وزارة البحث العلمي والتعليم العالي
MINISTERE DE L'ENSEIGNEMENT SUPEREUR ET DE LA RECHERCHE SCIENTIFIQUE
جامعة عبد الحميد بن باديس مستغانم
Université Abdelhamid Ibn Badis Mostaganem
كلية العلوم و التكنولوجيا
Faculté des Sciences et de la Technologie
DEPARTEMENT DE GENIE DES ELECTRONIQUE

N° d'ordre : M ...../GE/2020

MEMOIRE

Présenté pour obtenir le diplôme de

MASTER EN ELECTRONIQUE

Option : électronique des systèmes embarqués

Par

Nom et Prénom : Meslem Zakariya

Nessar Ahmed

Intitulé du sujet

Implémentation d'un régulateur PID sur Arduino nano pour la régulation en position d'un moteur à courant continu

Soutenu le 2020 devant le jury composé de :

Président :	Mr B.Djelti	Grade MC	Université de Mostaganem
Examinateur :	Mr M. Bentoumi	Grade MC	Université de Mostaganem
Examinateur :	Mme K.Berradja	Grade MC	Université de Mostaganem
Rapporteur :	Mr N.Abedllaoui	Grade MA	Université de Mostaganem

Année Universitaire 2019/2020

Dédicaces

Nous dédions ce mémoire à nos chers parents, nos seours et nos frères et toute la famille de nessar et meslem, sans oublier nos chérs amis et proches.

En dernier, on passe une grande dédicace pour l'ensemble du groupe M2 ESE Promotion 2019/2020 pour leurs solidarités, leurs attachements. Merci

Remerciements

Dieu merci pour la santé, la volonté, le courage et la détermination qui nous ont accompagnés tout au long de la préparation et l'élaboration de ce travail et qui nous ont permis d'achever ce modeste travail.

Nous tenons à remercier notre encadreur, Monsieur Abdellaoui Nasreddine son soutient, sa disponibilité ainsi que ses précieux conseils.

Nous tenons à exprimer nos remerciements les plus sincères aux membres du jury : Mr.Djelti et Mr.Bentoumi et Mme.Berradja, qui ont accepté d'examiner ce travail.

Table des matières

Introduction générale

Les avantages d'un régulateur numérique sont nombreux dont le principal est sans nul doute la facilité avec laquelle une loi de commande même compliquée peut être programmée. En effet, la modification des coefficients de cette loi en vue de l'adapter à différents systèmes à régler est une opération extrêmement simple, automatique dans certains cas .

Quand au changement de la loi de commande (modification de l'architecture du régulateur), elle n'implique qu'une reprogrammation, suivie d'une compilation et d'un chargement ou téléchargement du nouveau code exécutable. Et sur cette base, un régulateur numérique supplante indiscutablement sa version en analogique.

De plus, disposer d'un microcontrôleur ou processeur pour effectuer de la régulation, permet d'en profiter pour lui faire exécuter de multiples autres tâches telles que : la surveillance, la protection, le monitoring …etc…, à un point tel que la fonction de régulation devient en complexité et en temps d'exécution très secondaire et sur ce plan-là, un régulateur numérique est imbattable.

C'est autour de ce contexte que se situe l'objet de ce projet. Il s'agit réaliser un régulateur PID numérique de position angulaire d'un moteur à courant continu à l'aide d'une plateforme arduino nano.

Le projet en soi, étant relativement complexe et nécessitant d'abord l'acquisition d'un minimum de compétences pour pouvoir le réaliser, nous avons adopté la méthodologie suivante pour le faire aboutir : Dans un premier temps, une recherche bibliographique approfondie a été effectuée sur des projets similaires. Nous avons ensuite fait une sélection quand à « l'architecture du pid à implémenter. Ensuite, nous avons fait des choix quand aux « outils de conception » , « langages et environnement de développement ».

Ceci étant dit, le mémoire de notre projet est organisé en quatre chapitres. Le premier est consacré aux vocabulaires et à la présentation de toutes les notions relatives à l'asservissement et à la régulation des systèmes, le deuxième au contrôle par PID, le troisième aux méthodes de synthèse d'un régulateur PID numérique et le quatrième à la réalisation matérielle et logicielle et à la présentation des résultats obtenus.

Chapitre 1 : L'asservissement et la régulation de systèmes

1. Généralités sur les systèmes

1.1. Définition d'un système

De manière générale, un système est un ensemble d'objets interagissant entre eux en vue de réaliser une tâche donnée. Il peut être simple ou complexe.

1.2. Représentation d'un système

On le rappelle : En automatique, on représente un système par un schéma fonctionnel (Figure 1).

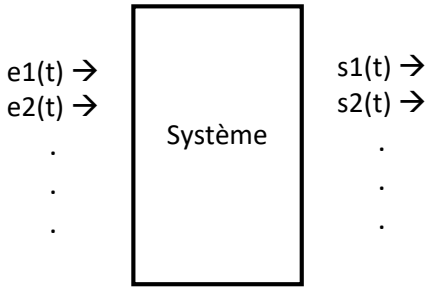


Figure 1 : représentation d'un système quelconque

1.3. Classification des systèmes

Les systèmes peuvent être scindés en trois grandes classes (figure 2).

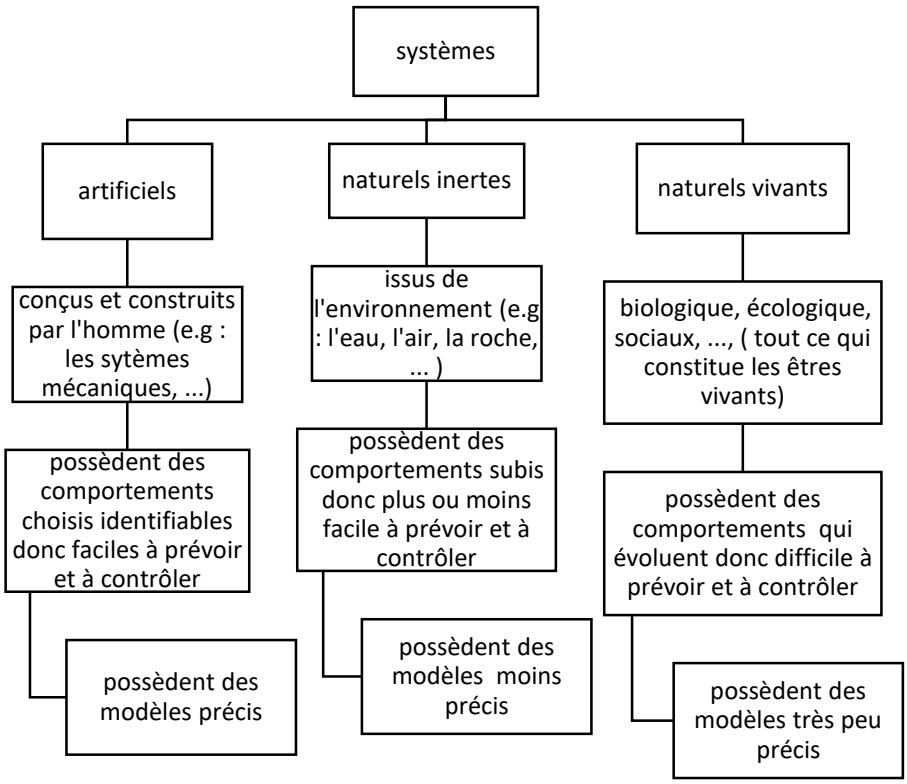


Fig.2 : Les différentes classes de systèmes

L'asservissement, porte essentiellement sur les systèmes construits par l'homme.

1.4. Propriétés d'un système

Afin de pouvoir contrôler un système, il est nécessaire de connaître un certain nombre de ses propriétés et de ses caractéristiques à savoir :

- Le nombre et la nature des entrées et des sorties ;
- Le régime permanent ou établi
- Le comportement statique ;
- Le comportement dynamique (temps de montée, nombre et période des oscillations, etc);
- Le modèle
- La linéarité/la non-linéarité ;
- La causalité
- La variance/l'invariance
- La staticité/ La dynamique

1.4.1. Le nombre et la nature des entrées et des sorties ;

On rencontre les systèmes suivants :

- Systèmes SISO (Single Input, Single Output): une seule entrée, une seule sortie.
- Systèmes MIMO (Multiple Input, Multiple Output) : plusieurs entrées, plusieurs sorties.
- Systèmes SIMO (Single Input, Multiple Output) : une seule entrée, plusieurs sorties.
- Systèmes MISO (Multiple Input, Single Output) : plusieurs entrées, une seule sortie.

Dans les paragraphes qui suivent, notre intérêt va porter tout particulièrement sur les systèmes SISO (Figure 3) étant donné que celui qu'on veut contrôler dans notre projet fait partie de cette catégorie.

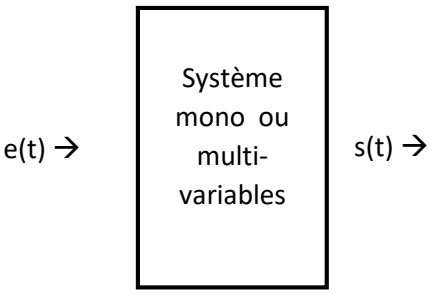


Figure 3 : représentation d'un système SISO

1.4.2. Le régime permanent ou établi d'un système

Il caractérise la réponse stabilisée d'un système à une entrée quelconque.

1.4.3. Comportement statique et dynamique d'un système

On donne à la figure 3 suivante à titre illustratif un exemple de tels comportements d'un système.

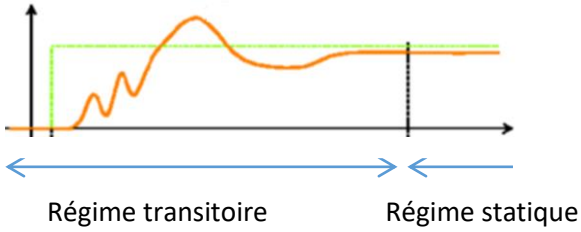


Figure 4 : Exemple de comportement dynamique et statique d'un système

Le comportement dynamique est caractérisé par le régime transitoire. Il caractérise l'évolution de la réponse avant que le régime permanent ne soit atteint. Il est souvent difficile de le qualifier et quantifier sur la base de l'analyse temporelle seule. Il nécessite des outils spécifiques tels que les transformées de Fourier et de Laplace.

Pour le comportement statique, c'est le régime permanent dans le cas où l'entrée est constante. Il est caractérisé par le gain statique **:**

$$K = \frac{lim_{t\to\infty} s(t)}{lim_{t\to\infty} e(t)} | e(t) = constante$$

1.4.4. Le modèle d'un système :

Lorsque l'on envisage de commander et de contrôler un système, la première étape consiste à le modéliser.

Modéliser un système consiste à élaborer une représentation mathématique qui permette de décrire et prédire son comportement dynamique et permanent lorsqu'il est soumis à des influences externes telles que des entrées de commande, consignes et perturbations.

Ceci dit, et on le rappelle : tout système physique peut être représenté selon quatre manières (Figure 5).

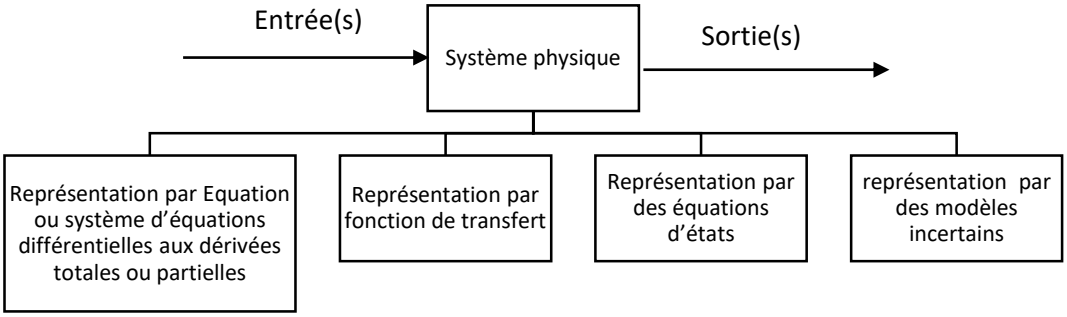


Fig. 5: Les différentes formes de modélisation d'un système physique

1.4.5. Système linéaire (non linéaire)

Un système est **linéaire** s'il satisfait au principe de superposition. Dans le cas contraire il est dit non linéaire.

1.4.6 Système causal :

Un système est **causal** si sa sortie s*(t)* à un instant t_0 ne dépend que des valeurs de son entrée e*(t)* pour $t \leq t0$. Un système causal ne répond pas avant d'être excité. C'est un système non anticipatif. Tous les systèmes physiques temporels réalisables sont *causaux*.

1.4.7. Système à temps invariant :

Un système à **temps invariant** a un modèle identique à tout instant : un retard τ ne change pas la loi du modèle.

1.4.8. Système instantané ou statique et système dynamique

Un système est dit **instantané si** à un instant donné sa sortie ne dépend que de l'excitation à cet instant : s*(t) = a.e(t).* Un tel système réagit donc instantanément, sans retard, sans régime transitoire ou temps d'établissement. Il est sans mémoire puisque le passé n'influence pas sa sortie présente. Un exemple de tel système est la résistance électrique idéale (Figure 6).

