« Systèmes à événements discrets »
TP : Le traitement numérique des signaux pour le contrôle du tangage du drone didactique D2C

Cette fiche détaille le travail à réaliser par le *groupe 2* dans ce TP :

Préalable :
L'objectif général du travail est l'amélioration progressive des performances du système de contrôle du tangage du drone ;
→ lire la description générale des activités du TP, p6 de la ***fiche « TP-systemes-evenements-discrets-presentation »*** pour comprendre le rôle de chaque groupe dans cette première phase, et pour comprendre l'utilité des différentes phases.

La séance de travail comporte trois phases qui sont à peu près de 30 minutes chacune ; chaque groupe prendra soin de réaliser des copies d'écran, pour présenter ses conclusions.
Des réponses écrites seront données pour les résultats qui ne pourront apparaître sur les copies d'écran.
A la fin de chaque phase, une synthèse de 5 minutes permettra à chaque groupe de prendre connaissance des résultats obtenus par les deux autres groupes.

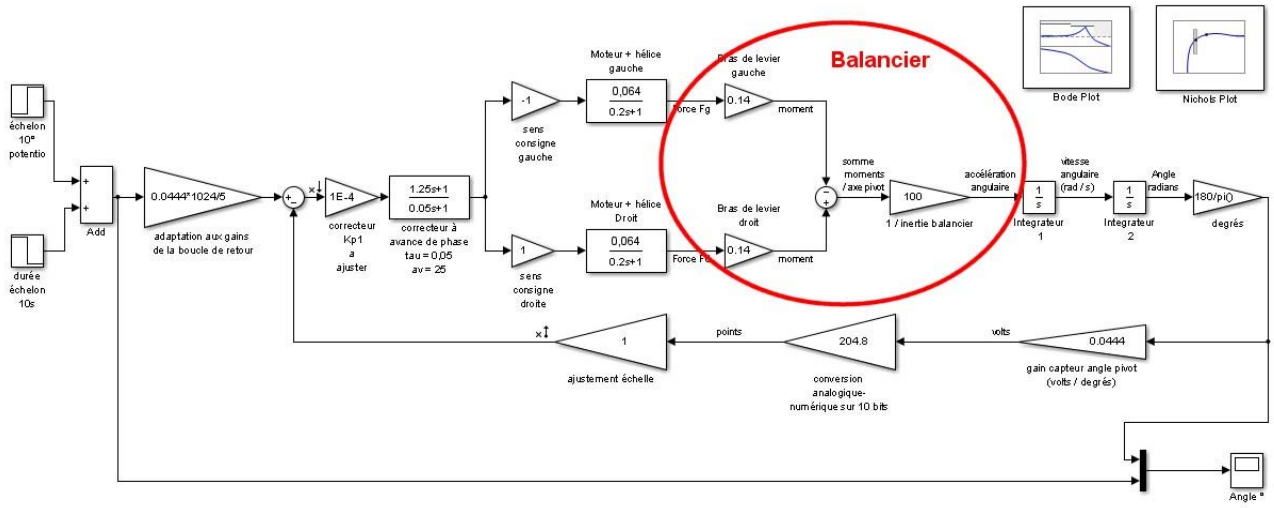
Important : les travaux proposés conduiront à modifier des fichiers de simulation (exploités sous Matlab-Simulink ou Scilab-Xcos) ou des fichier de programmation (exploités dans l'interface de programmation Arduino) ; il faudra dès leur ouverture, enregistrer ces fichiers avec votre nom, avant toute modification.

Phase 1 : « l'étude de l'asservissement à une seule boucle de position »

Dans cette phase, le travail du groupe est centré sur la modélisation de la boucle d'asservissement en position du drone didactique D2C. Une expérimentation permettra de justifier le modèle (fonction de transfert) de la motorisation, puis une simulation logicielle permettra d'ajuster la valeur du coefficient proportionnel Kp1 du correcteur à avance de phase.

Travail 2-1-1 : analyse du schéma-bloc

→Ouvrir le fichier ***« simu_D2C_Arduino_1boucle_position_a_ajuster »*** avec le logiciel de simulation du laboratoire (Matlab-Simulink ou Scilab-Xcos) ; l'enregistrer avec votre nom.



