2.

4.7.2. Régulateur PID

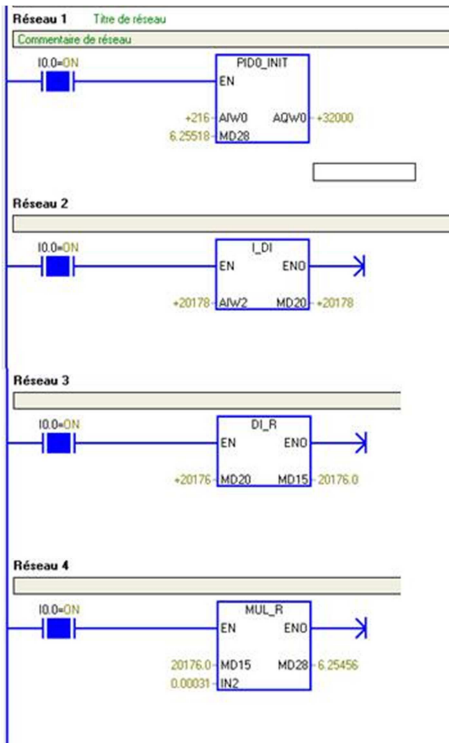


Figure **4.16 :** Programme poursuite PID

4.8. Résultats expérimentaux

4.8.1 Régulation TOR

- **Avec automate**

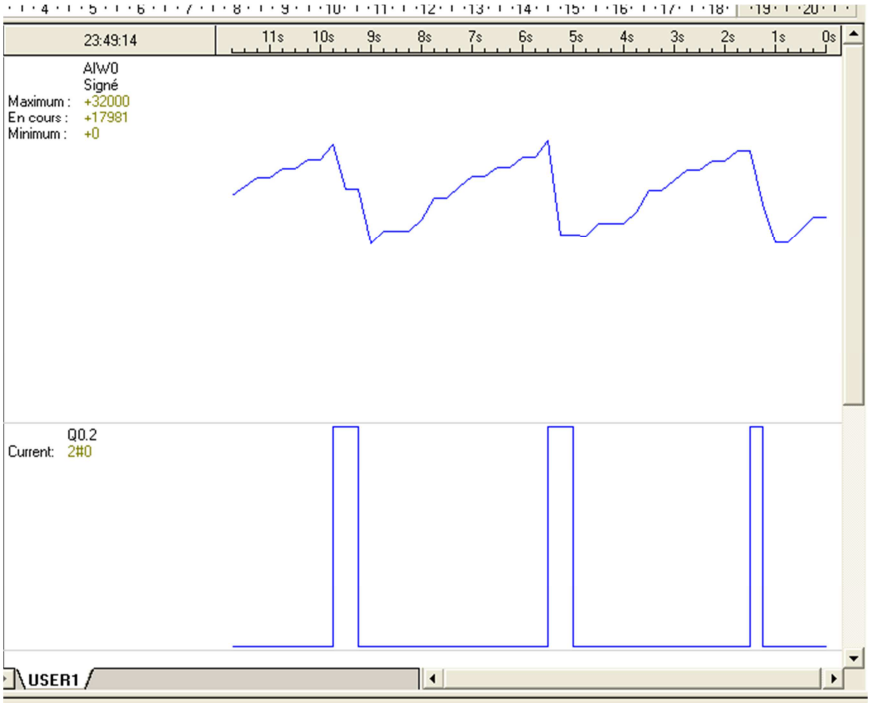


Figure **4.17 :** TOR automate

On remarque que :

La commande agit différemment sur les trois cycles (le temps d'enclenchement varie).

Les oscillations de la mesure sont dues à des bruits de mesure.

- **Avec LabVIEW**

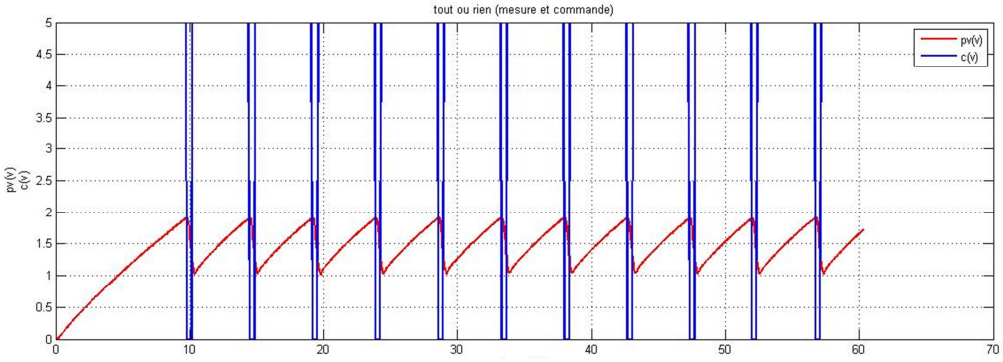


Figure **4.18** : TOR LabVIEW

La régulation tout ou rien obtenu avec le logiciel LabVIEW est plus appréciable parce que avec LabVIEW nous avons eu un signal de mesure sans oscillations et aussi le temps d'enclenchement de la commande est uniforme.