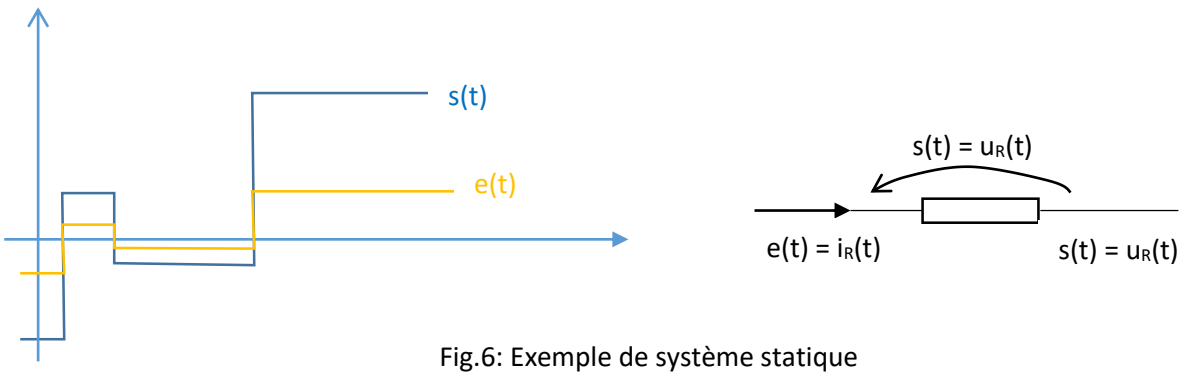


Fig.6: Exemple de système statique

Du point de vue de l'automaticien, un tel système peut être décrit par son gain statique. Dans tous les autres cas, il est dit à mémoire ou **dynamique**, par exemple : $s(t) = a.e(t-\tau)$ ou $s(t) = a.e(t) + b.s'(t)$.

Un tel système, a sa sortie qui peut dépendre non seulement de l'entrée présente mais aussi des entrées, éventuellement des sorties passées.

Un exemple de tel système est la capacité électrique (Fig.7). En définissant le courant de charge ic(t) comme signal d'entrée et la tension uc(t) aux bornes de C comme signal de sortie., on a:

$$s(t) = u_c(t) = \frac{1}{C}\int_{-\infty}^{t} i_c(\tau).d\tau = \frac{1}{C}\int_{-\infty}^{t} e(\tau)\,.d\tau$$

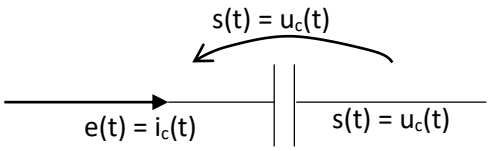
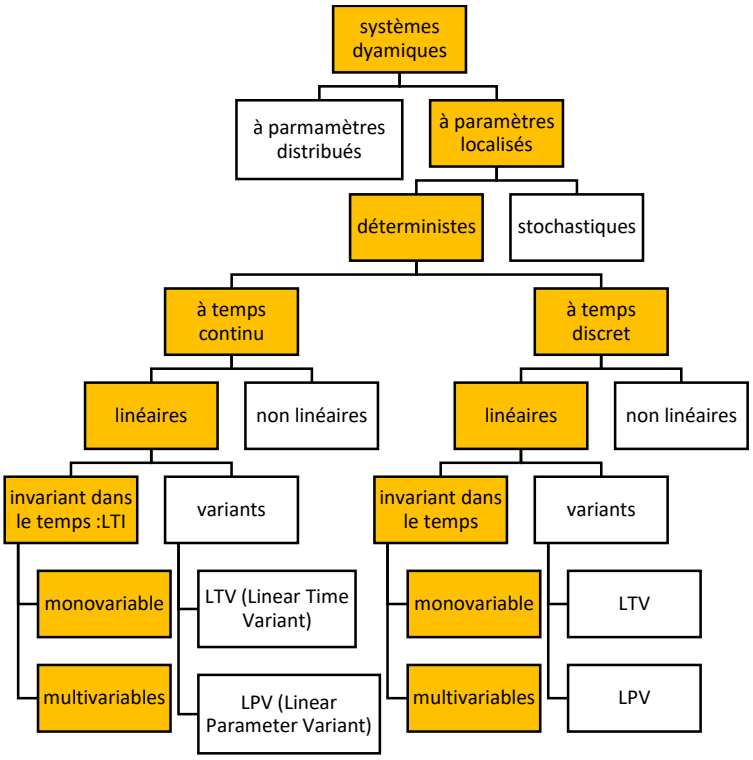


Fig.7 : Exemple de système dynamique

1.5. Classification des systèmes dynamiques

Les systèmes dynamiques peuvent être classés en plusieurs groupes (Fig. 8).



Fig…8 : Les différents types de systèmes dynamiques

1.6. Modèle d'un système dynamique

Un système dynamique est mathématiquement représentable par un système d'équations différentielles d'ordre 1, linéaires ou non comme suit :

$$\frac{dx1(t)}{dt} = f1\big(x1(t), \dots xn(t)\big) + g1\big(e(t)\big)$$

$$\frac{dx2(t)}{dt} = f2\big(x1(t), \dots xn(t)\big) + g2\big(e(t)\big)$$

$$\dots.$$

$$\frac{dxn(t)}{dt} = f2\big(x1(t), \dots xn(t)\big) + gn\big(e(t)\big)$$

$$s(t) = h\big(x1(t), \dots xn(t)\big) + d(e(t))$$

Les xi sont les variables internes (variables d'état) du système, e(t) l'entrée et s(t) la sortie.

Dans le cas où les paramètres du système sont constants, le système est dit à constantes localisées et les équations différentielles sont des équations différentielles aux dérivées totales. Dans le cas contraire, on a affaire à un système à paramètres distribués et sa représentation est faite à l'aide d'un système d'équations aux dérivées partielles.

1.6.1. Systèmes dynamiques linéaires à constantes localisés

C'est un système que l'on peut représenter par un système d'équations différentielles linéaires d'ordre 1 à coefficients constants.

1.6.2. Systèmes dynamiques non linéaires :

Les systèmes physiques (réels) ne sont pas nécessairement linéaires. Il est néanmoins souvent possible de les étudier avec les outils classiques de l'automatique linéaire après avoir linéarisé leur comportement autour d'un point de repos (façon de procéder familière aux électroniciens).

1.6.3. Systèmes dynamiques linéaires d'ordre supérieur à deux.

Même s'il existe des comportements dynamiques de systèmes automatiques très variés, une très grande partie d'entre eux, peut être décrite en utilisant des systèmes de 1er et 2nd ordre complétés par un retard.

1.7. Le modèle d'un moteur à courant continu

Un moteur électrique à courant continu est un système dynamique SISO, régit par des équations physiques qui découlent de ses caractéristiques électriques, mécaniques et magnétiques :

$u(t) = e(t) + R.i(t) + L.\frac{di(t)}{dt}$ — u : Tension appliquée au moteur

$e(t) = K_e.\omega(t)$ — e : Force électromotrice tension créée par le bobinage du moteur pour s'opposer à la variation du flux du champ magnétique le traversant

$c_m(t) = K_c.i(t)$ — i : Intensité du courant traversant le moteur

$c_m - c_r = J_T.\frac{d\omega(t)}{dt}$ — ω : Vitesse de rotation du rotor du moteur

c_m : Couple moteur généré
c_r : Couple résistant

Ce qui implique :

$$\Omega_m(P) = \frac{K_c}{K_eK_c+RJ_TP+LJ_TP^2}U(P) - \frac{R+LP}{K_eK_c+RJ_TP+LJ_TP^2}C_r(P)$$

Généralement l'inductance interne L du moteur est négligeable devant sa résistance interne d'où le modèle du MCC est un modèle du second ordre qui s'apparente à un modèle du premier ordre.

Soit :

$$\frac{\Omega_m(P)}{U(P)} = \frac{A}{1+\frac{2\varepsilon}{\omega_0}P+\frac{1}{\omega_0^2}P^2} \approx \frac{A}{1+\frac{2\varepsilon}{\omega_0}P}$$

Avec :

$A = \frac{1}{K_e}$ ----------→ gain statique

$\varepsilon = \frac{R}{2}\sqrt{\frac{J_T}{K_eK_cL}}$ ----------→ facteur d'amortissement

$\omega_0 = \sqrt{\frac{K_eK_c}{LJ_T}}$ ----------→ Pulsation propre

1.8. Les différentes techniques de commande des systèmes asservis (régulés)

On en donne à la Figure 9 ci-dessous une classification avec quelques exemples afin de situer celle que nous avons utilisée dans notre travail: commande par PID en boucle unique.

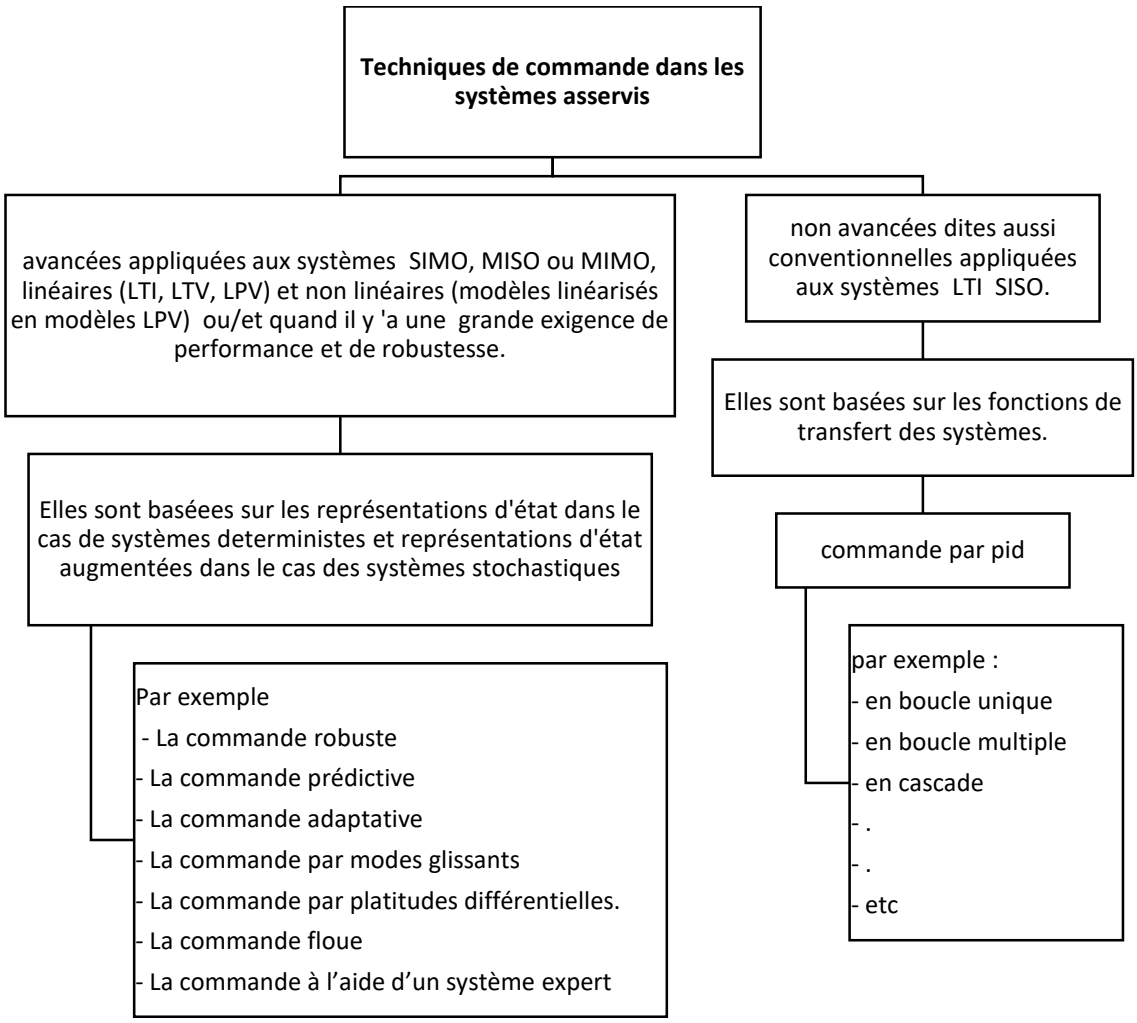


Fig. 9 : Classification des techniques de commande dans les asservissements

1.9. Objectif d'un asservissement ou régulation

On le rappelle, en automatique l'objectif d'un asservissement (ou régulation) est de corriger le comportement d'un système. On donne aux deux figures 10 et 11 suivantes et pour illustrer, un exemple de réponses avant et après correction d'un système SISO.

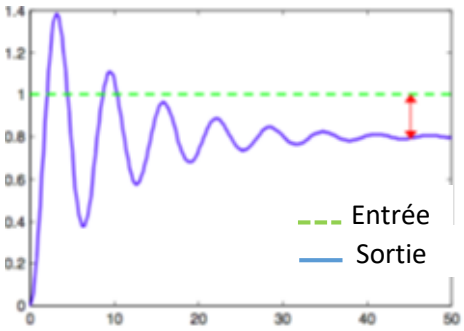


Fig. 10 : Système à commander avec réponse oscillatoire, non bien amortie et écart avec l'entrée en régime établi ou permanent.

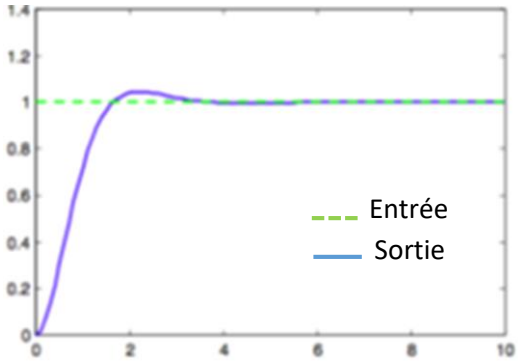


Fig11 : Système à commander avec réponse oscillatoire, après correction, bien amortie et écart nul avec l'entrée en régime établi.

1.10. Qualités ou performances d'un asservissement.

Ces qualités sont : la stabilité, la précision, la rapidité et la robustesse

1.10.1. Stabilité des systèmes asservis

Un système asservi est stable si à une entrée bornée correspond une sortie bornée (Fig. 12), (Fig.13).