Aéroparc Saint Martin – 12 rue de Caulet – 31300 TOULOUSE – ☎ : + 33 (0)5 62 88 72 72 🕮 : + 33 (0)5 62 88 72 79
Site internet : www.dmseducation.com Email : info@dmseducation.com

a) Analyse de la boucle de retour

→ ***Vérifier auprès du groupe 1***, la valeur 0,0444 indiquée pour le gain du capteur « angle pivot » ;
→ Justifier à l'aide de la ***« fiche conversion-signaux-analogiques »***, le coefficient 204,8 associé au convertisseur analogique numérique.
→ Justifier le coefficient de la chaîne directe nommé « adaptation aux gains de la boucle de retour »

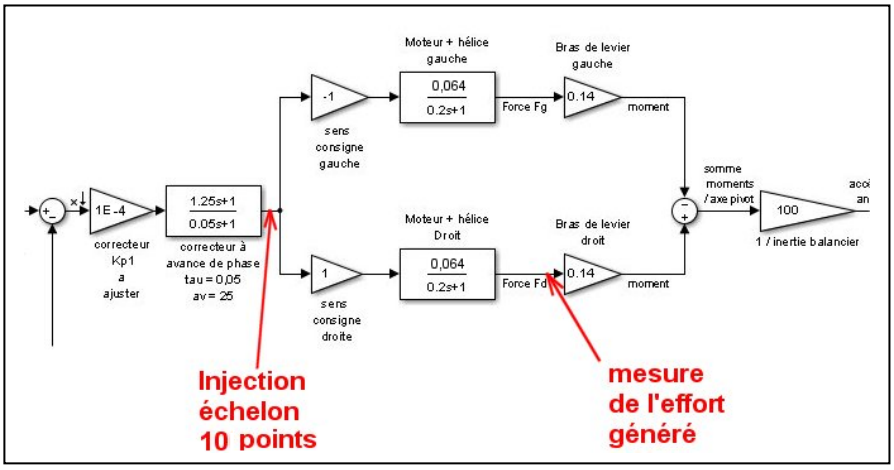
b) Analyse de la chaîne directe :

→ Indiquer quelle est l'équation qui a été utilisée pour modéliser la zone désignée par « Balancier » sur le schéma-bloc ci-dessus, et comment elle a été obtenue.
→ déterminer la « classe » de la fonction de transfert en boucle ouverte et justifier la nécessité d'utiliser un correcteur de type dérivé (on utilisera le correcteur à avance de phase), vis à vis du critère de stabilité du cahier des charges.

Travail 2-1-2 : validation du modèle de la motorisation

L'objectif est de valider la fonction de transfert de la motorisation, mise en place dans la modélisation ;
Pour cela on étudie les signaux de part et d'autre du bloc motorisation droit :

On se propose d'analyser (mesurer) l'effort généré par la motorisation et l'hélice, lors de l'injection d'une consigne en échelon de 10 points (voir sur la figure ci-contre).



→ Ouvrir le programme ***« _0_generation_echelons_moteurs.ino »*** dans l'interface de programmation Arduino ; l'enregistrer avec votre nom.
Les premières lignes présentent les variables qui vont permettre la programmation d'échelons de vitesse des moteurs :

```
// variables pour la commande programmée d'échelons qui se répètent :
int amplitude_echelon = 10;          // amplitude prévue pour l'échelon de commande moteur (en points de calculateur 10 bits : 0 à 1023)
int point_fonctionnement = 400;      // le point de fonctionnement de vitesse = le point de départ de l'écheleon (en points de calculateur 10 bits : 0 à 1023)
long temps_echelon = 2;              // temps en secondes prévu pour la réalisation de l'échelon
long temps_debut_mesure_echelon = 0;       // variable pour le calcul du temps de l'échelon
long temps_mesure_echelon = 0;       // variable pour le calcul du temps de l'échelon
int echelon = 0;                     // variable pour définir le niveau haut ou bas de la commande
```

→ Utiliser la ***« fiche mesure-caracteristiques-motorisation »*** et réaliser l'expérimentation pour valider les coefficients de la fonction de transfert de la motorisation $\frac{0,064}{1+0,2.p}$, proposés dans la simulation.

→ Des petits écarts peuvent exister entre les résultats de cette expérimentation et le modèle proposé ; présenter des explications.