4.8.2 Régulation PI

- **Avec automate**

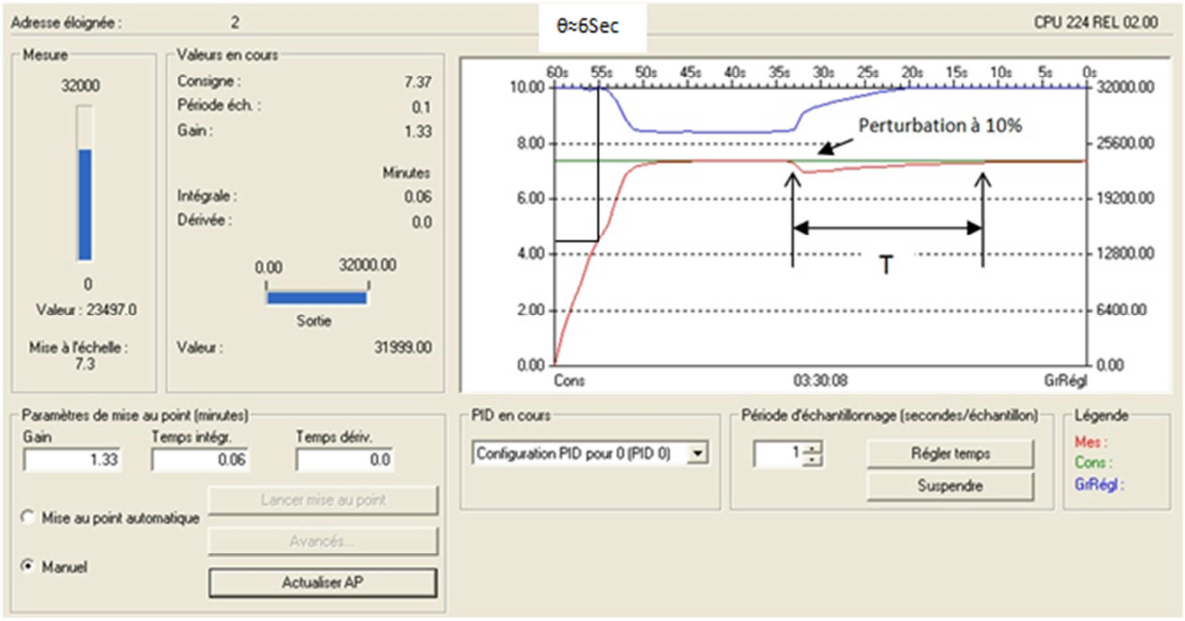


Figure **4.19** : Essai en boucle fermée avec automate S7-200

On remarque Que le régulateur PI à compenser la perturbation continue (ouverture à 10% de la vanne manuelle), sachant qu'il a mis T=25 s pour ramener la mesure à la consigne imposée qui est de 7.37(V).

Après l'application de la perturbation à cet instant la commande à augmenter d'une manière douce pour atteindre 100% (fermeture complète de vanne), elle reste à cette valeur tout le long du cycle pour maintenir la mesure à la consigne imposée.

- **Avec LabVIEW**

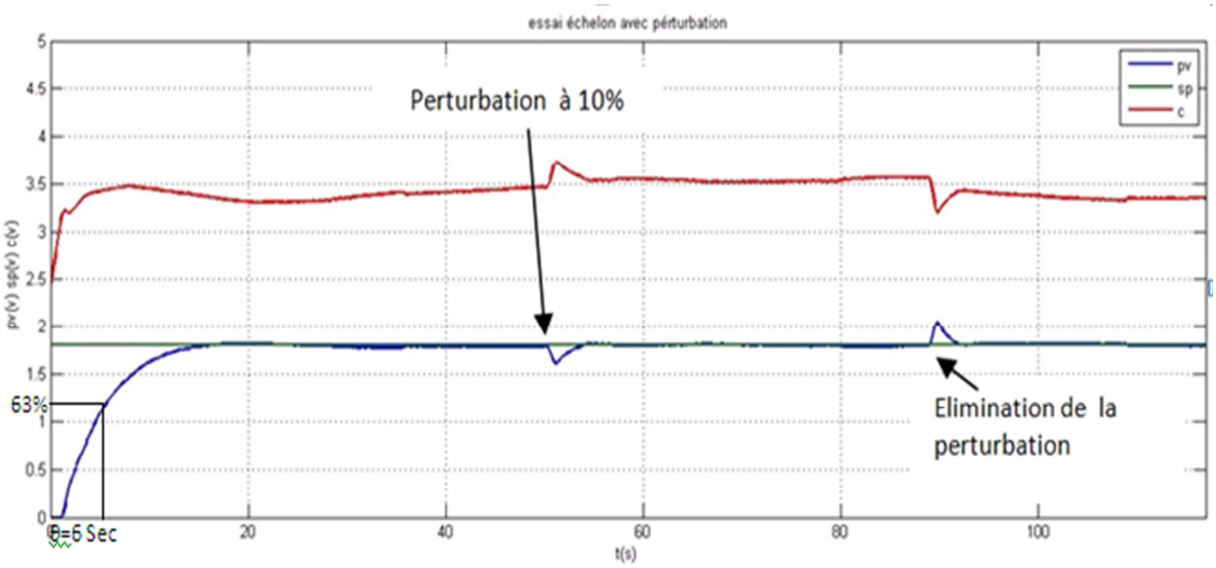


Figure **4.20** : Essai en boucle fermée LabVIEW

On remarque que le régulateur PI à compenser la perturbation qui est de 10% (ouverture de la vanne manuelle), il a mis 10 secondes pour amener la mesure à la consigne imposée qui est de 1.8V sachant que l'automate a pris 25Sec. La commande a augmenté d'une manière douce pour atteindre 100% (fermeture complète de vanne), elle reste tout le long du cycle pour maintenir la mesure à la consigne imposée.