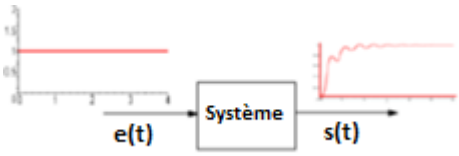


Fig.12 : Réponse du système stable

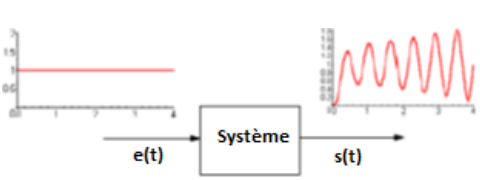


Fig.13 : Réponse du système instable

Autrement dit, un système asservi est stable s'il retourne vers son état d'équilibre lorsqu'il en est écarté. Il est instable dans le cas contraire.

1.10.2. Précision d'un système asservi

La **précision** s'exprime par l'écart « e » entre la consigne *yc* et la sortie *y* du système asservi.

1.10.3. La rapidité d'un système asservi

Pour ce qui est de la **rapidité**, on cherche à obtenir par les asservissements, une réponse rapide aux variations de la consigne et une aptitude à effacer rapidement les perturbations.

En pratique, on évalue la rapidité d'un système avec un **temps de réponse à 5%.** Ce dernier exprime le temps mis par la sortie du système soumis à un échelon pour atteindre sa valeur de régime permanent à ±5% près **et y rester (Fig.14).**

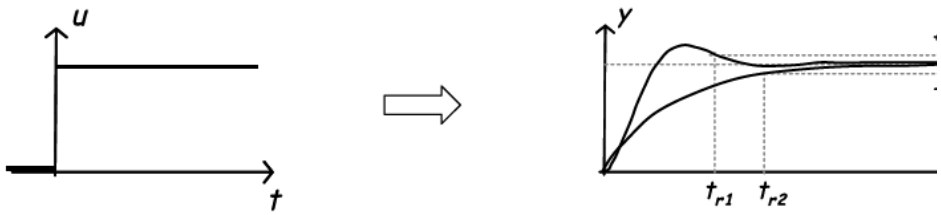


Fig. 14. : Illustration de la mesure du temps de réponse à 5% pour un système du 1er ordre et un système du 2ème ordre

Le temps de réponse est lié à l'inertie du système, c'est-à-dire à ses propriétés physiques.

1.10.4. La robustesse d'un système asservi

Un système asservi est dit robuste, s'il a la capacité de conserver ses performances (stabilité, rapidité et précision) même en présence de perturbations agissant sur le système ou variations de ses paramètres.

Chapitre 2 : Le contrôleur PID dans les systèmes d'asservissement

2.1. La régulation et l'asservissement par PID

C'est une méthode de commande, souvent employée pour des asservissements ou des régulations de systèmes SISO.

On rappelle aux figures 15 et 16 suivantes, respectivement les asservissements (régulations) par pid analogique et par pid numérique en boucle unique.

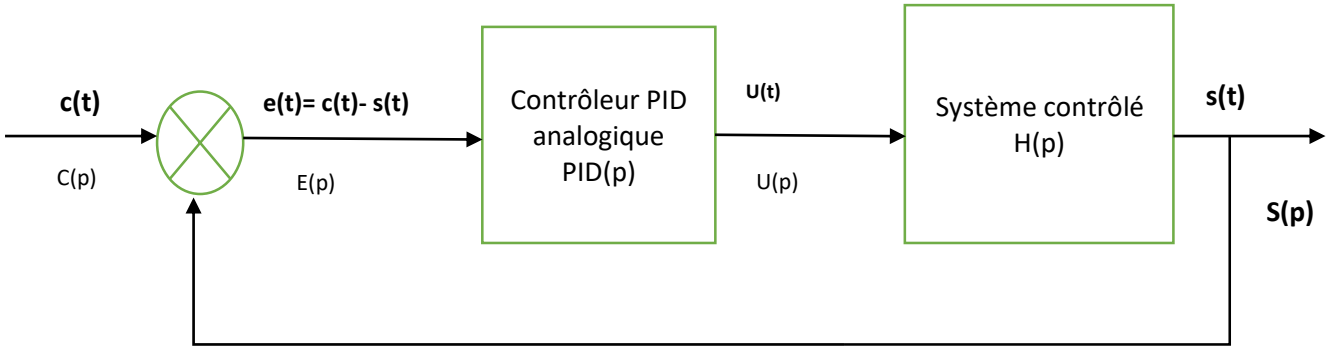


Figure 15. : asservissement (régulation) continu commandé par un signal analogique et correction avec pid analogique

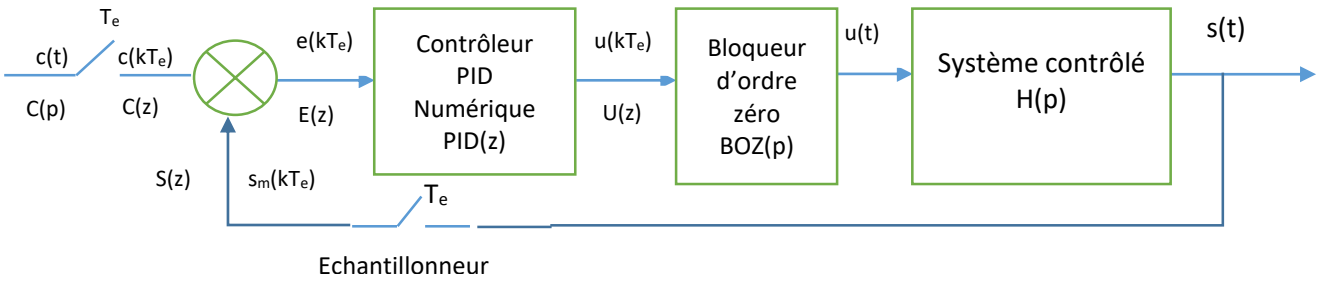


Figure …16 : asservissement (régulation) continu commandé par un signal échantillonné et correction avec pid numérique

2.3. P

P, I et D constituent des correcteurs dont l'association permet d'assurer un asservissement ou une régulation optimale d'un système LTI.

2.2.1. Rôle de l'action P :

Le terme Proportionnel permet d'augmenter la vitesse de montée de la réponse d'un système (atteint la consigne le plus rapidement possible) ou autrement dit, diminuer le temps de montée de la réponse du système commandé (le moteur dans notre cas). Il permet de vaincre les grandes inerties du système en lui donnant de la puissance.

Le contrôle de type P applique une correction proportionnelle à l'erreur corrigeant de manière instantanée tout écart de la grandeur à régler :

$$commande(t) = K_p \varepsilon(t)$$

Lorsque l'on augmente Kp, le système réagit plus vite et l'erreur statique s'en trouve améliorée, mais en contrepartie le système perd en stabilité. Le dépassement se fait de plus en plus grand, et le système peut diverger dans le cas d'un Kp démesuré.

On donne à la Figure 17 ci-dessous son impact sur la réponse d'un système dans le cas d'une régulation.

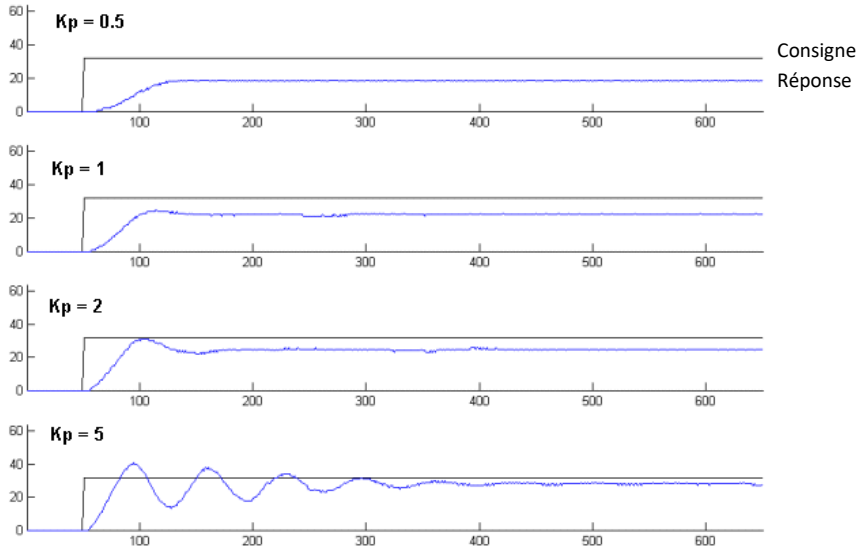


Fig. 17 : Effet de Kp sur la réponse d'un système en régulation (temps de montée de la réponse ou réactivité du système améliorée, erreur statique améliorée, stabilité diminuée)

2.2.2. Rôle de l'action I

ElIe complète l'action P en réduisant voire annulant l'erreur statique de l'action P, sans modifier la réaction du régulateur PID. L'idée est d'intégrer l'erreur depuis le début et d'ajouter cette erreur à la commande : lorsque l'on se rapproche de la valeur demandée, l'erreur devient de plus en plus faible. Le terme proportionnel n'agit plus mais le terme intégral subsiste et reste stable, ce qui maintient la sortie du système à la valeur demandée dans le cas par exemple d'une régulation de vitesse d'un moteur à courant continu :

$$\boldsymbol{commande(t) = K_p.\varepsilon(t) + K_i.\int_0^t \varepsilon(\tau)\,.d\tau}$$

L'intégrale agissant comme un filtre sur le signal intégré, elle permet aussi de diminuer l'impact des perturbations (bruit, parasites), et il en résulte alors un système plus stable. Malheureusement, un terme intégral trop important peut lui aussi entraîner un dépassement de la consigne, une stabilisation plus lente, voire même des oscillations divergentes (Figure 18).

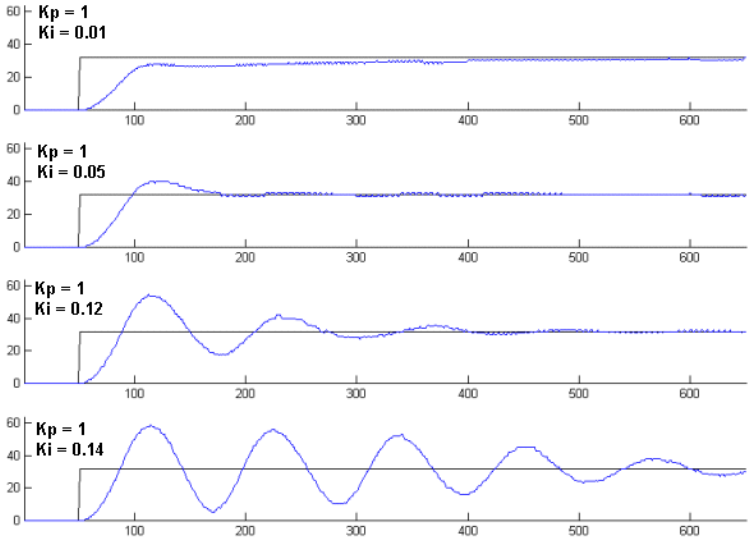


Fig. 18 : Effet de Ki sur la réponse d'un système en régulation (erreur statique améliorée, stabilité diminuée)

2.2.3. Rôle de l'action D

Le terme Dérivé agit sur le dépassement en le réduisant. Elle améliore donc la stabilité du système en amortissant rapidement les oscillations provoquées par une perturbation ou un changement brusque de la consigne :

$$\boldsymbol{Commande(t) = K_P.\varepsilon(t) + K_i.\int_0^t \varepsilon(\tau)\,d\tau + K_d.\frac{d}{dy}\varepsilon(t)}$$

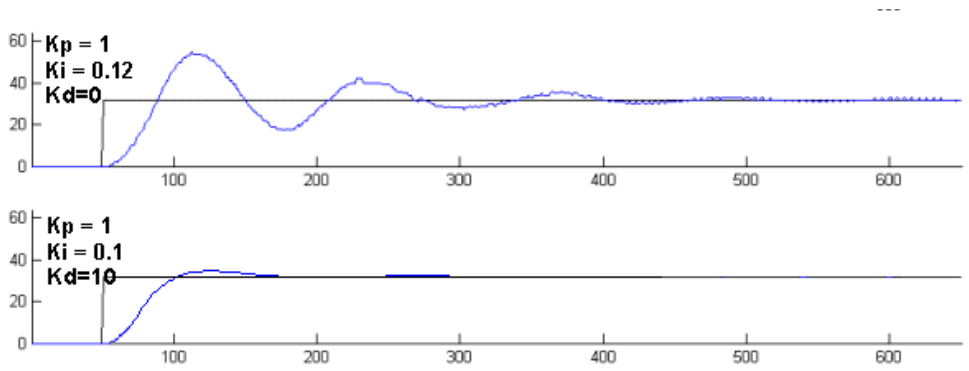


Fig. 19 : Effet de Kd sur la réponse d'un système en régulation (stabilité améliorée, dépassement diminué)

Remarque :

L'action dérivée est surtout utilisée dans le cas de grandeur à régler non bruitées, car la dérivation est très sensible au bruitage du signal : on diminuera donc son influence dans un asservissement où la grandeur à régler est soumise à de nombreuses perturbations (e.g : vitesse d'un moteur) .

Pour récapituler on donne au tableau 1 suivant un résumé des effets combinés des trois actions P, I et D :

Paramètre	Temps de montée	Temps de stabilisation	Dépassement	Erreur statique
Kp	**Diminue**	**Augmente**	**Augmente**	**Diminue**
Ki	**Diminue**	**Augmente**	**Augmente**	**Annule**
Kd	-	**Diminue**	**Diminue**	-

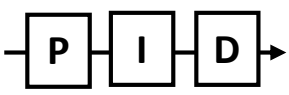
2.3. Les limites du contrôle par PID

Le contrôle par PID convient parfaitement pour les systèmes qui ont un dynamique très simple (système de 1er ordre). Il est inadapté dès que les systèmes sont extrêmement dynamiques (systèmes de second ordre).

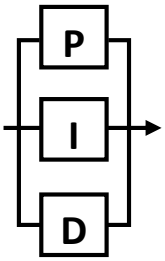
2.4. Les différentes associations possibles des actions P, I et D

Il existe plusieurs façons d'associer les actions P, I et D.