Société DMS
Aéroparc Saint Martin – 12 rue de Caulet – 31300 TOULOUSE – ☎ : + 33 (0)5 62 88 72 72 📖 : + 33 (0)5 62 88 72 79
Site internet : www.dmseducation.com Email : info@dmseducation.com

Travail 2-1-3 : justification des coefficients du correcteur

Le correcteur à avance de phase a pour fonction de transfert : $Kp1.\frac{1+av.tau.p}{1+tau.p}$

Le schéma-bloc fourni propose : Kp1 = 0,0001 ; tau = 0,05 ; av = 25

Problématique :

Pour obtenir un système rapide, il faut pouvoir mettre un gain proportionnel important dans la boucle. En testant quelques valeurs pour le coefficient proportionnel Kp1, on montre (par l'analyse des diagrammes de Bode ou de Black) qu'il faut que l'avance de phase se produise aux plus hautes fréquences possibles ;

Il faut donc une constante de temps « tau » la plus faible possible.

Mais la constante de temps « tau » du correcteur ne peut pas être choisie trop faible, au risque que le traitement numérique de l'équation de récurrence du correcteur ne puisse pas être effectué ; pour cette raison, on choisit pour la suite, la valeur de tau à 2,5 fois la période d'échantillonnage.

→ Obtenir auprès du groupe 3, la valeur de la période d'échantillonnage du programme Arduino et justifier le choix du dénominateur de la fonction de transfert du correcteur.

→ Comparer les possibilités de la solution Arduino avec celles la solution DsPic de période d'échantillonnage 0,0032 s.

Le choix de la constante « av » du correcteur est fixé par la marge de phase souhaitée ; Lancer la simulation et montrer que la valeur proposée dans le schéma-bloc permettra de respecter la condition de stabilité (marge de phase = 45°) du cahier des charges, si le coefficient Kp1 est correctement choisi.

→ Effectuer une ou plusieurs simulations logicielles pour déterminer la valeur optimale du coefficient proportionnel Kp1. Multiplier ce coefficient par 2 pour tenir compte des écarts entre la modélisation et le système réel et ***Indiquer la valeur de ce nouveau coefficient Kp1 au groupe 3 pour qu'il puisse effectuer la programmation dans l'interface Arduino.***

→ Déterminer le temps de réponse à 5% et conclure sur le choix de l'asservissement à une seule boucle vis-à-vis du respect de tous les critères du cahier des charges.

Conclusion de la phase 1 :

→ Rédiger, dans le compte-rendu commun aux trois groupes, une courte synthèse des démarches réalisées et des résultats obtenus, en respectant (ou en créant) l'organisation logique de ce compte-rendu de façon à présenter les détails la fonction "traiter" réalisée par le microcontrôleur.

Une synthèse présentée sous forme de synoptique sera bien appréciée.

Aéroparc Saint Martin – 12 rue de Caulet – 31300 TOULOUSE – ☎ : + 33 (0)5 62 88 72 72 🕮 : + 33 (0)5 62 88 72 79
Site internet : www.dmseducation.com Email : info@dmseducation.com

Phase 2 : Etude de l'asservissement de la vitesse de tangage ;

Dans cette phase, le travail du groupe essentiellement réalisé autour de l'acquisition des informations de vitesse angulaire fournies par le capteur « gyromètre ».
Ce capteur présente l'intérêt de fournir ses informations sous deux formes différentes :

- **première forme : signal analogique** ; l'analyse portera sur l'échantillonnage et la quantification réalisés par le convertisseur analogique-numérique du microcontrôleur de l'Arduino sur ce signal ;
- **deuxième forme : information numérique** transmise par un bus « SPI » ; l'analyse portera sur le mode de transmission du signal ainsi que sur sa quantification.

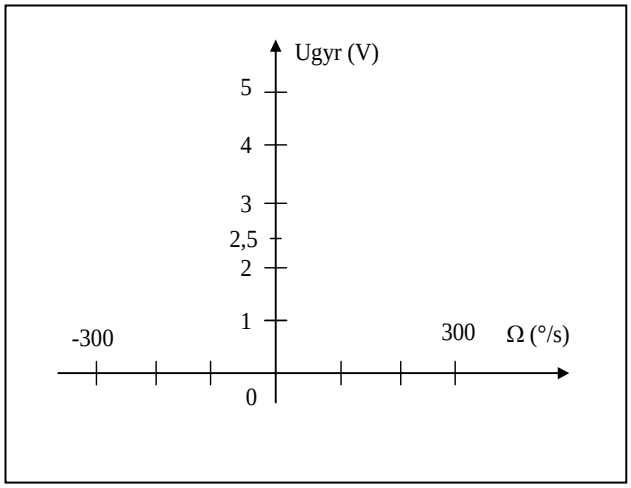
Pour les deux formes, il s'agira de compléter la programmation réalisée pour la lecture du signal dans le microcontrôleur de l'Arduino.

