

## Travaux Pratiques - AUTOMATIQUE IMPRIMANTE 3D

### RAPPELS :

- **Un compte rendu sera rendu à la fin de chaque séance.**
- Chaque compte rendu donnera lieu à une note ainsi que la présentation.
- La présentation sera faite par **l'ensemble des membres du groupe de TP** **(chacun doit parler !!)**
  
- **A la fin de la séance :**
  - **RANGEZ VOTRE MATERIEL**
  - **ETEIGNEZ LE SYSTÈME**
  - **FERMEZ VOTRE SESSION** (ne pas éteindre l'ordinateur)

Les TPs en ilots (chef de projet, équipe modélisateurs, équipe expérimentateurs) permettent de :

- Développer l'autonomie et la prise d'initiative.
- Initier à l'ingénierie simultanée et au travail collaboratif.



Questions de **REFLEXIONS GLOBALES** pour **TOUTE L'EQUIPE**



Questions pour **l'EQUIPE EXPERIMENTATEURS**



Questions pour **l'EQUIPE MODELISATEURS**

## Travaux Pratiques - AUTOMATIQUE IMPRIMANTE 3D



Nous allons ici étudier l'asservissement en température d'un plateau chauffant d'imprimante 3D.  
*Ce TP est adapté d'un TP réalisé par DémoSciences.*

### Objectifs du TP

- Mesurer le comportement en boucle ouverte
- Modéliser le banc et identifier les paramètres du modèle
- Mettre en œuvre un asservissement numérique sous Python
- Comparer les résultats de mesure et de simulation.

## 1 – Présentation de l'imprimante

L'imprimante 3D comporte un plateau chauffant constitué d'un fil résistif dans sa périphérie. On remarque également une plaque de verre positionnée sur le plateau pour permettre une meilleure adhérence de la matière et une meilleure diffusion de température. Un capteur de température est placé au milieu de la plaque. Les têtes d'impression comportent elles aussi une résistance chauffante positionnée dans un bloc métallique.

Lors d'une phase d'impression, la tête est chauffée pour faire fondre le fil. La température doit être réglée de manière précise pour que l'écoulement de la matière se fasse correctement.

De la même manière, le plateau est chauffé pour permettre une bonne adhérence du fil. Si la température varie trop par rapport à une consigne donnée, la matière n'adhérera pas au plateau ou bien elle fondra trop et la forme ne sera pas correcte.

Ainsi on remarque qu'il est nécessaire de correctement réguler la température non seulement pour la tête mais aussi pour le plateau.

Nous n'étudierons que la régulation pour le plateau dans ce TP.

## 2 – Mesure du comportement en Boucle Ouverte

Le banc est composé :

- D'un plateau en aluminium (carré de côté  $a = 214$  mm et d'épaisseur  $e = 3$  mm) équipé d'une résistance chauffante sur toute la surface supérieure ( $R = 1.1$  Ohm) et d'une thermistance 100 kOhm.
- D'une alimentation stabilisée 12 V.
- D'une carte de commande *Arduino Mega 2560* équipée d'une carte fille de commande d'imprimante 3D dont seuls le hacheur du plateau et le conditionnement de la thermistance seront utilisés.

Le hacheur permet de faire varier la tension d'alimentation de la résistance chauffante entre 0 et 12 V à partir d'une consigne dans le programme sur 8 bits (0-255) envoyée sur la broche 8. La mesure analogique de la tension aux bornes de la thermistance (branchée sur la broche 14 ; CAN sur 10 bits) permet d'obtenir une image  $M$  (en incréments) de la température  $T$  (en degrés) du plateau. La loi liant la  $M$  et  $T$  est non linéaire et une fonction python est fournie pour convertir la mesure  $M$  en température  $T$ .

La carte de commande peut être directement commandée depuis Python. Ouvrir l'interface *Pyzo* et charger le programme *IMPRIMANTE\_3D\_mesure\_BO\_Eleves.py*.

Ce programme permet de réaliser un essai en boucle ouverte, en imposant durant 30 min une consigne de tension de 200 inc pendant 1000 s. La mesure de température est faite toutes les secondes (en réalité, le programme fait la moyenne d'un grand nombre de mesure faites durant le pas de 1s de façon à réduire le bruit de mesure). Le résultat est stocké dans les variables T (mesure de température) et t (temps).