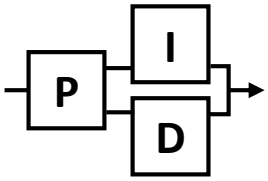
➔ P.I.D série :



➔ P.I.D parallèle :



➔ P.I.D mixte:



2.5. Les différents types d'implémentations d'un PID

Le régulateur pid est passé par tout une série de versions (Fig. 20) allant du pneumatique à l'intégré. Et aujourd'hui, presque toutes les applications des contrôleurs PID sont sous forme numérique.

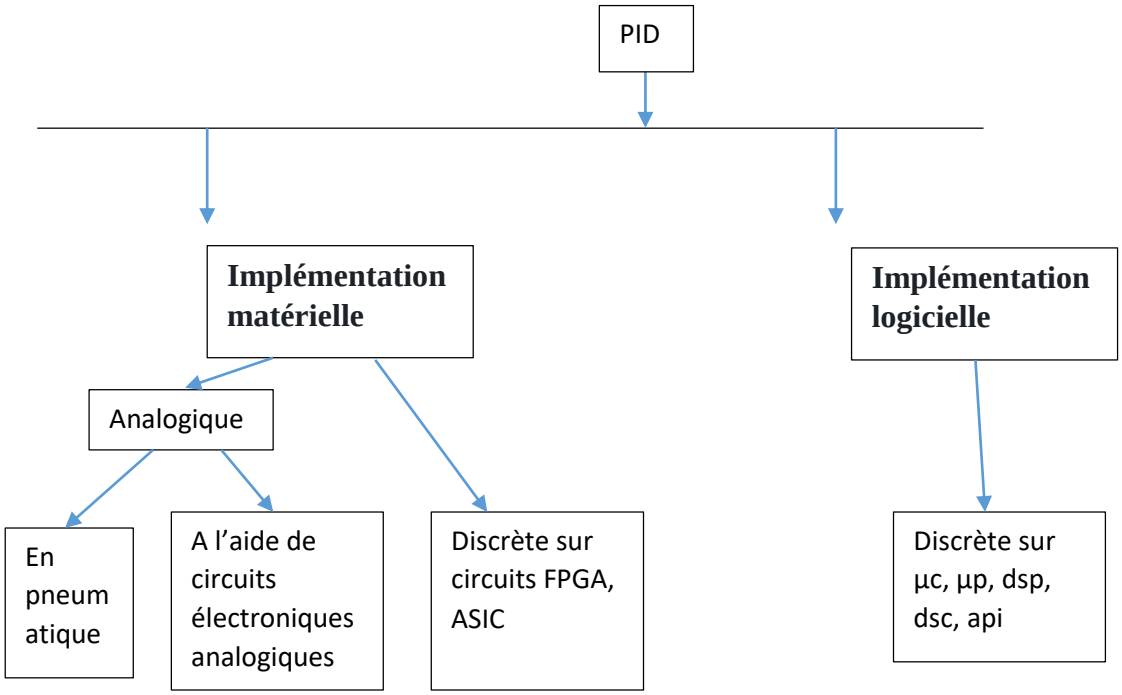


Fig. 20 : les diverses implémentations d'un PID

2.5.1. Implémentation matérielle analogique ou continue en technologie pneumatique

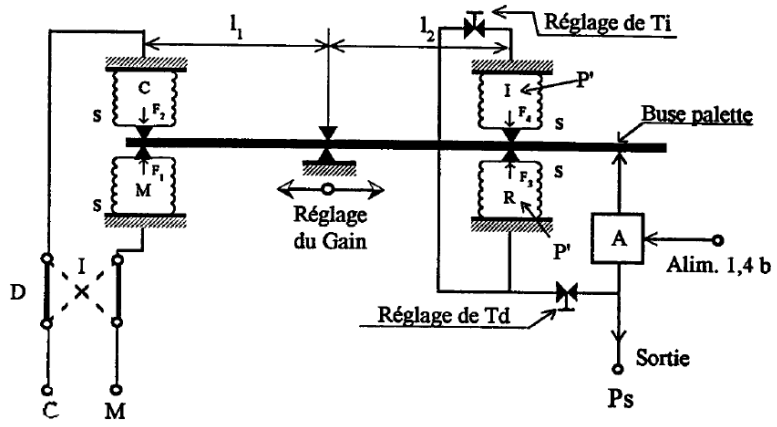


Fig.21: Exemple de PID à équilibre de moments de forces, implémenté en technologie pneumatique.

2.5.2. Implémentation matérielle analogique à l'aide de circuits électroniques analogiques

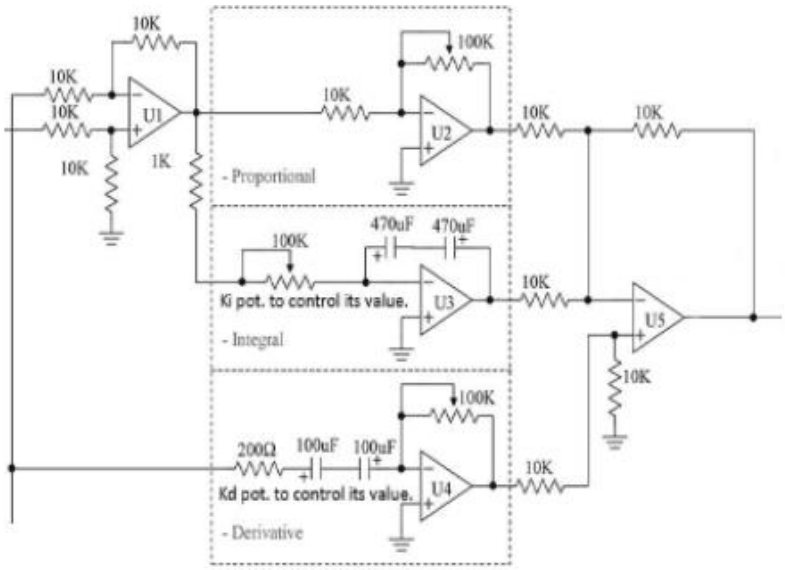


Fig. 22 : Exemple de PID réalisé à l'aide d'AOp

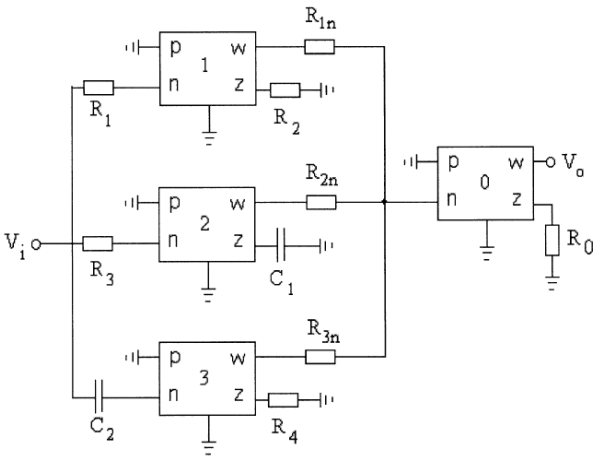


Fig. 23 : Exemple de PID réalisé à l'aide de circuits CDBA(current differencing buffered amplifier)

Utilisé en nanotechnologies et en technologie des MEMS(micro-electromechanical systems)

2.5.3. Implémentation matérielle discrète sur circuit FPGA et sur ASIC

Les FPGA on le rappelle, permettent l'informatique reconfigurable flexible comme celle réalisée par les programmes informatiques. Le résultat de ce type d'implémentation est un algorithme PID câblé, constitué d'un ensemble de blocs CLB (Configurable Logic Bloc) configurés chacun pour exécuter une opération ou une fonction du PID, et interconnectés au sein du FPGA à l'aide d'une hiérarchie d'interconnexions reconfigurables. D'où en pratique ce type d'implémentation est aussi qualifié d'implémentation matérielle sur FPGA. Ceci dit, pour pouvoir procéder à une telle implémentation, il faut d'abord développer la forme digitale ou discrète du PID. On donne à la figure 24, à titre illustratif, un exemple de PID câblé sur un FPGA.

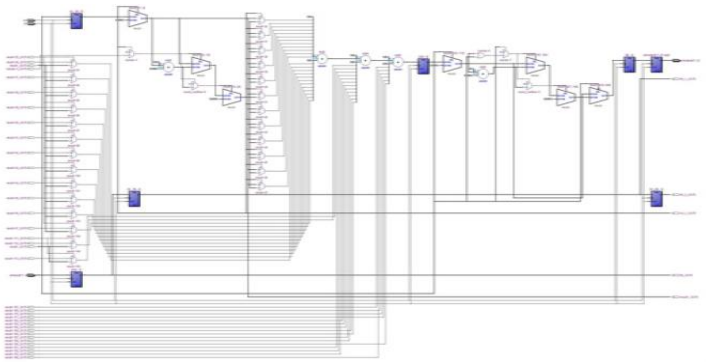


Fig. 24 : Exemple de PID implanté sur FPGA

2.5.4. Implémentation logicielle discrète sur µp/µc ou dsp/dsc

Dite aussi implémentation software, elle est constituée d'un algorithme PID, codé avec un langage de programmation de haut niveau, le plus souvent le langage c/c++ et exécuté sur du matériel tel qu'un µc, DSP ou DSC.

Tout comme pour les fpga, une telle implémentation s'appuie sur la forme numérique du PID. On donne à la figure 25, à titre illustratif, un exemple d'algorithme PID que l'on peut exécuter par exemple sur un µc.

```
previous_error := 0
integral := 0
loop:
    error := setpoint − measured_value
    integral := integral + error × dt
    derivative := (error − previous_error) / dt
    output := Kp × error + Ki × integral + Kd × derivative
    previous_error := error
    wait(dt)
    goto loop
```

Fig. 25 : Exemple de PID implanté sur µc

2.5.5. Implémentation logicielle discrète sur automate programmable (api ou plc)

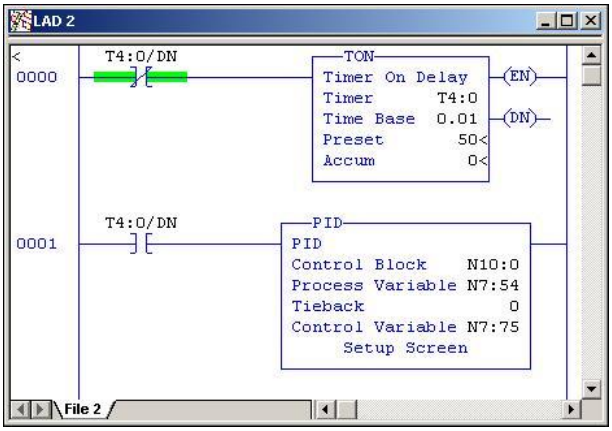


Fig. 26 : Exemple de PID implanté sur API

Chapitre 3 : Synthèse d'un contrôleur PID numérique

1.Méthodes de synthèse

On le rappelle, l'obtention de la forme discrète d'un régulateur (pid inclus) peut se faire selon trois manières (Fig.27).

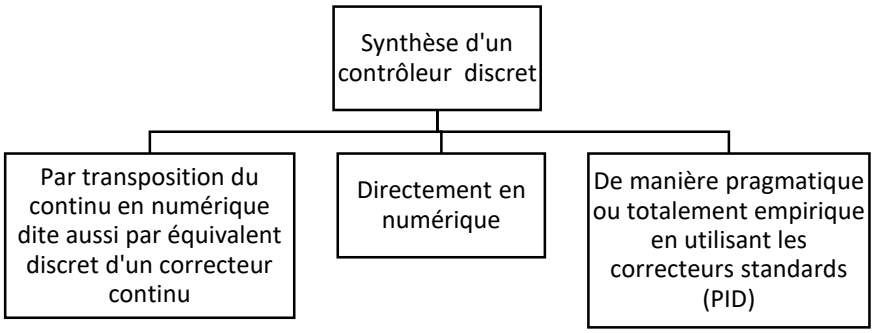


Figure 27 Les trois méthodes de synthèse de la forme discrète d'un contrôleur pid

Dans les paragraphes qui suivent, nous allons les rappeler de manière succincte.

Dans ce projet, et étant donné les circonstances dans lesquelles a été réalisé ce dernier, nous avons utilisé pour des raisons de simplicité la troisième (l'empirique).

2. Méthodes de synthèse par équivalent discret d'un correcteur continu

Fort utilisée en pratique et simple à appliquer, cette méthode de synthèse, ne nécessite presque aucune connaissance des systèmes échantillonnés.

On rappelle à la figure 28 ci-dessous les étapes de cette approche.