Travail 2-2-1 : analyse du signal analogique fourni par le capteur Gyromètre
→ analyser les spécifications électriques du « gyromètre », fournies dans la **fiche technique « gyro-MLX90609_datasheet.pdf »** par le constructeur :
- à la page 5; relever les deux valeurs extrêmes du signal de sortie « Output, max » et « Output, min » ;
- à la page 6, relever l'amplitude du signal de sortie « Output Full Scale » en Volts,
l'étendue de mesure « Full Scale Range » en degrés par seconde, de la version R2 utilisée sur le drone didactique D2C, ainsi que la valeur à vitesse nulle « « Zero Rate Output ».

→ en déduire le tracé du graphe de la réponse du gyro (figure ci-contre à reproduire et compléter) ;

→ en déduire la validation de la valeur de la « sensibilité » (ou gain) du gyromètre de $6{,}67.10^{-3}$ V/(°/s)

→ Transmettre cette valeur de gain Kgyro au groupe 2 pour qu'il puisse valider le schéma-bloc avant d'effectuer les simulations.



→ Ouvrir la porte du côté droit du système D2C, positionner le bloqueur sur « TANGAGE LIBRE » et manipuler le balancier pour vérifier les réponses précédentes ; on utilisera le voltmètre placé entre la borne « MASSE » et la borne « Ugyr » du pupitre.

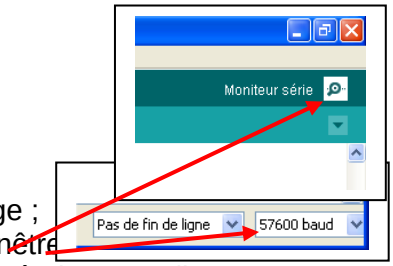
Travail 2-2-2 : analyse de la conversion analogique / numérique
Comme l'indique le document **« Fiche_traitement-numerique-signaux »,** aux paragraphes 1 et 2, le microcontrôleur qui va effectuer la conversion analogique / numérique des signaux possède un convertisseur qui fonctionne sur 10 bits pour une plage de signaux d'entrée de 0 à 5 volts ;
→ en déduire :

- la plage de variation des signaux convertis (en « points » de calcul) ;
- la valeur obtenue à vitesse nulle, lors de la mesure de la vitesse de tangage par le capteur « gyromètre ».

Aéroparc Saint Martin – 12 rue de Caulet – 31300 TOULOUSE – ☎ : + 33 (0)5 62 88 72 72 🕮 : + 33 (0)5 62 88 72 79
Site internet : www.dmseducation.com Email : info@dmseducation.com

→ Calculer la « résolution » de l'acquisition (ou « quantum ») en degrés par seconde, de la mesure du capteur « gyromètre», pour pouvoir la comparer ultérieurement à celle obtenue par l'acquisition avec la forme numérique. (rappel : la résolution est la plus petite variation mesurable).

Travail 2-2-3 : mise en œuvre de la programmation des acquisitions
→ Après avoir identifié le pavé des entrées analogiques sur l'Arduino-box, mettre en place (ou vérifier) le câblage des entrées (Ugyr) entre le pupitre et le microcontrôleur de l'Arduino-box
(voir la **« fiche connexion-pupitre-arduino_box »**) ;
→ Utiliser le programme ***« _0_Test-entrees.ino »*** dans l'interface de programmation Arduino ; l'enregistrer avec votre nom ; supprimer les « // » devant les lignes qui permettront de visualiser les valeurs de l'entrée « gyro ».



→ Téléverser le programme dans la mémoire de l'Arduino ;
→ Cliquer sur l'icône du port série (en haut à droite) pour obtenir l'affichage ;
→ Ajuster la vitesse du port série à 57600 Bauds (en bas à droite de la fenêtre
→ Valider les résultats du travail 2-2-2 sur la plage de variation et la valeur à vitesse nulle.