Avec le logiciel LabVIEW nous avons obtenu un résultat similaire qu'avec l'automate sauf que, le temps qui a mis le logiciel pour récupérer l'effet de la perturbation est moins important. Donc avec le logiciel LabVIEW on a gagné en robustesse.

4.8.3 Réalisation d'une poursuite

- **Avec l'automate**

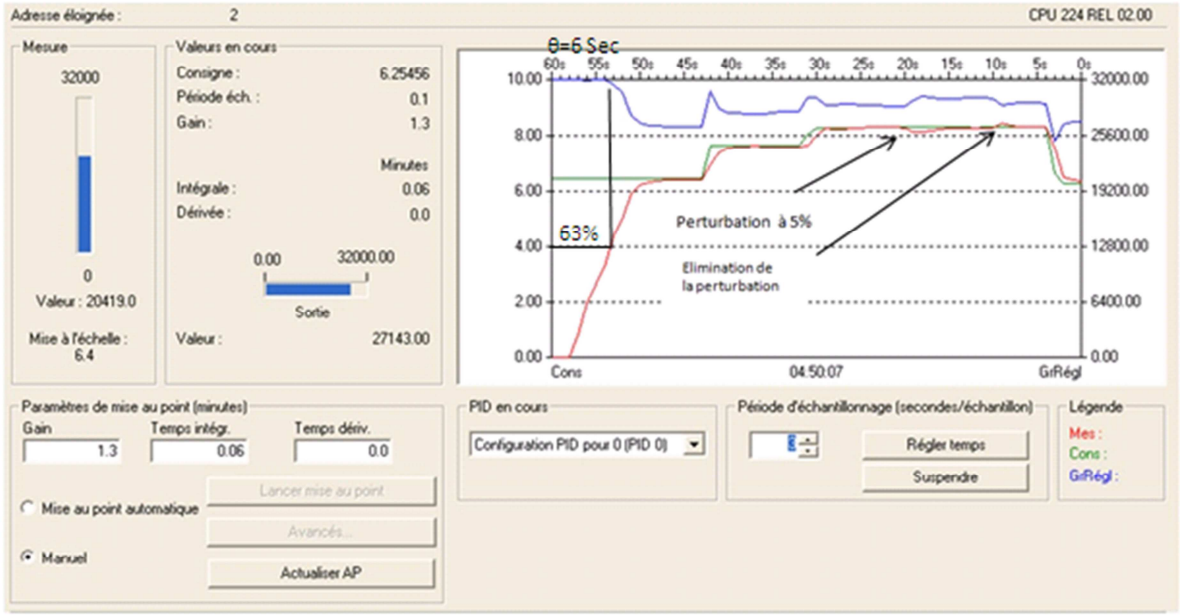


Figure **4.21 :** Poursuite automate S7-200

Au début, le signal de commande est à 100%, cela veut dire que la vanne est complètement fermée, dès que la consigne change, la mesure suit.

L'application de la perturbation à l'instant 40 secondes est éliminé après 10 Sec. Cet essai nous a permis de conclure que le régulateur PI agit convenablement c'est-à-dire de pression ramène le système à la consigne désirée quel que soit la nature et la durée de la perturbation.

- **Avec LabVIEW**

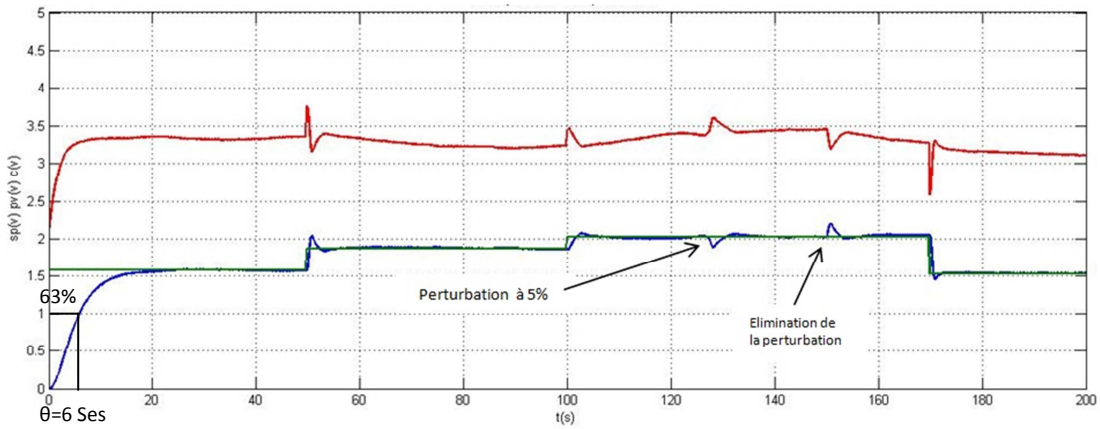


Figure **4.22 :** Poursuite LabVIEW

Les performances du système reste les mêmes qu'avec l'automate, l'avantage, avec le logiciel LabVIEW la poursuite est plus facile à programmer.

4.9. Conclusion

La régulation repose sur la détermination des paramètres de régulateur PID. Pour cela, nous avons déterminé les paramètres d'un régulateur PI à structure parallèle qui doit assurer un bon fonctionnement de la station, et répondre aux exigences voulus qui sont de ramener la mesure à la consigne désirée et d'avoir un signal de commande doux. Au cours de ce chapitre, nous avons vu que le logiciel LabVIEW et S7-200 sont deux outils intéressants pour le contrôle des processus, le premier en utilisant une carte d'acquisition et le deuxième à travers un automate.