Lancer la mesure pour 200 inc et 1000 temps (lignes 40 et 41 du programme *Python*). Pour cela, lancer le programme sous Python *IMPRIMANTE\_3D\_mesure\_BO\_Eleves.py*



Expliquer les étapes importantes du programme qui s'exécute.

### 3 – Modélisation thermique du plateau chauffant

Le modèle thermique du plateau est directement issu du premier principe de la thermo-dynamique : la variation d'énergie interne ( $C \cdot T$ ) est égale à la puissance reçue de la part de la résistance moins la puissance perdue par échange avec l'air par convection.

$$C \frac{dT}{dt} = P_{elec} - P_{convection}$$

- $C$  : Capacité thermique de la plaque.
- $P_{elec}$  : Puissance reçue par effet Joule (elle sera exprimée en fonction de R et U).
- $P_{convection}$  : Puissance perdue par convection, modélisée par la loi  $P_{convection} = H(T - T_0)$ .
- H : coefficient d'échange thermique de la plaque.

L'aluminium a pour capacité thermique massique  $c = 897 \text{ J.kg}^{-1}.\text{K}^{-1}$  et pour masse volumique  $\rho = 2700 \text{ kg.m}^{-3}$ .

La masse de la plaque est alors de  $m = \rho a^2 e = 0,37 \text{ kg}$  et la capacité thermique de la plaque vaut  $C = c \cdot m = 333 \text{ J.k}^{-1}$ .

Le coefficient  $H$  est moins bien connu. Le coefficient d'échange surfacique  $h$  entre un solide et l'air par convection peut varier entre  $h = 5 \text{ W.m}^{-2}.\text{K}^{-1}$  et  $h = 25 \text{ W.m}^{-2}.\text{K}^{-1}$ . Ces valeurs extrêmes conduisent à un coefficient d'échange pour la plaque (qui dispose de 2 faces) comprises entre  $H = 2ha^2 = 0,46 \text{ W.K}^{-1}$  et  $H = 2,3 \text{ W.K}^{-1}$ . La valeur exacte devra être identifiée sur les mesures.

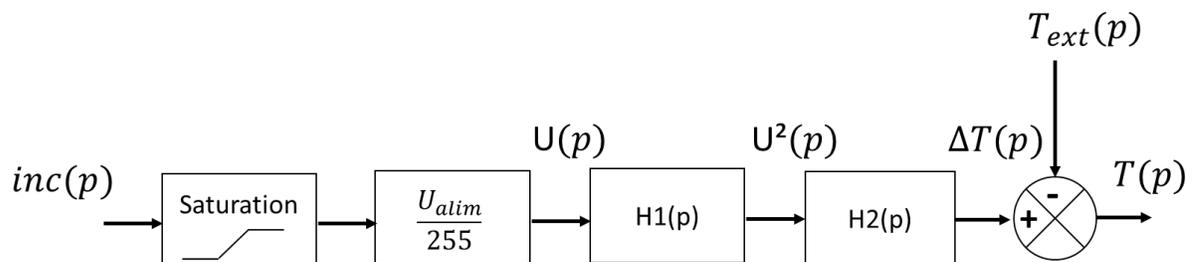
On prendra la valeur moyenne  $h = 10 \text{ W.m}^{-2}.\text{K}^{-1}$  dans un premier temps.



A partir des équations précédentes, déterminer la relation entre  $U$  et  $T$ .