Travail 2-2-4 : analyse du signal numérique fourni par le capteur Gyromètre
Données : le document ***« Fiche_traitement-numerique-signaux »*** présente au paragraphe 3, les principes d'obtention de la mesure numérique de la vitesse réalisée par le gyromètre ;
→ en déduire :

- la plage de variation des signaux convertis (en « points » de calcul) ;
- la valeur obtenue à vitesse nulle, lors de la mesure de la vitesse de tangage par le capteur « gyromètre ».

→ Après avoir identifié les connexions du bus SPI, d'une part sur le pupitre du drone didactique, et d'autre part sur l'Arduino-box, mettre en place (ou vérifier) le câblage de ce bus SPI entre le pupitre et l'Arduino-box (voir le §3-1 de la ***« Fiche_traitement-numerique-signaux »***)

→ Ouvrir le programme ***« _0_Mesure_Gyro_SPI.ino »*** dans l'interface de programmation Arduino ; vérifier l'enchaînement des étapes permettant la lecture du signal ; indiquer quelle est l'opération logique qui permet d'extraire les 11 bits qui contiennent le signal.

→ Téléverser le programme dans la mémoire du microconrôleur de l'arduino, et valider (ou non !) les deux réponses précédentes (plage et valeur à l'arrêt). (on règlera la vitesse de transmission du port série à 9600 bauds).

→ Calculer la « résolution » de l'acquisition (ou « quantum ») en degrés par seconde, de la mesure du capteur « gyromètre», et comparer celle-ci à celle obtenue par l'acquisition avec la forme analogique, et conclure quant-au choix optimal : numérique ou analogique dans le cas où l'on cherche la précision de mesure du signal gyromètre.

Conclusion de la phase 2 :
Rédiger dans le compte-rendu commun aux trois groupes, une courte synthèse des démarches réalisées et des résultats obtenus dans ce mode de pilotage en vitesse (respecter l'organisation logique de ce compte-rendu de façon pour présenter les détails de la fonction "traiter" réalisée par le microcontrôleur dans ce mode vitesse). Une synthèse présentée sous forme de synoptique sera bien appréciée.

Société DMS
Aéroparc Saint Martin – 12 rue de Caulet – 31300 TOULOUSE – ☎ : + 33 (0)5 62 88 72 72 🕮 : + 33 (0)5 62 88 72 79
Site internet : www.dmseducation.com Email : info@dmseducation.com

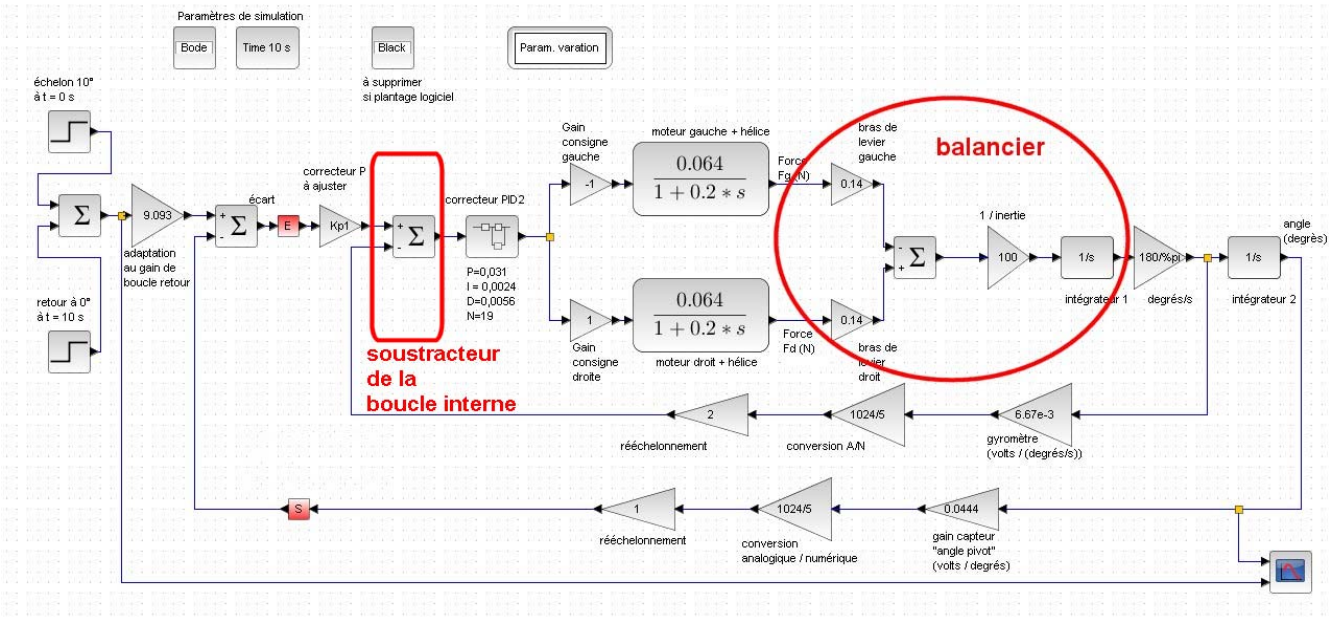
Phase 3 : Etude de l'asservissement à deux boucles imbriquées ;