A partir des résultats obtenus, nous avons constaté que LabVIEW est un logiciel vigoureux pour la conception des applications en temps réel, destinées au contrôle et commande des processus.

Conclusion Générale

L'objectif de notre travail a été l'implémentation de la régulation Tout ou Rien et PID sous automate S7-200, ensuite le développement d'une plate forme de commande à base du logiciel LabVIEW dans le but de contrôler la station de pression PUP-4 de la société d'ElettronicaVeneta.

Nous avons mené dans le cadre de ce travail, une large recherche bibliographique afin d'avoir une meilleure idée sur l'utilisation du logiciel LabVIEW dont le principal intérêt est de concrétiser les connaissances théoriques dans le domaine pratique.

Nous avons pris comme application la régulation de pression en utilisant l'automate S7-200 de la firme Siemens et une carte d'acquisition Labjack U3. A partir des résultats obtenus nous avons constaté que la plate forme de commande à base de LabVIEW donne des résultats très satisfaisants. Il constitue donc un outil très intéressant grâce à ses propriétés.

Enfin, nous avons proposé une plate forme ouverte pour une éventuelle implantation de nouvelles structures de commande en vue de piloter un système dynamique à base de logiciel LabVIEW. On mentionne la possibilité d'implémenter d'autres lois de commandes telles que : La logique floue, neuro-floue, réseaux de neurones…etc.

Ce travail a été bénéfique, car nous avons pu mettre en pratique les connaissances acquises pendant notre cursus d'études.

Enfin, on espère que nos efforts puissent servir à quelque chose et que ce mémoire soit un bon guide pour les promotions futures.

Références Bibliographiques

[1] PROUVOST. P., *« Automatique contrôle et régulation ».* Edition Dunod, 2004.

[2] SERMONDADE.C, *« Régulation Tome 1, Régulation élémentaire, notions de base, éléments de régulation »*, Editions Nathan, Paris, 1994.

[3] PROUVOST.P, « *Instrumentation et Régulation en 30 fiches* », Dunod, Paris, 2010.

[4]VALENCE. J.M, « *Le carnet du régleur : Mesure et régulation* », Edition Valence, 2005.

[5]PROVOUST. P, *« Contrôle Régulation, Exercices et problèmes résolus »*, Editions Nathan, Paris, 1997.

[6] RIVOIRE.M / FERRIER.J.L *« Commande par calculateur Identification »*, Editions Eyrolles,Paris,1997.

[7] RICK .B, *«LabVIEW Advanced Programming Techniques*», CRC Press LLC, 2001.

[8] HALVORSEN. H.P, « LabVIEW Programming »,Telemark University,2010.

[9] LabVIEW, « *Principe de base de LabVIEW National Instruments Corporation* », Edition de Août 2006.

[10] CHAUVASEAU C, « *Amplificateur Opérationnel- Cours et Problème*s»,version du 11 février 2013.

[11] « *Manuel Professeur/Etudiant, Contrôle de Processus avec API(PLC), mod. PRC/EV Processus de Pression mod. PUP-4/EV*», Italie.

[12] « *SIMATIC Automate Programmable S7-200, Manuel système* », Seimens, 2000.