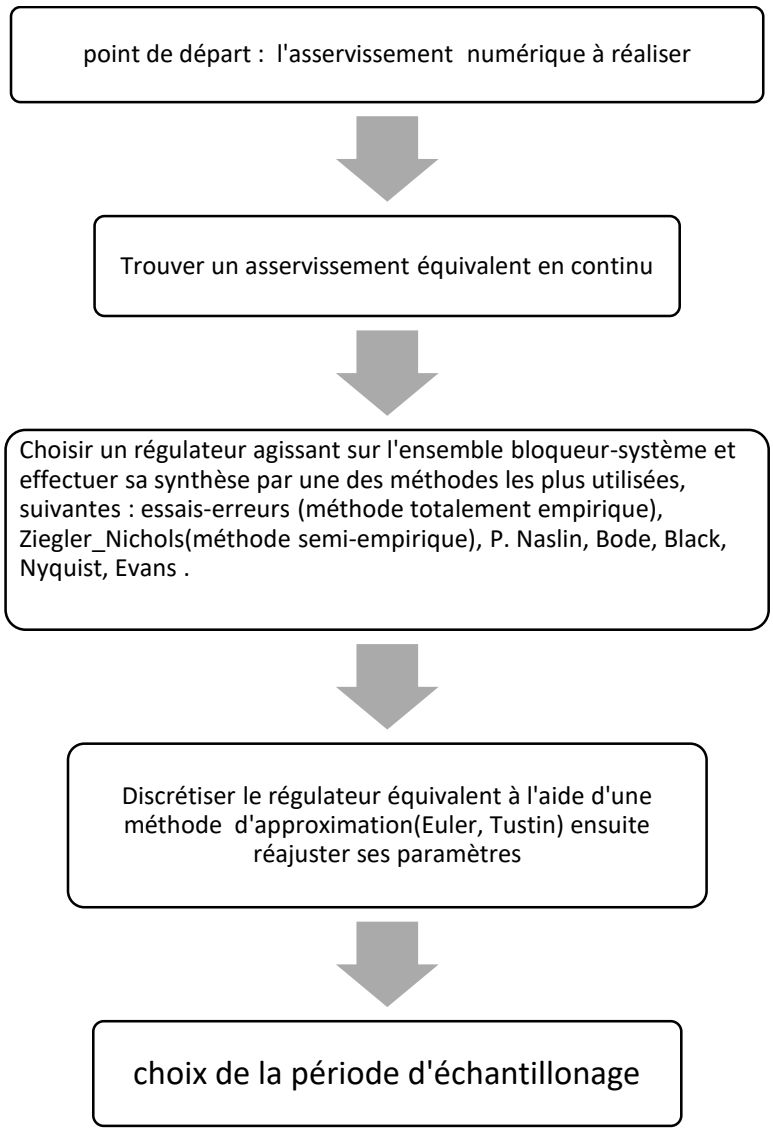


Fig. 28: étapes de synthèse d'un régulateur numérique par transposition du continu en numérique

Etape 1 : aspect général d'un asservissement numérique à réaliser:

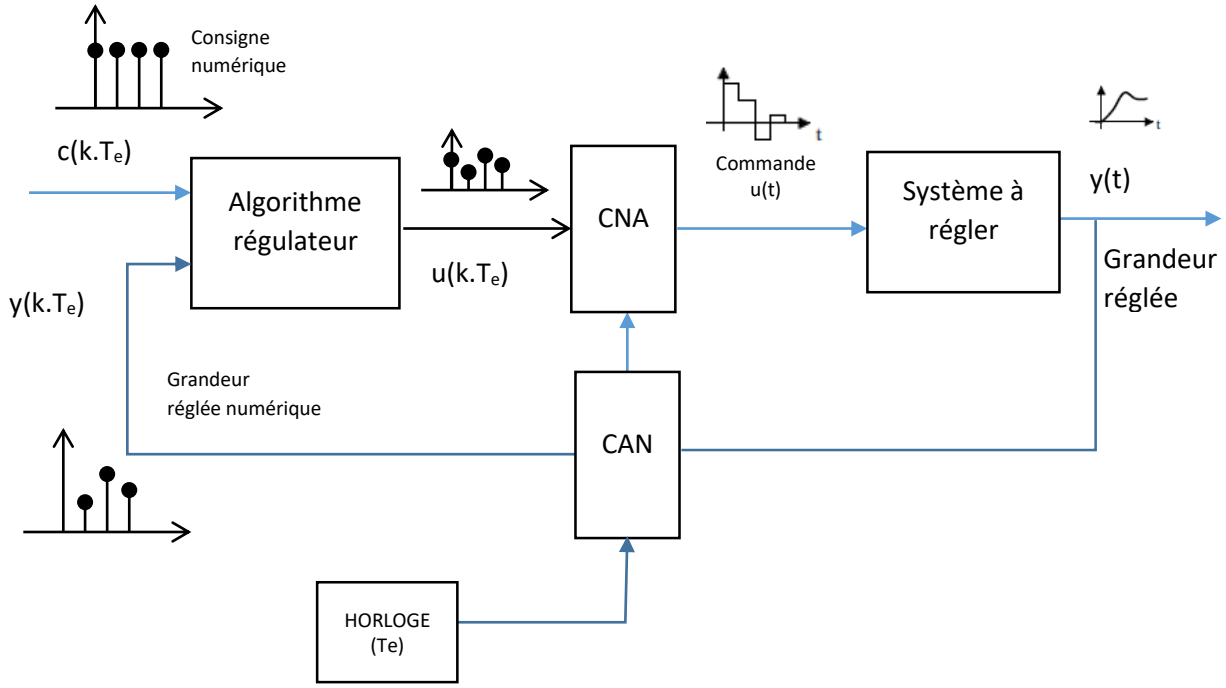


Fig.29: schéma fonctionnel général d'un asservissement numérique

Etape 2 : Le système d'asservissement équivalent en continu

Pour trouver un asservissement continu équivalent, il faut d'abord modéliser les deux convertisseurs.

<u>Modélisation du CAN</u>

On le rappelle, un CAN peut être schématisé sous la forme suivante (Fig.30) :

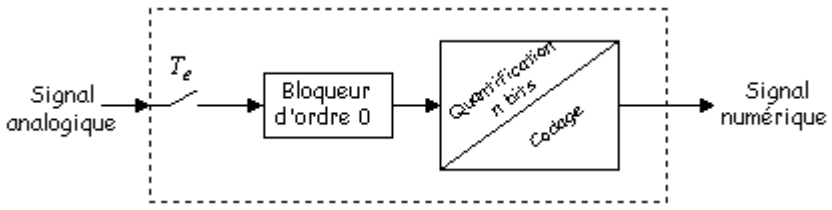


Fig.30: schéma fonctionnel d'un convertisseur analogique-numérique (CAN)

Etant donné qu'en pratique, l'ensemble échantillonnage-numérisation est très rapide , ce schéma peut être simplifié sous la forme d'un simple échantillonneur (Fig.31).

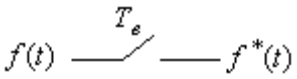


Fig. 31: modèle d'un CAN

Modélisation du CNA

De même, un CNA peut être schématisé sous la forme suivante (Fig.32) :

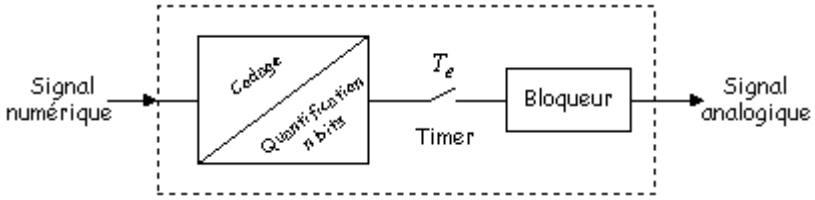


Fig32:schéma fonctionnel d'un convertisseur numérique-analogique (CNA)

L'on peut simplifier ce modèle par un simple bloqueur d'ordre zéro (Fig.33) .

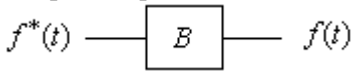


Fig.33: modèle d'un CNA

D'où l'asservissement en continu équivalent pour la synthèse d'un régulateur équivalent discret:

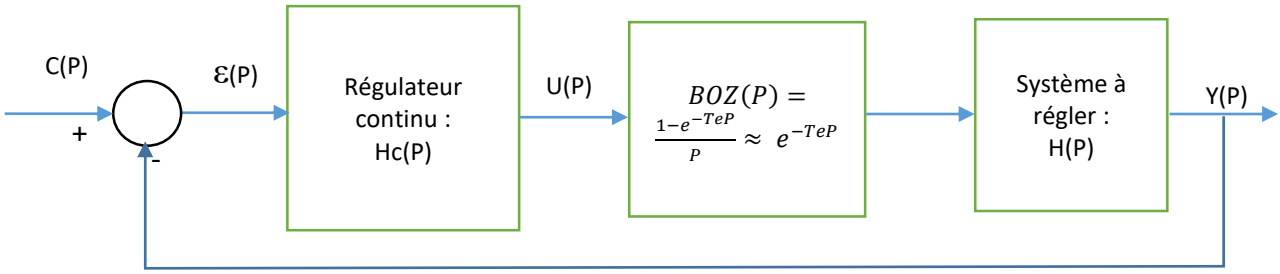


Fig.34 : Equivalent en continu, d'un système d'asservissement numérique

On le rappelle : le bloqueur BOZ peut être assimilé à un retard pur pour les fréquences d'échantillonnage élevées, ce qui implique une fonction de transfert dans le domaine de Laplace égale à e^{-PTe} où Te est la période d'échantillonnage.

Etape 3 : Synthèse en continu d'un régulateur analogique équivalent : les exigences

Dite aussi réglage, elle consiste à déterminer les coefficients ou paramètres du régulateur afin d'obtenir une réponse adéquate du système et de la régulation ou asservissement. Les objectifs sont d'être **stable**, **rapide**, **précis et robuste**

Le point de départ des différentes méthodes de synthèse citées ci-dessus, est un cahier des charges dans lequel sont décrites les exigences que doit satisfaire le système de régulation ou d'asservissement (**Fig.35)**.

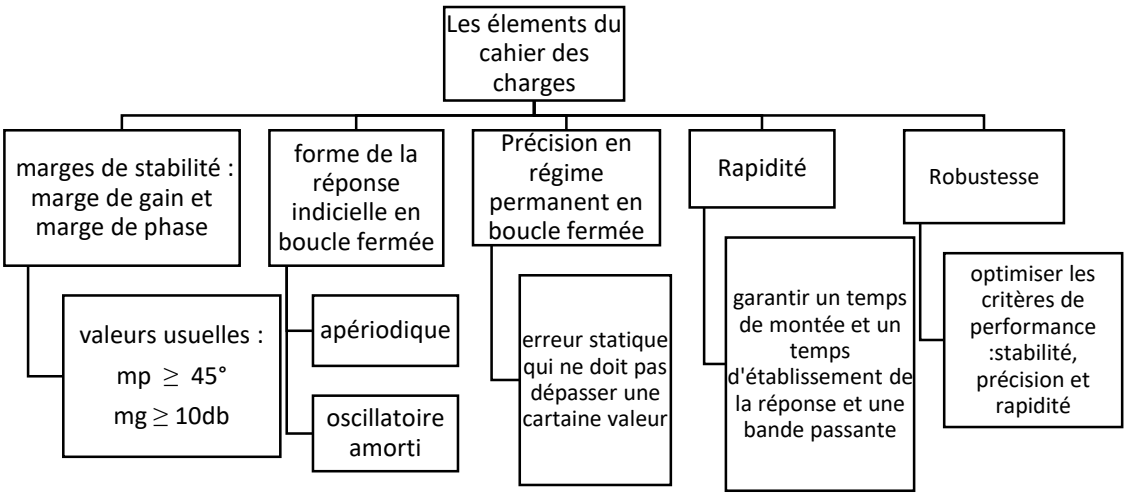


Fig.35: Les exigences pour la synthèse d'un régulateur d'un asservissement

Etape 4 : calcul d'une version discrète

Dans cette étape, on revient en numérique et on calcule une version discrète du régulateur analogique de l'étape 3 (Fig.36).

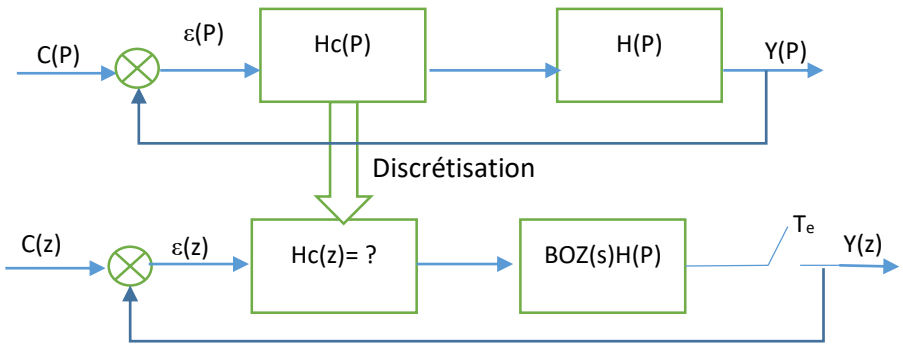


Figure36 : Synthèse du pid discret par transposition du pid analogique

Ce calcul peur être effectué en utilisant les méthodes d'approximations d'Euler (différences vers l'avant), d'Euler (différences vers l'arrière) et de Tustin ,

Dans les paragraphes qui suivent, nous allons brièvement rappeler le principe de fonctionnement de ces méthodes.

Selon ces méthodes, la variable P de la fonction de transfert du régulateur analogique (dans le domaine de Laplace), est approchée par :

$$P = \frac{2}{T_e}\frac{z-1}{z+1} = \frac{2}{T_e}\frac{1-z^{-1}}{1+z^{-1}}$$ si Tustin

ou $P = \frac{z-1}{T_e} = \frac{1-z^{-1}}{T_e z^{-1}}$ si Euler (vers l'avant)

ou $P = \frac{z-1}{T_e z} = \frac{1-z^{-1}}{T_e}$ si Euler (vers l'arrière)

Ces approximations sont obtenues on le rappelle, en linéarisant la relation liant z à P, z étant la variable dans le domaine discret (transformée en z) et P la variable dans le domaine continu de Lapalce (transformée de Laplace) comme suit :

La relation liant z à P permettant de basculer entre ces deux domaines est égale à :

$z = e^{PT_e} \Rightarrow P = \frac{1}{T_e}\ln(z)$ qui est une relation non linéaire qu'on linéarise en effectuant les approximations suivantes :

Dans le cas de Tustin, on a :

$$z = \frac{e^{\frac{PT_e}{2}}}{e^{\frac{-PT_e}{2}}} \approx \frac{1+\frac{T_e}{2}P}{1-\frac{T_e}{2}P} \Rightarrow P = \frac{2}{T_e}\frac{z-1}{z+1} = \frac{2}{T_e}\frac{1-z^{-1}}{1+z^{-1}}$$

Dans le cas d'Euler (vers l'avant), on a :

$$z = e^{PT_e} \approx 1 + PT_e \Rightarrow P = \frac{z-1}{T_e} = \frac{1-z^{-1}}{T_e z^{-1}}$$

Et dans le cas d'Euler (vers l'arrière), on a :

$$z = \frac{1}{e^{-PT_e}} \approx 1 - PT_e \Rightarrow P = \frac{z-1}{T_e z} = \frac{1-z^{-1}}{T_e}$$

D'où les versions numériques dans le cas d'un régulateur pid par les deux méthodes de transposition conseillées avec dérivée filtrée :

$PID(z) = K_p\left[1 + \frac{1}{T_i}\frac{T_e}{z-1} + \frac{N(z-1)}{z-(1-\frac{NT_e}{T_d})}\right]$ avec Euler arrière (la plus utilisée en pratique car plus simple).