L'objectif est maintenant de travailler sur le modèle du système à deux boucles, pour :

- d'une part analyser le schéma-boc en vue de comprendre la programmation à effectuer ;
- d'autre part, mettre en place certains éléments de cette programmation et implanter le programme dans la mémoire du microcontrôleur Arduino, puis expérimenter en commande de tangage.

Travail 2-3-1 : programmation du soustracteur de la boucle de vitesse

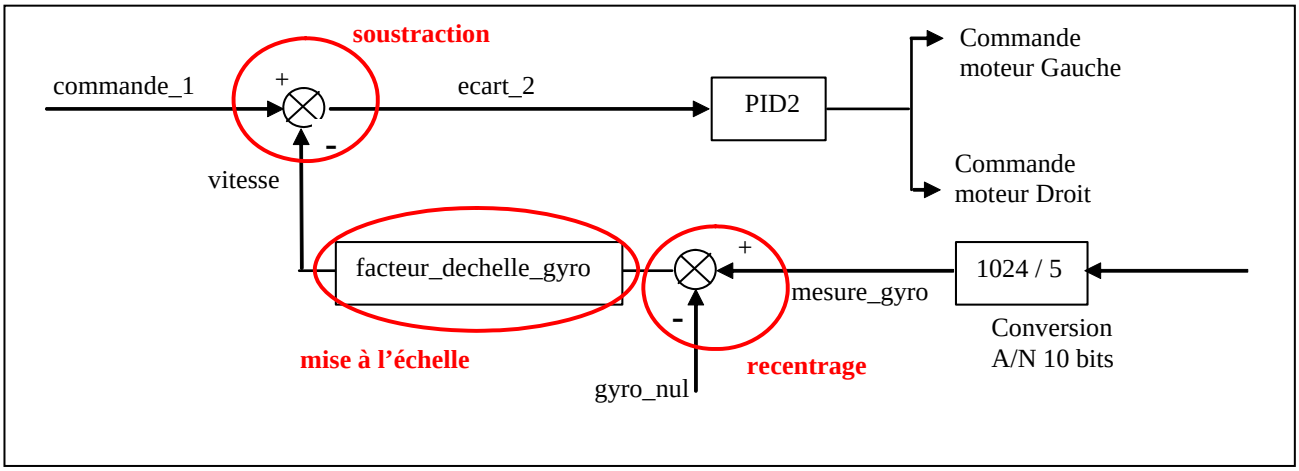
Le fichier de simulation ***« simu_D2C_Arduino_2boucles_a_ajuster »*** ***utilisé par le groupe 1***, avec le logiciel de simulation du laboratoire (Matlab-Simulink ou Scilab-Xcos) propose le schéma-bloc de l'asservissement à deux boucles :



on y retrouve les différents coefficients de la boucle de retour de vitesse étudiée dans la phase précédente.

L'objectif de ce travail est de mettre en place la commande qui réalise le soustracteur de la boucle de vitesse dans le schéma-bloc ci-dessus.