[13] « *Datasheet 2N2222, BC557 and UA741 ».*

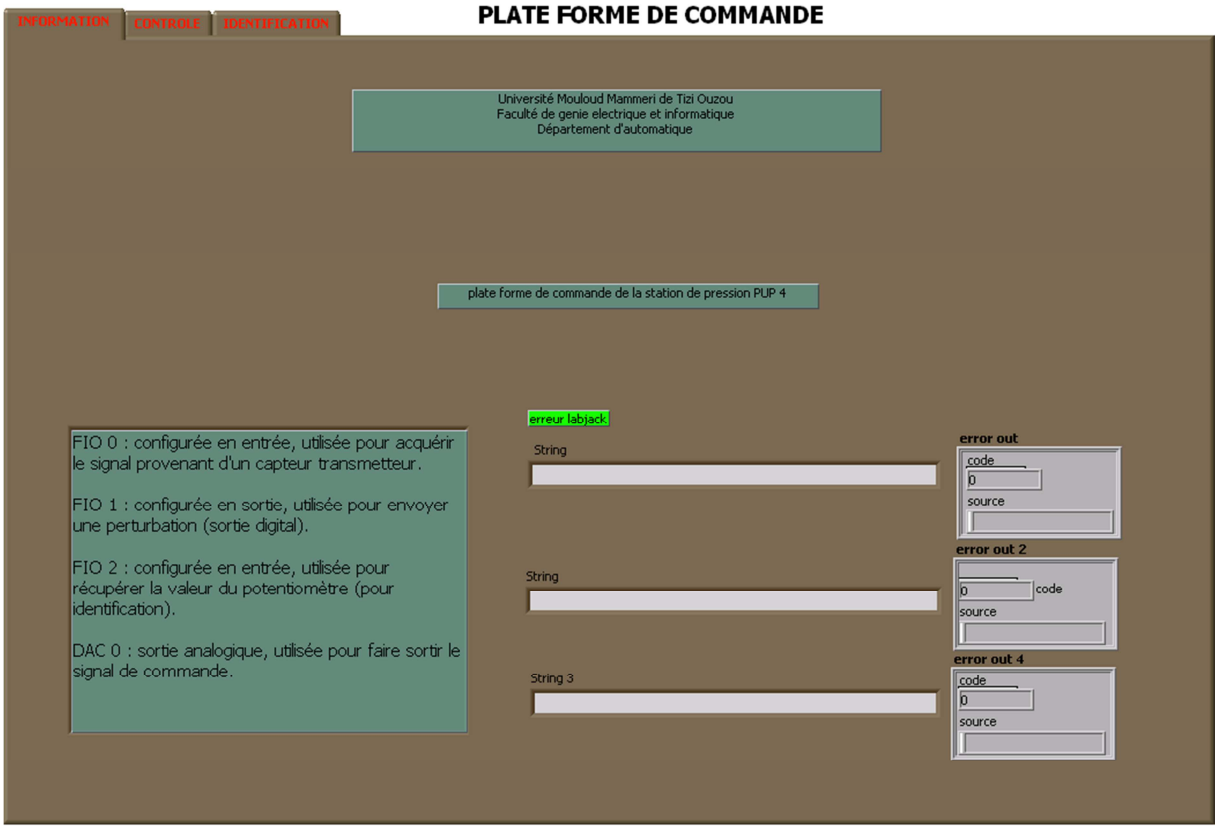
Logiciels Utilisés

MATLAB 2010

Automate S7-200

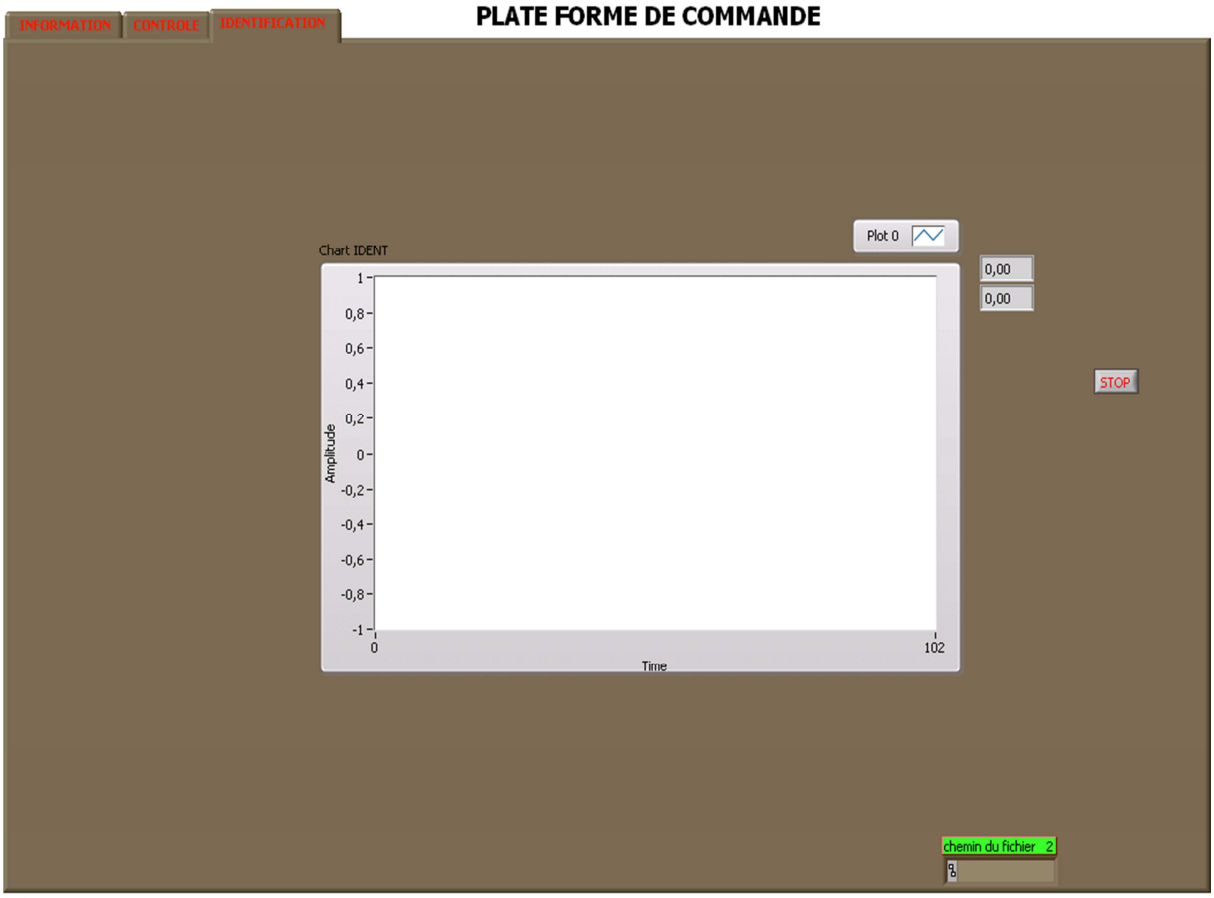
LabVIEW 2011

1. Face avant (information)



TAB

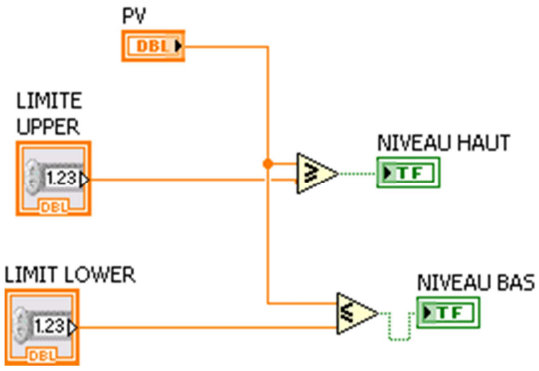
2 .Face avant (identification)