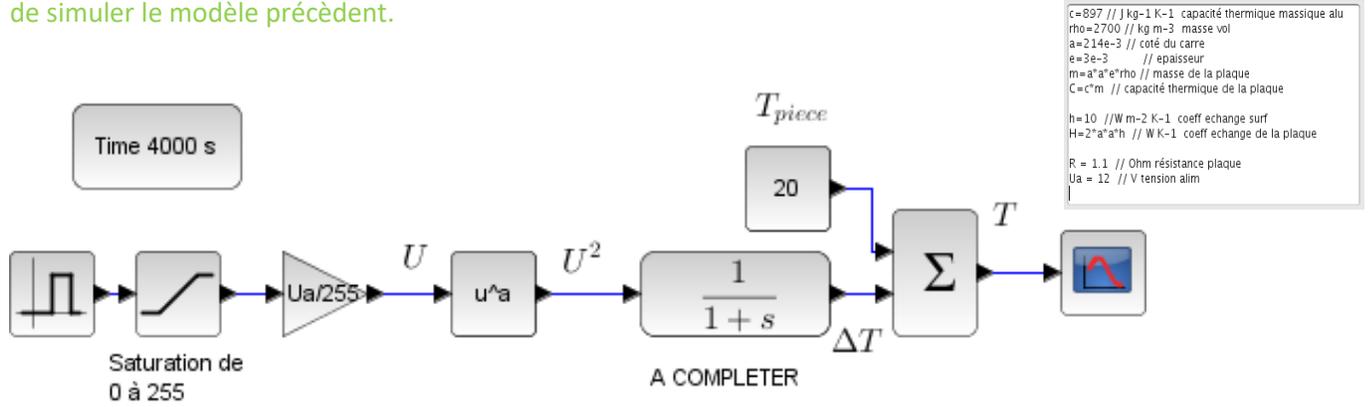
Compléter le schéma bloc ci-dessous (déterminer  $H_1(p)$  et  $H_2(p)$ ), allant de la consigne de tension sur 8 bits du micro-contrôleur à la température de la plaque (incluant les saturations de la commande).



Déterminer la constante de temps du processus en boucle ouverte et la valeur à convergence pour une entrée consigne à 200 incréments.

## 4 – Simulation du comportement en Boucle Ouverte

Compléter le schéma Scilab fourni (*IMPRIMANTE\_3D\_modele\_BO\_Eleves.zcos*) permettant de simuler le modèle précédent.



Tracer le comportement attendu à l'issu de la mesure en boucle ouverte. Commenter le résultat obtenu.

Lorsque la mesure en BO sera terminée, mesurer la constante de temps réelle et la valeur à convergence réelle.

Comparer la constante de temps et la valeur à convergence obtenues par le calcul et par simulation.

Identifier la constante H, de façon à recalibrer le modèle.

## 5 – Influence d'une perturbation

Pour réaliser une perturbation sur le système, lors de la montée (ou descente) en température du système, nous enverrons quelques gouttelettes d'eau. Pour cela, tremper ses bouts de doigts dans l'eau et par pichette, envoyer des gouttes d'eau sur la table chauffante (de préférence au centre, où se situe le capteur). Répéter plusieurs fois l'opération si nécessaire.



Après avoir réfléchi à comment prendre en compte la perturbation, compléter le schéma Scilab et lancer la simulation. Commenter les résultats obtenus.



Réaliser une mesure (sur un temps adapté) pour laquelle des perturbations seront effectuées. Commenter les résultats obtenus.



Expliquer l'influence de la perturbation sur le système étudié. Comment peut-elle être réduite ?

## 6 – Asservissement de température – Correction Proportionnelle

Le correcteur Proportionnel a pour fonction de transfert :  $C(p) = K_p$



Proposer une modification du schéma Scilab pour implanter un asservissement de température avec **correction proportionnelle**.

Remarque : Le comparateur ainsi que le correcteur seront placés entre l'entrée du système () et le bloc saturation ()



Déterminer la constante  $K_p$  du correcteur assurant un fonctionnement correct.



Ecrire un programme Python mettant en œuvre un asservissement de température avec **correction proportionnelle**, en complétant le programme proposé pour la mesure en BO.



La commande `Tension.write()` admet en argument un entier (et pas un flottant). Convertir si nécessaire par la commande `int()`.



A partir de la simulation, déterminer la constante de temps attendue. Commenter les résultats obtenus, notamment pour la phase de descente en température.



Réaliser une mesure (sur un temps adapté) pour vérifier les performances. Commenter les résultats obtenus, notamment pour la phase de descente en température.



Comparer les résultats obtenus expérimentalement à ceux obtenus lors de la simulation numérique. Conclure.

## 7 – Asservissement de température – Correction Proportionnelle-Intégrale

L'asservissement précédent est rapide par rapport à la boucle ouverte, mais il n'est pas précis.

Le correcteur PI a pour fonction de transfert :  $C(p) = K_p + \frac{K_i}{p}$



Proposer une modification du schéma Scilab pour implanter un asservissement de température avec **correction proportionnelle-intégrale**.