Le détail des opérations est expliqué ci-dessous :



Aéroparc Saint Martin – 12 rue de Caulet – 31300 TOULOUSE – ☎ : + 33 (0)5 62 88 72 72 🕮 : + 33 (0)5 62 88 72 79
Site internet : www.dmseducation.com Email : info@dmseducation.com

Données : les grandeurs qui proviennent de la consigne sont des grandeurs positives et négatives ; or l'étude de la partie 2 a permis de montrer que si l'on utilise la sortie analogique du gyromètre, l'information numérique qu'il délivre est codée sur 10 bits, donc de 0 à 1023.
Il s'agit donc non seulement d'effectuer une soustraction entre la consigne et la mesure, mais préalablement de recentrer cette mesure autour de zéro.

Trois opérations sont donc à effectuer : recentrage, mise à l'échelle et soustraction.

Le programme « ***_3_asservit_2_boucles_P1_PIDF2_a_completer.ino*** » lisible sous l'interface de programmation arduino réalise la commande associée au schéma-bloc ci-dessus.
Les variables utilisées pour réaliser la commande sont :

- sur la chaîne directe : « commande_1 » ; « ecart_2 »
- sur la boucle de retour : « mesure_gyro » ; « gyro_nul » ; « facteur_dechelle_gyro » et « vitesse »

→ compléter les lignes 141 et 142 de ce programme, en exprimant successivement les variables « vitesse » puis « ecart_2 » ;

```
//*************************************************************************************************************************************************************
//boucle principale
void loop(){
  temps_debut_cycle = micros();                  // pour le calcul et la fixation du temps de cycle (nécessaire aux correcteurs)

  /* ************ lecture des entrées capteurs ************************************** */
  mesure_gyro=analogRead(broche_gyro);                      // acquisition de la tension du gyro 0 à 5V  (valeurs de 0 à 1023 sur 10 bits)
  mesure_angle_pivot = analogRead(broche_angle_pivot);           // acquisition de la tension du potentio 0 à 5V sur 10 bits (valeurs de 0 à 1023 )

  // ******************************   asservissement ***********************************
  //******************* traitement de la commande du tangage (boucle de position) **************
  mesure_potentio_droit = analogRead(broche_potentio_droit);           // acquisition de la tension du potentio 0 à 5V sur 10 bits (valeurs de 0 à 1023 )
  ecart_1 = 0.5*(float(mesure_potentio_droit - 512)) - float(mesure_angle_pivot - mesure_angle_pivot_nul);   // calcul de l'écart entre consigne et mesure (0.5 pourt amplifier le m
  commande_1 = PID1_out(ecart_1, periode_echantillonnage);       // application du correcteur PID1

  //******************* traitement de la boucle de vitesse **************
  vitesse = // <---- c'est ici qu'il faut écrire l'équation de variable "vitesse" générée par la boucle de retour
  ecart_2 = // <---- c'est ici qu'il faut écrire l'équation de variable "ecart_2" en fonction de "vitesse et "commande_1"
  commande_2 = PID2_out(ecart_2, periode_echantillonnage);
  commande_unitarisee = commande_2 / 1023.0f;  // rééchelonnement de la commande de tangage : valeur de la commande (-1023 à 1023) ramenée sur une échelle de -1 à 1

  //********************** traitement de la commande des gaz :
  mesure_potentio_gauche = analogRead(broche_potentio_gauche);           // acquisition de la tension du potentio gauche à 5V sur 10 bits (valeurs de 0 à 1023 sur 10 bits)
  commande_gaz_unitarisee = float(mesure_potentio_gauche) / 1023.0f;  // rééchelonnement de la commande de gaz : valeur de la commande (0 à 1023) ramenée sur une échelle de 0 à 1
  valeur_nominale_creneau = creneau_min + (amplitude_max_creneau * commande_gaz_unitarisee) ;  // créneau nominal généré par la commande des gaz à partir de la valeur de potentio

  //******************************** commande des moteurs ***************************************************
  creneau_gauche = valeur_nominale_creneau + (amplitude_max_creneau * commande_unitarisee) ;      // calcul largeur d'impulsion moteur gauche autour de la valeur nominale
  if (creneau_gauche < creneau_min) creneau_gauche = creneau_min ; // on limite à des valeurs autorisées par la carte de puissance
  if (creneau_gauche > creneau_max) creneau_gauche = creneau_max ;
  creneau_droit = valeur_nominale_creneau - (amplitude_max_creneau * commande_unitarisee) ;      // calcul largeur d'impulsion moteur droit autour de la valeur nominale
  if (creneau_droit < creneau_min) creneau_droit = creneau_min ; // on limite à des valeurs autorisées par la carte de puissance
  if (creneau_droit > creneau_max) creneau_droit = creneau_max ;
  moteur_brushless_gauche.writeMicroseconds(int(creneau_gauche));   // utilise la méthode "writeMicroseconds" de la classe Servo pour générer les créneaux en sortie pour vitesse mo
  moteur_brushless_droit.writeMicroseconds(int(creneau_droit));     // utilise la méthode "writeMicroseconds" de la classe Servo pour générer les créneaux en sortie pour vitesse mo
```