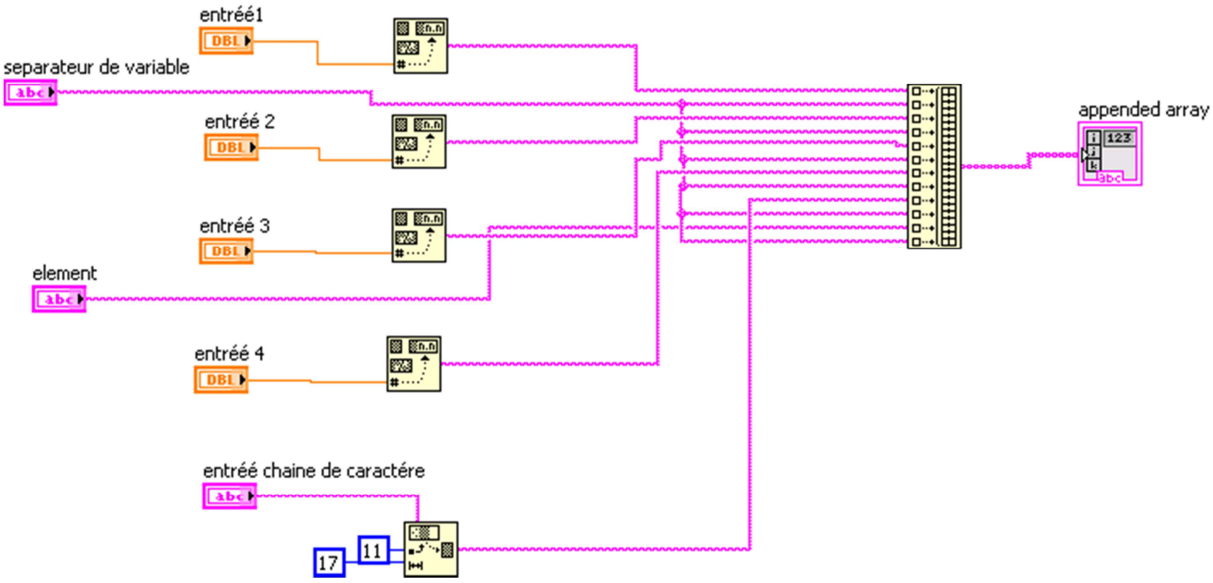
TAB

4. Sous-programme

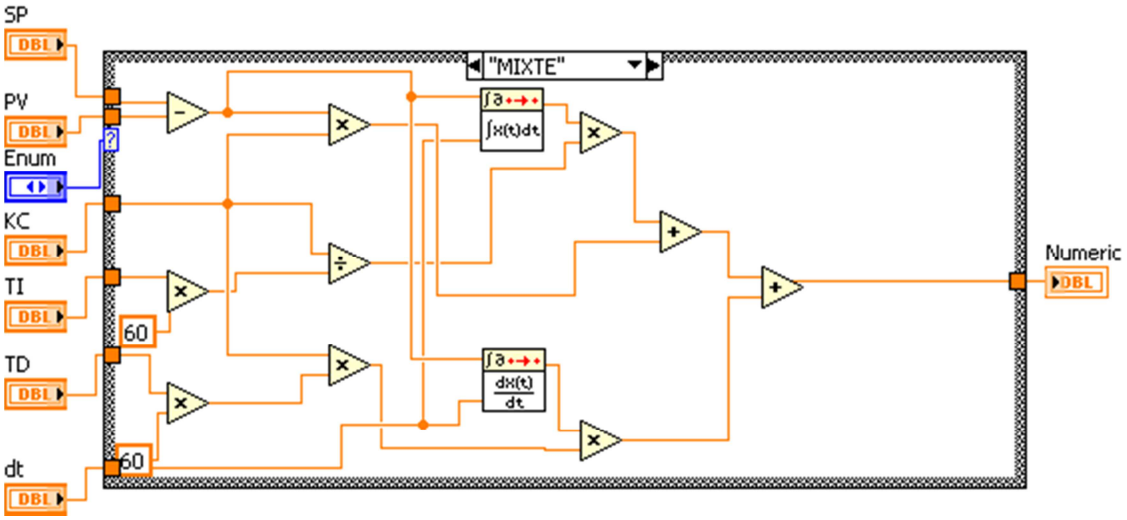
4.1 Sous-programme alarme

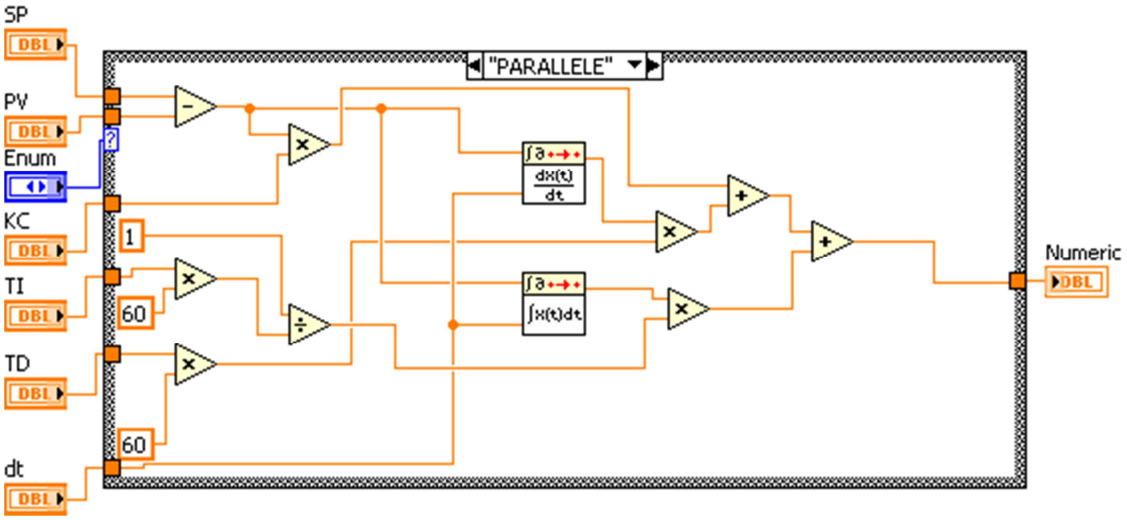


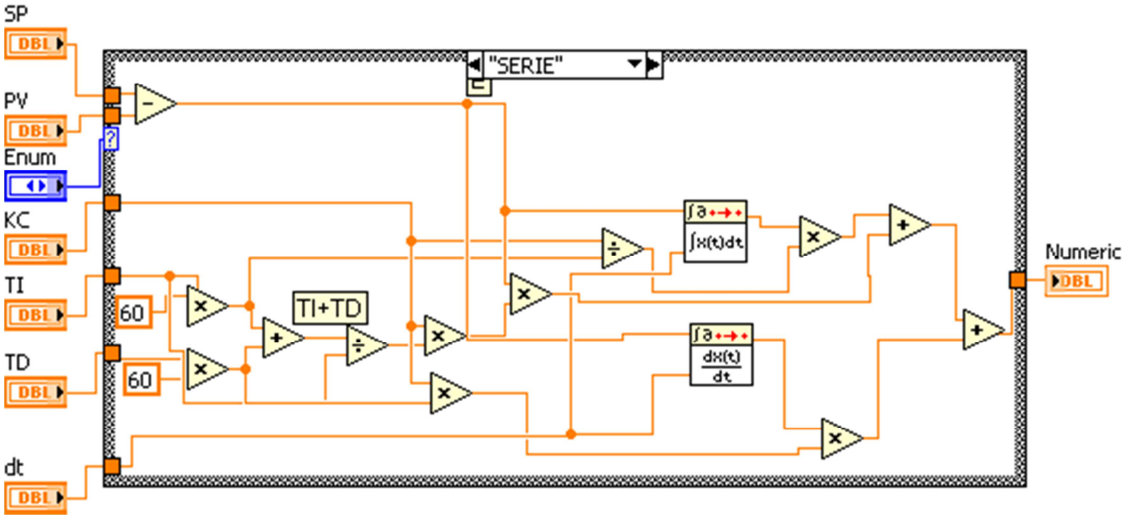