$PID(z) = K_p\left[1 + \frac{T_e}{2T_i}\frac{z+1}{z-1} + \frac{N(z-1)}{(1+\frac{NT_e}{2T_d})z-(1-\frac{NT_e}{T_d})}\right]$ avec Tustin

Dans le domaine temporel, cela revient : :

- à approximer le terme dérrivée du pid,
    - par la formule des rectangles (Euler), comme une *différence vers l'arrière si Euler arrière ;*

$$\frac{de(t)}{dt} \approx \frac{e(t)-e(t-T_e)}{T_e}$$

$$Z\left\{\frac{e(t)-e(t-T_e)}{T_e}\right\} = \frac{1-z^{-1}}{T_e}E(z) = \frac{z-1}{zT_e}E(z)$$

D'où : $P \leftrightarrow \frac{z-1}{zT_e}$

    - par la formule des rectangles (Euler), comme une différence vers l'avant si Euler avant ;

$$\frac{de(t)}{dt} \approx \frac{e(t+T_e)-e(t)}{T_e}$$

$$Z\left\{\frac{e(t+T_e)-e(t)}{T_e}\right\} = \frac{z-1}{T_e}E(z)$$

D'où : $P \leftrightarrow \frac{z-1}{T_e}$

- à approximer l'intégrale du pid soit :
  - par la formule des trapèzes si Tustin ;

$I_k = \int_0^{kT_e} e(t)\,dt \approx I_{k-1} + T_e\left(\frac{e_k+e_{k-1}}{2}\right)$

D'où : $I(z)=\frac{2}{T_e}\frac{z-1}{z+1}e(z) = \frac{2}{T_e}\frac{1-z^{-1}}{1+z^{-1}}e(z)$

  - par la formule des rectangles (Euler), comme une *différence vers l'arrière si Euler arrière ;*

$I_k = \int_0^{kT_e} e(t)\,dt \approx I_{k-1} + T_e.e_k$

  - par la formule des rectangles (Euler), comme une *différence vers l'avant si Euler avant;* $I_k = \int_0^{kT_e} e(t)\,dt \approx I_{k-1} + T_e.e_{k-1}$

3. Méthode de synthèse pragmatique

Valable uniquement pour des régulateurs standards comme dans notre cas, cette méthode de synthèse est basée sur la discrétisation de l'équation différentielle ou de la fonction de transfert en P, du pid analogique choisi comme régulateur, sans passer par sa conception à l'aide de l'une des méthodes présentées ci-dessus, ensuite procéder à son réglage.

L'argument justifiant cette méthode est l'utilisation d'un taux d'échantillonnage suffisamment élevé. Elle consiste en trois étapes (Fig.37).

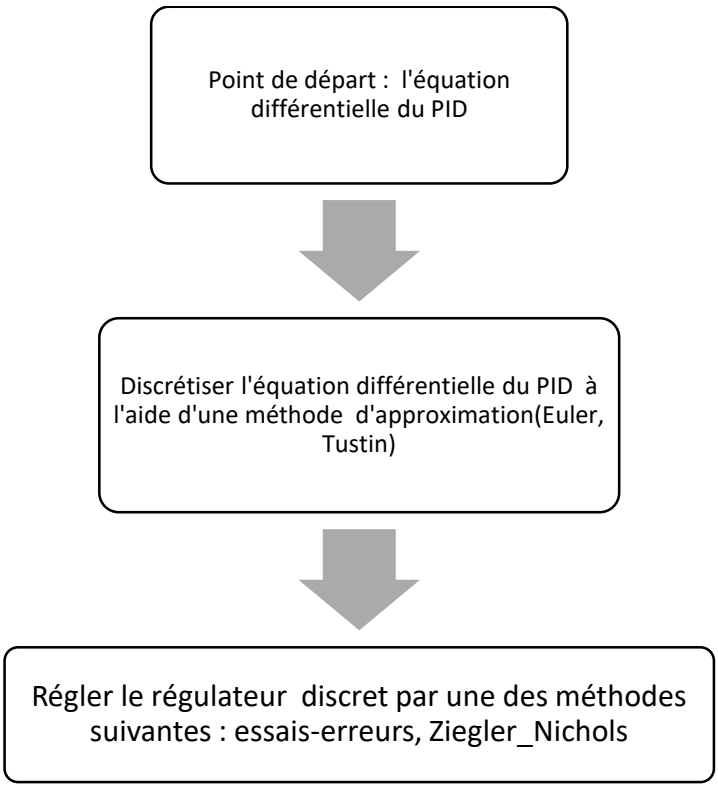


Fig.37: **étapes de synthèse d'un régulateur PID numérique par la méthode empirique**

4. Synthèse directe en numérique

On donne à la figure 38 suivante les différentes étapes de cette méthode :

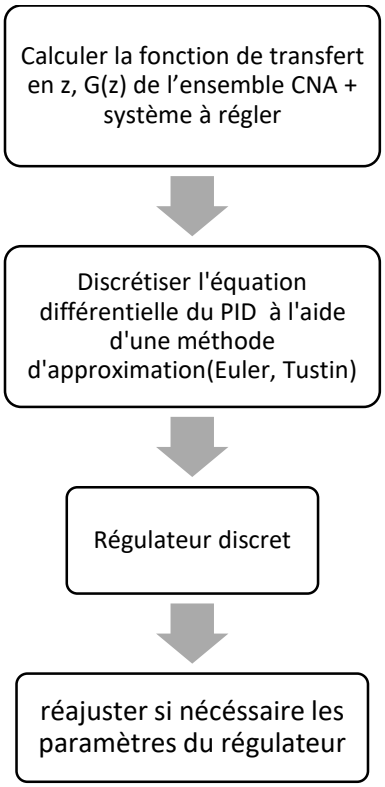


Fig.38: étapes de synthèse en numérique d'un régulateur discret

5. Les différentes formes discrètes de contrôleurs PID numériques

Deux formes discrètes peuvent être obtenues suite à la discrétisation de l'équation différentielle du PID :

La forme discrète dite de position et la forme discrète dite incrémentale.

5.1. La forme discrète de position

$$u(k) = K_p[e(k) + \frac{T_e}{T_i}\sum_{i=0}^{k-1} e(i) + \frac{T_d}{T_e}\big(e(k) - e(k-1)\big)]$$

5.2 La forme discrète incrémentale

$$\Delta \boldsymbol{u(k) = u(k) - u(k-1)}$$
$$\boldsymbol{= K_p\left[e(k) - e(k-1) + \frac{T_e}{T_i}e(k-1) + \frac{T_d}{T_e}\big(e(k) - 2e(k-1) + e(k-2)\big)\right]}$$
$$\boldsymbol{= q_0 e(k) + q_1 e(k-1) + q_2 e(k-2)}$$

Avec :

$$\boldsymbol{q_0 = K_p\left(1 + \frac{T_d}{T_e}\right)}$$
$$\boldsymbol{q_1 = -K_p\left(1 + \frac{2T_d}{T_e} - \frac{T_e}{T_i}\right)}$$
$$\boldsymbol{q_2 = -K_p\frac{T_d}{T_e}}$$

L'avantage majeur de cette forme est qu'elle est récursive. Elle permet de calculer la sortie incrémentale à chaque instant d'échantillonnage. Par conséquent, elle ne nécessite de stocker que les trois valeurs précédentes : e (k), e (k-1) et e (k-2). En plus, elle a plusieurs autres avantages.

Donc pour cette raison, c'est cette forme ci que nous avons utilisée dans notre projet.

Chapitre 4 : Réalisation matérielle et logicielle

1. Réalisation matérielle

1.1.Schéma bloc de l'asservissement réalisé

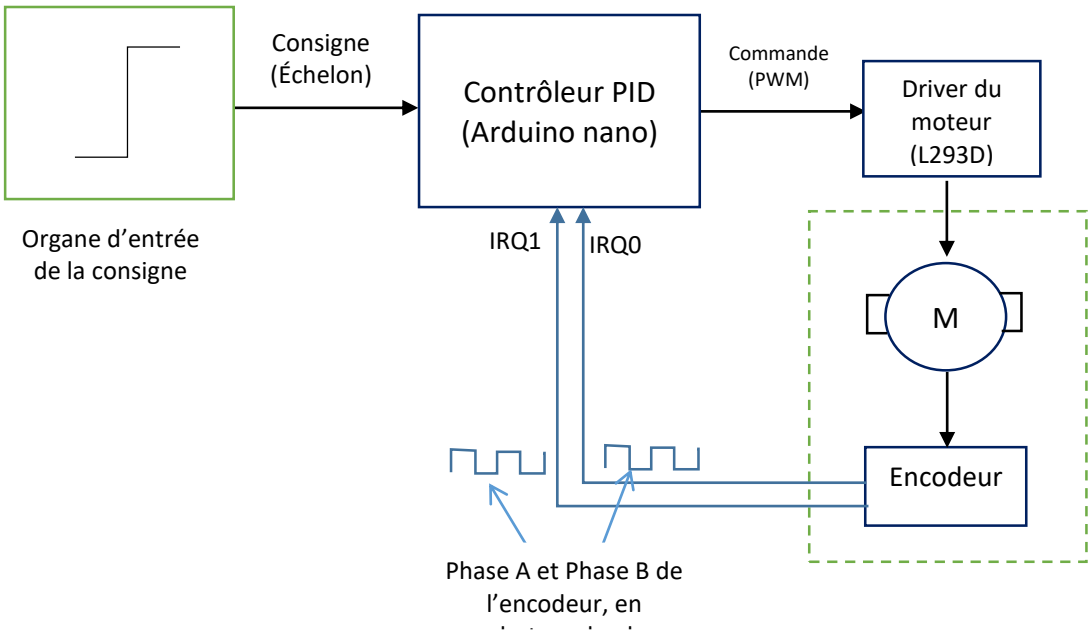


Fig.39: schéma synoptique du régulateur réalisé

Remarque : Les circonstances dans lesquelles ce projet a été fait (Covid-19) ne nous ont pas permis de le réaliser matériellement. Nous nous sommes alors limités à sa simulation sous proteus.

1.1.1. Description fonctionnelle

Le contrôleur PID évalue l'erreur entre la consigne et la position angulaire du moteur contrôlée et applique une correction PID à ce dernier sous la forme d'une impulsion PWM.

La position angulaire de sortie du moteur est mesurée par l'encodeur et transmise au contrôleur PID. Le contrôleur PID utilise l'algorithme de contrôle pour déterminer une nouvelle sortie (PWM) si nécessaire pour réduire l'erreur et une nouvelle sortie de l'encodeur permet de reprendre la boucle.

1.1.2. Le matériel utilisé pour la réalisation de l'asservissement

- Moteur à courant continu avec encodeur optique
- Plateforme arduino nano
- CI LD293D
- Afficheur LCD

Dans les paragraphes qui suivent nous allons nous limiter aux parties les plus importantes : le moteur à courant continu, l'encodeur optique incrémental et la plateforme arduino nano.

1.1.2.1. La commande des moteurs à courant continu

Un moteur à courant continu peut être commandé selon deux modes (Fig. 40).

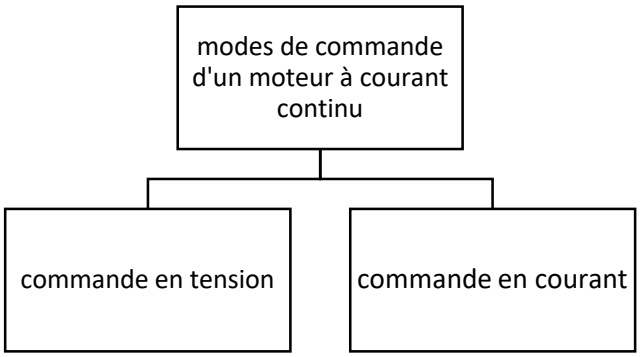


Fig.40: les modes de commande d'un moteur à courant continu

La commande en tension est utilisée avec les petits moteurs à courant continu et la commande en courant est utilisée avec les gros moteurs à courant continu.

Remarque :

Le moteur utilisé dans notre projet étant supposé (car simulé) un petit moteur, nous avons alors utilisé la commande en tension. Donc une régulation avec une boucle unique.

1.1.2.2. L'encodeur optique incrémental

Comme tout encodeur optique, c'est un type de capteur qui permet de mesurer la position angulaire ou la vitesse de rotation d'un arbre de moteur ainsi que son sens de rotation

Il est conçu avec un disque à segments translucides et opaques configurés pour laisser passer la lumière en certains endroits (Figure 41).

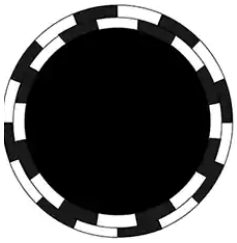


Fig.41: Exemples de disque optique. C'est un disque dit incrémental car les segments sont décalés de 90°.

En plaçant une LED et des photodiodes de part et d'autre du disque, les photodiodes détectent la lumière traversant le disque et génèrent des formes d'ondes impulsionnelles *avec une différence de phase de 90° entre eux,* correspondant aux formes translucides et opaques sur les segments du disque (Fig.42). De plus lorsque l'arbre du codeur tourne dans le sens horaire, la forme

d'onde A précède la forme d'onde B. Si le sens de rotation devient anti-horaire, le signal B précède le signal A.