lignes 141 et 142

Travail 2-3-2 : Expérimentation

→ ***Récupérer auprès du groupe 1 qui effectue la simulation, la valeur du coefficient Kp1*** du correcteur et fournir sa valeur à la ligne 29 du programme « ***_3_asservit_2_boucles_P1_PIDF2_a_completer.ino*** » ; l'enregistrer avec votre nom.

```
//************** variables pour les correcteurs ************************************************
int integrateur_max = 100 ;          // plafond du calcul d'integration
long periode_echantillonnage = 20;   // frequence de mise à jour des PID (en millisecondes) ;
// **** le correcdteur PID1 de la boucle de position :
float integrateur_1 = 0;                // variable globale pour la fonction PID1
float precedente_entree_PID1 = 0;          // variable globale pour la fonction PID1
float precedente_sortie_PID1 = 0;          // variable globale pour la fonction PID1
float kp1 = 0;             // valeur à compléter      ligne 29
float ki1 = 0;
float kd1 = 0;
float taul = 1/50;        // = fin du reglage du PID filtré 1
// **** le correcdteur PID2 de la boucle de vitesse :
float integrateur_2 = 0;                // variable globale pour la fonction PID2
float precedente_entree_PID2 = 0;          // variable globale pour la fonction PID2
float precedente_sortie_PID2 = 0;          // variable globale pour la fonction PID2
float kp2 = 0.031;         // valeurs déduites de la simulation sans retards
float ki2 = 0.0024;
float kd2 = 0.0056;
float tau2 = 1/19;        // = fin du reglage du PID filtré 2
```

→ Téléverser le programme dans la mémoire du micro-contrpoleur de l'Arduino ; mettre en place (ou vérifier) le câblage des différents capteurs ainsi que ceux de la motorisation (voir la **« fiche connexion-pupitre-arduino_box »**) ;
→ Mettre en route la motorisation à l'aide du bouton gauche du pupitre ; réaliser un échelon de consigne de 10° .
→ Comparer la réponse du système à celle attendue (***voir la réponse du système simulé fournie par le groupe 1*** et voir la ***« fiche cahier des charges »***).

Travail 2-3-3 : Evolution des correcteurs pour prendre en compte les retards
Au vu des résultats de l'expérimentation précédente, le fonctionnement du système tel qu'il a été réglé est incompatible avec les exigences du cahier des charges ; on se propose de faire évoluer la modélisation et de revoir la détermination des coefficients des correcteurs en conséquence.
Ce sont les différents retards dans les boucles d'asservissement qui vont être ***pris en compte par le groupe 1*** dans une nouvelle modélisation.

→ Prendre en compte les évolutions apportées par le groupe 1 à la modélisation ; pour modifier la valeur du coefficient Kp1 et les valeurs des coefficients du correcteur PID2 (lignes 37 à 39) ;

→ Reproduire l'expérimentation précédente et ***travailler avec le groupe 1 pour proposer des causes aux éventuels écarts constatés*** entre :
- cahier des charges et système réel programmé ;
- système simulé et système réel programmé.

Conclusion de la phase 3 :
→ ***Compléter le compte-rendu commun aux trois groupes***, en ajoutant les résultats associés à cette commande en double boucle.
→ A l'aide de la **« Fiche_ecarts_simu-reel.pdf »** choisir deux ou trois explications pertinentes aux écarts constatés entre les résultats de simulation et les résultats d'expérimentation.

Aéroparc Saint Martin – 12 rue de Caulet – 31300 TOULOUSE – ☎ : + 33 (0)5 62 88 72 72 🕮 : + 33 (0)5 62 88 72 79
Site internet : www.dmseducation.com Email : info@dmseducation.com