4.2 Sous-programme affichage



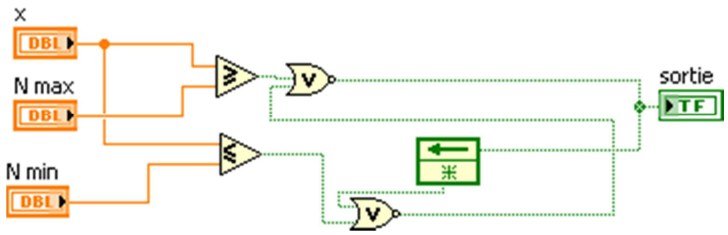
4.3 Sous-programme PID



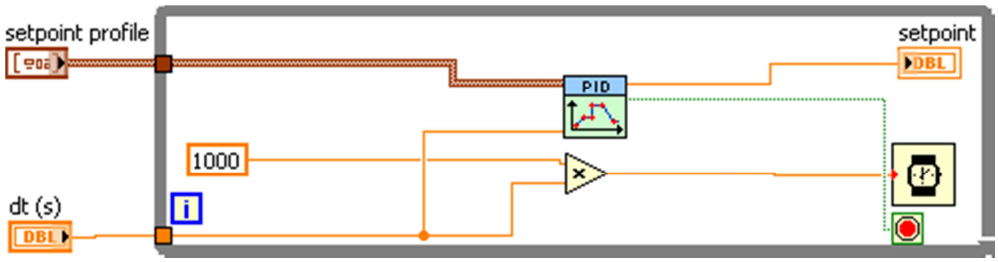




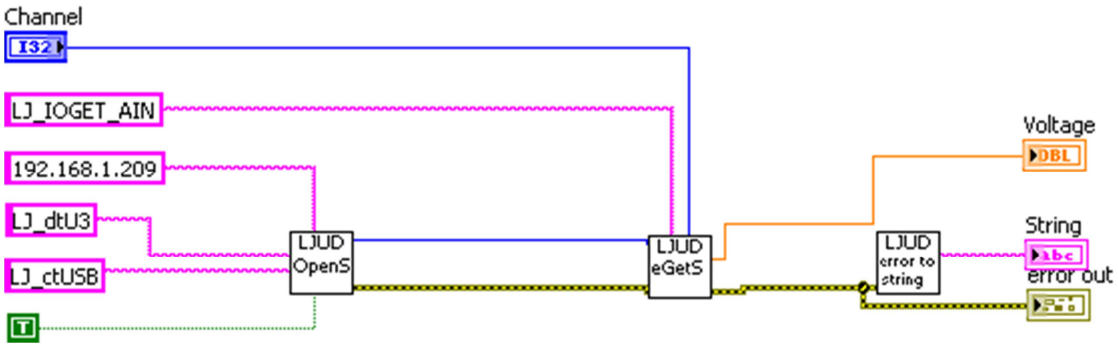
4.4 Sous-programme TOR



4.5 Sous-programme poursuite