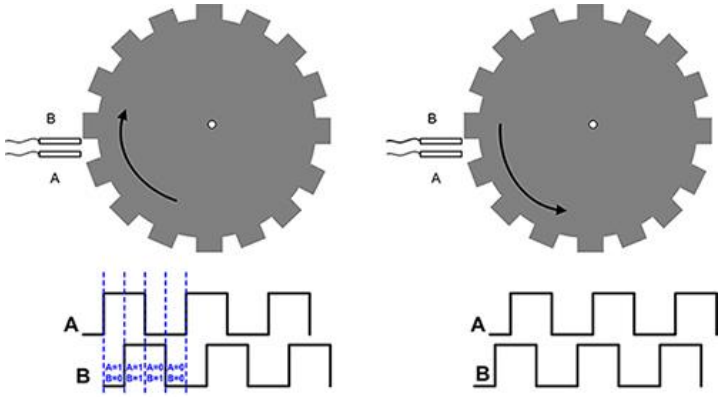


Fig. 42: L*es signaux en quadrature générés par un encodeur optique incrémental.*

1.1.2.3. La plate-forme Arduino nano

1.1.2.3.1. Spécifications

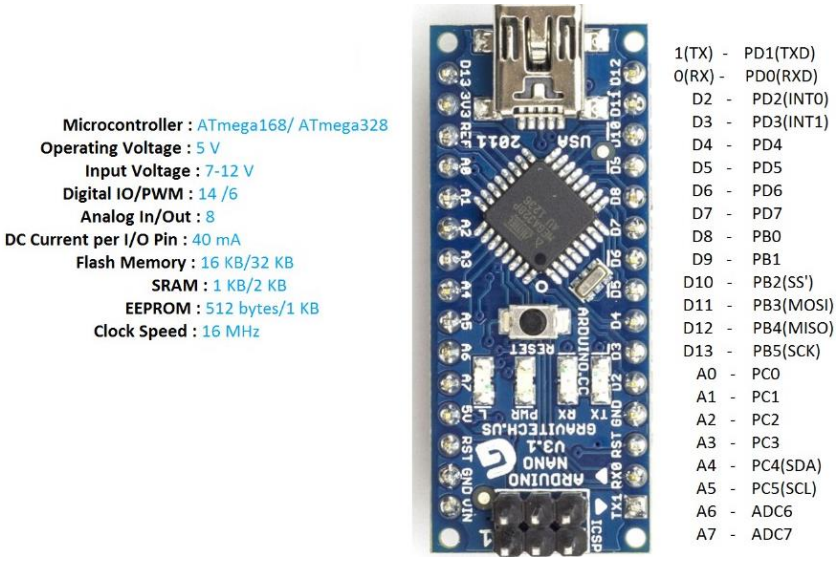


Fig.43: caractéristiques de arduino nano

1.1.2.3.2. La conversion numérique analogique utilisée dans arduino nano

En se référant au schéma bloc de la figure 29, pour pouvoir commander un système continu (moteur à courant continu dans notre cas), il faut convertir en analogique le signal de commande numérique à la sortie du régulateur. Pour ce faire, il existe deux techniques (Fig.44).

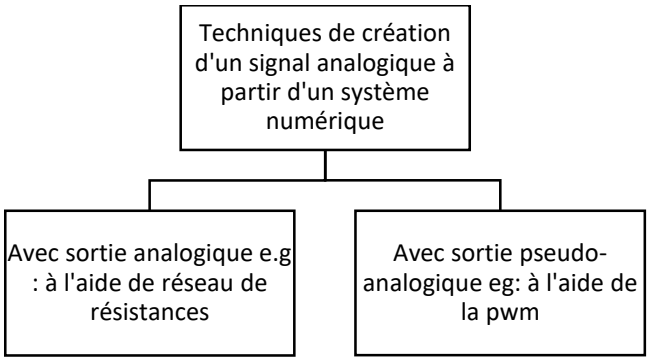


Fig.44 : Techniques de conversion d'un signal numérique en signal analogique

Arduino nano intégrant la pseudo-analogique, c'est celle-ci que nous avons alors utilisée dans notre projet.

1.1.2.3.3. La PWM

Le principe est de créer un signal logique (valant 0 ou 1), à fréquence fixe mais dont le rapport cyclique est contrôlé numériquement, la valeur moyenne de ce signal étant une grandeur analogique égale au produit du rapport cyclique par l'amplitude maximale du signal (Fig.45).

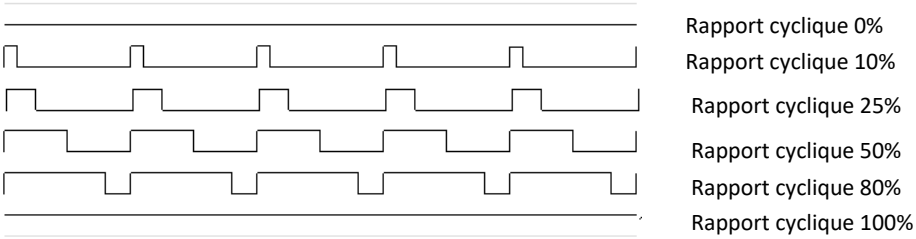


Fig.45 : Signal PWM pour différents rapports cycliques

En extrayant la moyenne de ce signal (au moyen d'un filtre passe-bas), on obtient une valeur analogique V_{moy}, proportionnelle à ce rapport cyclique :

$$V_{moy} = \frac{1}{T}\left(\int_{0}^{T_{ON}} E.dt + \int_{T_{ON}}^{T} 0\,.dt\right) = E.\frac{T_{ON}}{T} = E.\frac{N}{N_{MAX}}$$

Avec :

E : amplitude maximale du signal logique

T : période du signal logique

T_{ON} : durée de l'impulsion du signal logique

N : la valeur numérique correspondant à T_{ON}

N_{MAX} : la valeur numérique correspondant à T.

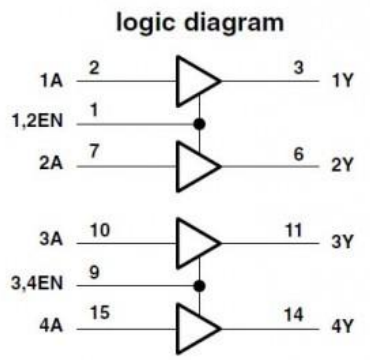
1.1.2.3.4. Limites d'utilisation de la PWM

Pour qu'un signal de PWM puisse être utilisé, il faut que la charge se comporte comme un filtre passe-bas conservant la composante variable du signal. La période doit donc être petite (ou la fréquence grande) devant la fréquence de coupure du filtre pour que le signal de PWM n'influence pas la valeur moyenne.

Maintenant le moteur se comportant comme un filtre passe bas, alors l'utilisation d'un tel filtre pour le commander n'est pas nécessaire.

1.1.2.4. Le LD293D

Simple d'utilisation et se présentant sous forme d'un circuit intégré, il est constitué d'un double pont en H à transistors bipolaires pour charges inductives, comme les moteurs CC (petits moteurs CC).



FUNCTION TABLE
(each driver)

INPUTS†		OUTPUT
A	EN	Y
H	H	H
L	H	L
X	L	Z

H = high level, L = low level, X = irrelevant,
Z = high impedance (off)
† In the thermal shutdown mode, the output is in the high-impedance state, regardless of the input levels.

Fig. 46 : logigramme et table de fonctionnement du LD293D

On donne à la figure .47 ci-dessous comment nous l'avons mis en œuvre dans notre projet.

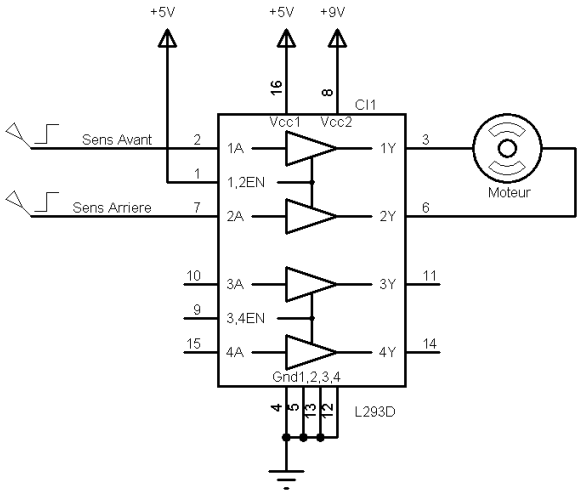


Fig. 47 : Branchement du moteur au LD293D

Nous avons branché le moteur entre deux sorties ayant la même ligne de validation (ligne 1). Cette ligne de validation est mise à un niveau logique haut. Pour faire tourner le moteur dans un sens, on appliquera un niveau logique haut sur une des deux entrées et un niveau logique bas sur l'autre. Pour faire tourner le moteur dans le sens inverse, on inversera les niveaux logiques sur les entrées. On a donc besoin de deux lignes seulement pour piloter un moteur. Maintenant lorsque les signaux d'entrées sont au même niveau, d'après la table de fonctionnement du LD293, les deux bornes du moteur seront au même potentiel ; situation qui correspond au court-circuit du moteur et donc à un freinage dynamique de ce dernier.

2. Réalisation logicielle

2.1. Le PID analogique sélectionné pour l'implémentation

Équation (1) : $u(t) = K(e(t) + \frac{1}{T_i}\int_0^t e(\tau)d\tau + T_d\dot{e(t)})$

2.2 La synthèse du contrôleur PID

Pour les raisons évoquées ci-dessus, nous avons opté de faire la synthèse de la version digitale de type incrémental en utilisant la méthode pragmatique.

Partant de l'équation différentielle (1) décrivant le fonctionnement du PID dans le domaine continu, nous avons abouti à la version numérique en procédant comme suit :

Etape 1 : Différentiation de u

Equation (2) : $\dot{u}$ = k($\dot{e} + \frac{1}{\tau_i} e + \tau_d \ddot{e}$)

Etape 2 : Approximation de $\dot{u}$

Equation (3) : $\dot{u} = \frac{u_k - u_{k-1}}{T_e}$ en utilisant Euler

Où « T_e » est égale à la période d'échantillonnage dite aussi constante de discrétisation ;

$u_k = u(kh)$ et k = 0, 1, 2, 3, …….., représentant les instants de discrétisation.

Etape 2 : Approximation de $\dot{e}$

Equation (4) : $\dot{e} = \frac{e_k - e_{k-1}}{T_e}$

Etape (3) : Approximation de $\ddot{e}$

Equation (5) : $\ddot{e} = \frac{\dot{e}_{k-1} - \dot{e}_{k-1}}{T_e}$

Etape 4 : Remplacement de (4) dans (5)

Equation (6) : $\ddot{e} = \frac{e_k - 2e_{k-1} + e_{k-2}}{{T_e}^2}$

Etape 5 : Remplacement de (3), (4) et (6) dans (2), on obtient l'expression finale du signal de contrôle à l'instant discret k :

Equation (7) : $u_{k=} u_{k-1} + K_0 e_k + K_1 e_{k-1} + K_2 e_{k-2}$

Avec : $K_0 = K(1 + \frac{T_e}{\tau_i} + \frac{\tau_d}{T_e})$, $K_1 = -K(1 + \frac{2\tau_d}{T_e})$, $K_2 = K\frac{\tau_d}{T_e}$

2.3. Choix de la période d'échantillonnage du PID

Le problème du choix de la période d'échantillonnage n'est pas facile à traiter. Il n'existe pas de règle stricte permettant de la fixer.

On peut légitimement lier T_e au temps de montée du système asservi par la relation :

$$N = \frac{T_m}{T_e}$$

Avec T_m égale à la durée nécessaire pour que la sortie réglée passe de 10% à 90% de sa valeur finale (Fig.48), et N le nombre de périodes d'échantillonnage, utilisé pendant T_m.

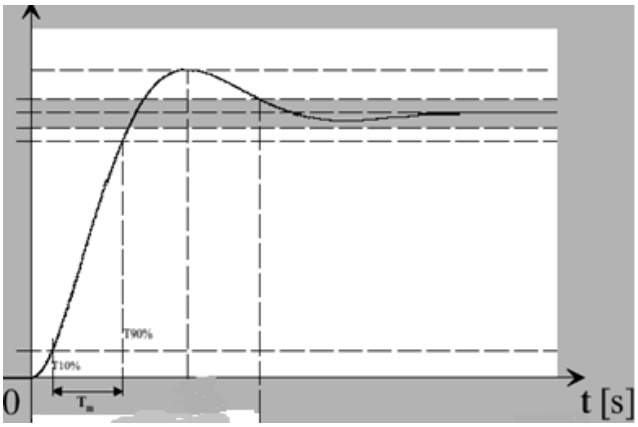


Fig.48: Temps de montée d'une réponse indicielle d'un système du 1er ou 2ème ordre

D'où, pour fixer Te, il suffit de tracer la réponse indicielle et de mesurer Tm et de prendre Te égale par exemple à 10 Tm.

Dans notre projet, n'ayant pas pu se procurer un moteur à courant continu vu les circonstances dans lesquelles ce dernier a été réalisé mais un modèle simulable de celui-ci, nous avons alors procéder tout autrement pour fixer un ordre de grandeur de Te.

Partant de l'idée que les processeurs actuellement sont très rapide (hypothèse vraie), et par conséquent permettent un échantillonnage rapide devant le temps de montée de n'importe quel système du 1er ou 2ème ordre (hypothèse vraie aussi), nous nous sommes basés pour fixer un ordre de grandeur de Te, sur les différents retards accusés par les différentes opérations dans l'asservissement que nous avons réalisé.

Dans notre projet, un seul retard existe et qui est celui produit par la boucle principale du logiciel que nous avons développé. On donne à la figure 49 ci-dessous, les détails relatifs aux différentes fonctions exécutées par cette boucle.