4.6 Programme pour l'acquisition de données



4.7 Programme pour la sortie analogique

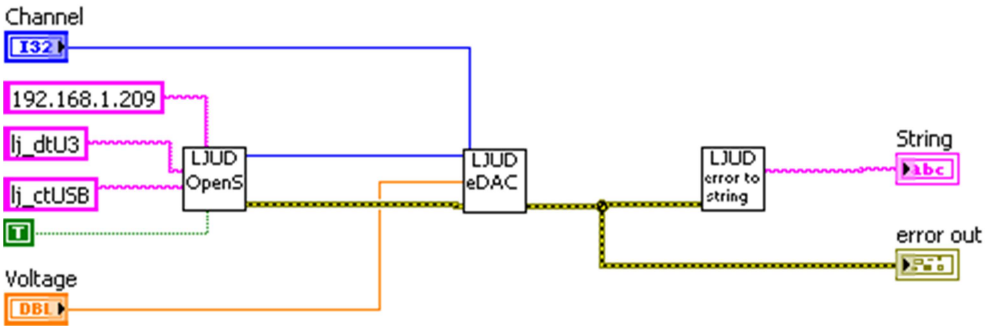


Diagramme contrôle

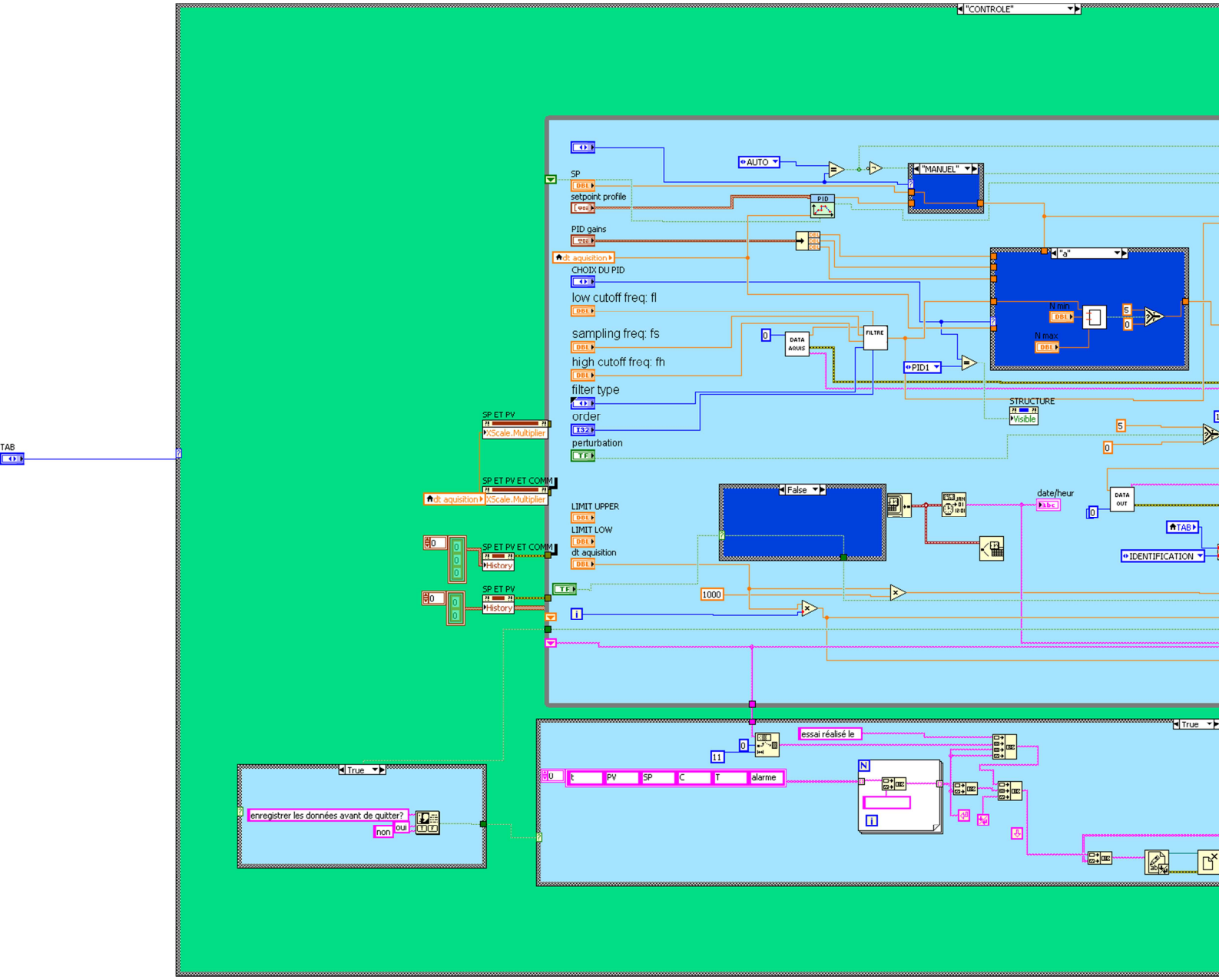


Diagramme identification

TAB