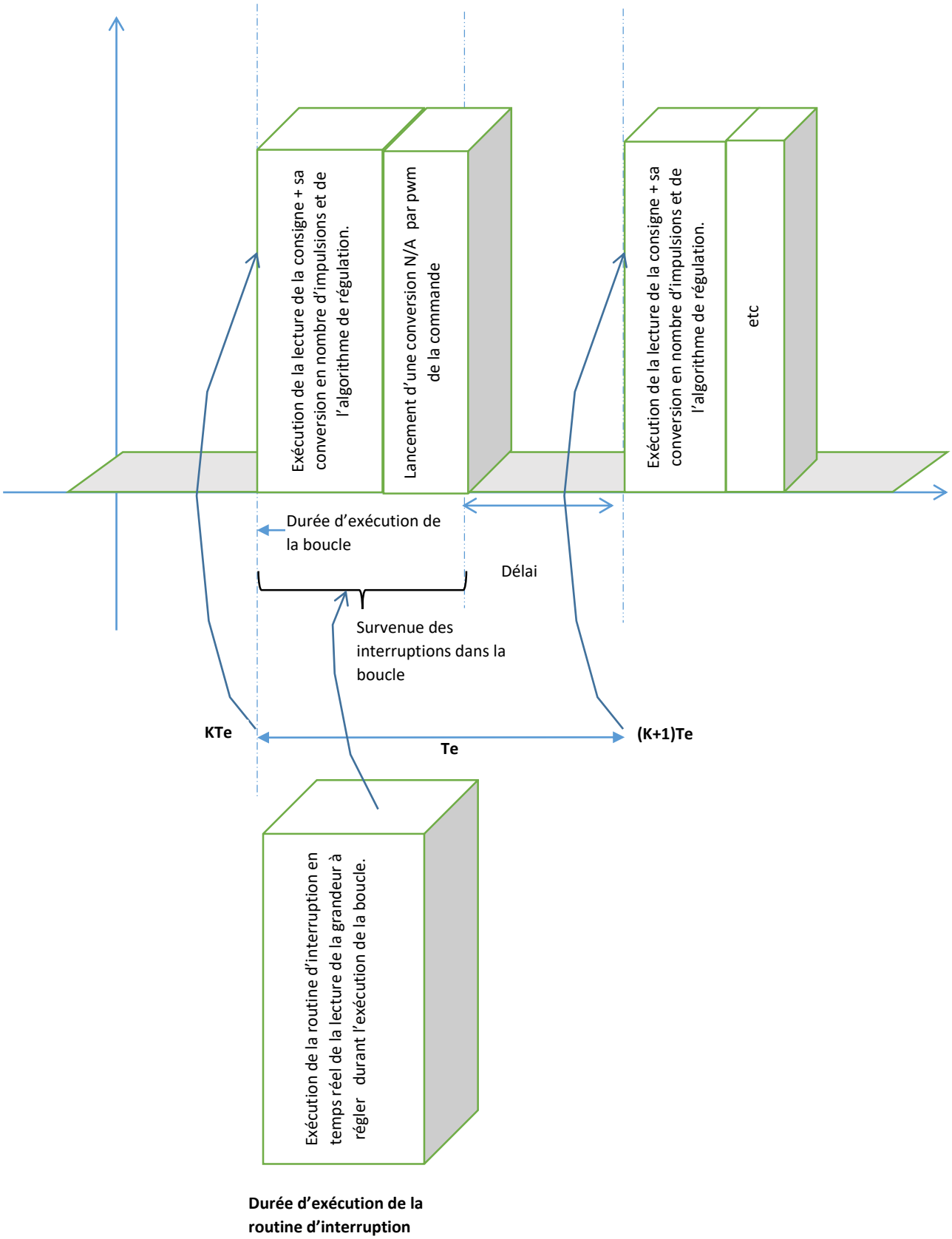


Fig.49: Séquence des opérations effectuées dans le logiciel développé

Donc pour fixer un ordre de grandeur de Te, il suffit de déterminer la durée d'exécution de la boucle.

Cette durée est égale à la somme des durées d'exécution des fonctions de:

- Lecture de la consigne.
- Conversion de la consigne
- Mesure de la grandeur à régler
- Exécution de l'algorithme de régulation
- Conversion NA du résultat d'exécution de l'algorithme de régulation
- Exécution de la routine d'interruption
- Affichage de la valeur mesurée de la grandeur à régler.

Pour déterminer ces différentes durées ainsi que celle de la boucle, nous l'avons fait par programmation.

Pour les fonctions nous avons procédé comme suit :

```
Int Duree  = micros() ;
Fonction() ;
Duree = micros() – Duree ;
Serial.print(Duree) ;
```

Et pour la boucle comme suit :

```
Loop(){
Int Duree  = micros() ;
//le corps de la boucle
Duree= micros() – duree ;
Serial.print(Duree) ;
}
```

Remarque :

En se référant au schéma bloc général d'un asservissement numérique (Fig.29), la conversion A/N est absente dans notre projet car on n'a pas utilisé à proprement parler, un composant matériel pour la réaliser mais un capteur de la grandeur à régler de type encodeur optique incrémental, (Fig. 42) qui fournit intrinsèquement en temps réel, une mesure de la grandeur sous forme discrète lue et stockée en mémoire par une routine d'interruption. Donc en termes de temps cela correspond à la durée d'exécution de cette dernière.

2.4 Le logiciel de régulation réalisé

On en donne à la figure 50 ci-dessous le schéma conceptuel

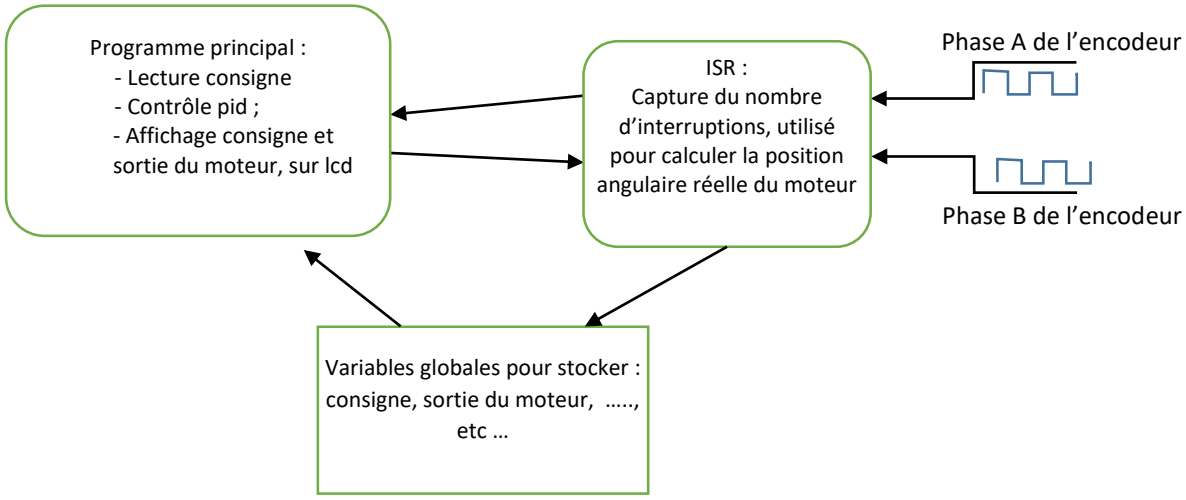


Fig. 50 : architecture du logiciel développé

2.5. L'implémentation du contrôleur PID

Partant de la version digitale (équation (7)) du contrôleur PID, établie dans ce qui précède, nous avons élaboré l'algorithme PID suivant :

Pour k=0,1,2,3,…
- lire y ; y est le signal à la sortie du capteur
- calculer e_k ;
- calculer le signal de contrôle u_k ;
- mettre à jour u_{k-1}, e_{k-1} et e_{k-2} pour la prochaine itération
$u_{k-1} = u_k$;
$e_{k-1} = e_k$;
$e_{k-2} = e_{k-1}$;
- appliquer le signal de contrôle à l'actionneur ;
- attendre delai secondes ;

D'où **Te = (Delai + dureeDexecutionDeLaBoucle)** avec dans notre cas, « **Delai** » un retard que nous avons introduit volontairement et qui est calculé sur la base de **Te** et qui correspondant au temps qu'il faut attendre avant d'entreprendre une nouvelle régulation étant donné que **Te** doit être toujours choisie plus grand ou à la limite égale à la durée d'exécution de la boucle : **dureeDexecutionDeLaBoucle <= Te**.

2.6. Lecture de la position angulaire

Les signaux en quadrature de l'encodeur, nous informent sur le sens de rotation du moteur (SAM ou SCAM). Donc en capturant les fronts montants et descendants de ces deux signaux et en utilisant une variable qui va être incrémentée ou décrémentée selon le sens de rotation du moteur, l'on peut savoir à tout moment quelle est la position de ce dernier.

Maintenant pour incrémenter ou décrémenter, ou ne rien faire, c'est là où réside le problème !

L'habituel consiste à vérifier à l'aide d'instructions de contrôle un tas de conditions et d'ajouter 1 ou de soustraire 1 à une variable définie pour contenir la position en nombre de fronts capturés, du moteur en mouvement. Or qui dit instructions de contrôle dit programmation plus compliquée et temps d'exécution plus élevé et par conséquent non vraiment adaptée pour ce genre d'applications.

Dans notre projet, nous avons développé une nouvelle méthode que nous ne l'avons nullement vue utilisée dans tous les projets similaires que nous avons rencontrés !

Elle consiste pour simplifier la programmation, à utiliser un tableau de consultation, prédéfini de 16 entrées (Tableau 2), contenant chacune soit : 0, +1, ou -1 et à y accéder à chaque fois qu'il y un changement sur les entrées d'interruption pour lire les états des deux phases A et B

de l'encodeur, leur effectuer un certain traitement ensuite extraire l'information quand au type d'opération à effectuer (incrémenter, décrémenter ou ne rien faire)

0	-1	1	0	1	0	0	-1	-1	0	0	1	0	1	-1	0

Tableau 2 : table de consultation de la méthode proposée

3 Tests et discussions des résultats obtenus

Voir vidéo sur le lien suivant :

https://drive.google.com/file/d/1qS-Bi8S8J7ATk4XDsSky6z_4fgdPFSEp/view?usp=sharing

Nous avons pour vérifier le bon fonctionnement du régulateur, utilisé l'oscilloscope et le fréquencemètre pour visualiser l'évolution des phases A et B de l'encodeur optique, et l'afficheur LCD pour afficher la consigne ainsi que la position angulaire réelle et ce, depuis le départ du moteur jusqu'à son arrêt quand la consigne est atteinte.

Sur l'oscilloscope on devrait voir les périodes des deux phases A et B varier depuis une valeur maximale jusqu'à s'annuler quand le moteur s'arrête.

Idem pour le fréquencemètre, on devrait voir la fréquence de l'une des deux phases A et B varier depuis une valeur maximale jusqu'à s'annuler quand le moteur s'arrête.

Aucune anomalie n'a été constatée. Cependant et étant donné que ce test est fait par simulation sur Proteus et que nos machines ne sont pas très rapides, cette dernière n'est pas faite en temps réelle et on a l'impression que le moteur met du temps pour se positionner à la valeur demandée.

On fait remarquer que les valeurs de test utilisées, nous les avons introduites directement dans le programme que nous avons développé. L'utilisation d'un périphérique pour ce faire, ne constituait aucun problème pour nous si nous avions voulu l'ajouter.

En perspective, des améliorations pourraient être apportées à notre projet comme par exemple :

- La maitrise de la vitesse du moteur durant son déplacement vers la consigne car une régulation PID de position avec boucle unique ne le permet pas.
- La réalisation d'une IHM du type GUI.

Conclusion générale

Nous avons réalisé dans ce projet de fin d'études un prototype de régulateur PID numérique de régulation de position angulaire de petits moteurs à courant continu avec un peu de nouveau en programmation.

Le nouveau réside dans la méthode que nous utilisons pour lire la position angulaire du moteur durant sa rotation.

En effet, ce qui est fait dans l'habituel, consiste à vérifier à l'aide d'instructions de contrôle un tas de conditions et d'ajouter 1 ou de soustraire 1 à une variable définie pour contenir la position en nombre de fronts capturés des deux phases A et B en quadrature de l'encodeur alors que le moteur est en mouvement. Or qui dit instructions de contrôle dit programmation plus compliquée et temps d'exécution élevé et par conséquent non vraiment adaptée pour ce genre d'applications. Et pourtant, c'est ce qui est fait dans tous les projets similaires que nous avons rencontrés jusqu'à maintenant.

A la place et pour simplifier, la méthode que nous proposons dans notre projet, consiste tout simplement à utiliser une table de consultation qui en y accédant à chaque fois que l'on détecte un front parmi les fronts des phases A et B de l'encodeur, elle nous renseigne sur le fait de ne rien faire, d'incrémenter ou de décrémenter la variable contenant la position actuelle du moteur. Donc en matière de programmation, elle est beaucoup plus élégante et pour ce qui est la rapidité elle est plus adaptée.

Ceci dit, ce projet nous a fait beaucoup apprendre et il nous a permis de consolider nos connaissances en régulation et asservissement aussi bien en théorie qu'en pratique et de se familiariser davantage avec des méthodes de développement qui nous ont permis d'améliorer nos compétences et nos acquis dans la programmation en langage arduino.

Enfin ce projet peut constituer un très bon départ et un très bon prélude pour des projets futurs se faisant dans ce domaine car nous y avons rassemblés toutes les bases nécessaires et fondamentales.

Bibliographie

[1] http://brettbeauregard.com/blog/2011/04/improving-the-beginners-pid-introduction/

[2] https://zestedesavoir.com/tutoriels/686/arduino-premiers-pas-en-informatique-embarquee/747_le-mouvement-grace-aux-moteurs/3437_le-moteur-a-courant-continu/

[3] http://perso.iut-nimes.fr/fgiamarchi/?p=740

[4] http://www.specialautom.net/synthese-empirique.htm

[5] http://cerig.pagora.grenoble-inp.fr/tutoriel/automatique/page01.htm

[6] http://prive.iutenligne.net/n8sSMGgmRhBh4HOj/automatique-et-automatismes-industriels/konn/asnum/numH/temps_fini.html

[7} https://www.abcelectronique.com/annuaire/cours/automatique.php

[8] http://www.bedwani.ch/regul/discret/homedisc.htm