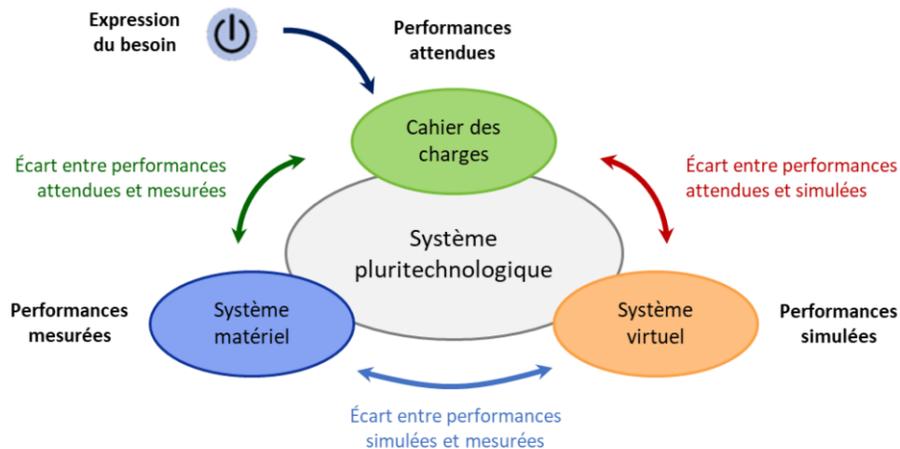
Déterminer la constante  $K_I$  du correcteur assurant un fonctionnement correct. Commenter les résultats obtenus.



Ecrire un programme Python mettant en œuvre un asservissement de température avec **correction proportionnelle-intégrale**, en complétant le programme proposé pour la mesure en BO (voir ANNEXE). Commenter les résultats obtenus.



Comparer les résultats obtenus expérimentalement à ceux obtenus lors de la simulation numérique. Conclure.



## ANNEXE : Réalisation de la partie intégrale du correcteur PI

### Discrétisation de la fonction de transfert du correcteur

On note  $U$  la commande et  $E$  l'écart.

Pour connaître la loi à implanter, il faut écrire la fonction de transfert dans le domaine temporel au moyen d'une équation différentielle :  $\mathbf{u}(t) = \mathbf{K}_p \mathbf{e}(t) + \mathbf{K}_i \int_{T=0}^t \mathbf{e}(T) dT$

On peut discrétiser cette équation à l'instant courant  $t_i$ .

L'intégrale peut s'écrire sous la forme  $\int_{T=0}^t e(T)dT = \int_{T=0}^{t_{i-1}} e(T)dT + \int_{T=t_{i-1}}^{t_i} e(T)dT$

Le calcul numérique de l'intégrale se fait par une des trois méthodes ci-dessous :

- $I_i = I_{i-1} + T_e e(t_i)$  pour une **méthode des rectangles à droite**.
- $I_i = I_{i-1} + T_e e(t_{i-1})$  pour une **méthode des rectangles à gauche**.
- $I_i = I_{i-1} + T_e \frac{e(t_{i-1}) + e(t_i)}{2}$  pour la **méthode des trapèzes**.

On note dans la suite :  $I_i = I_{i-1} + dE$  avec  $dE$  fonction du schéma utilisé.

On implantera par exemple la méthode des rectangles à droite.

Ainsi :  $u(t_i) = K_p e(t_i) + K_i (I_{i-1} + dE)$

## Implantation du correcteur

Le programme sera donc le suivant :

```
while(True) :
    eps=Tc-T
    lterm+= Te*eps
    commande=Kp*eps+Ki*lterm
    preceps=eps
    if commande > 255 :
        commande=255
    elif commande<0 :
        commande=0
    resistance.write(commande)
```

En général pour ne pas que le terme *lterm* devienne trop grand numériquement on limite sa valeur également à  $I_{max}=255$ .

```
if lterm>IMax :
    lTerm = IMax
elif lTerm < 0 :
    lTerm = 0
```

On implante le correcteur en python et on regarde si le réglage du correcteur est satisfaisant (sachant que les perturbations ne sont pas constantes !).

## Autre méthode de discrétisation

On peut aussi écrire la relation à l'instant  $i$  et à l'instant  $i-1$  puis faire la différence :

$$u_i = u_{i-1} + K_p(e_i - e_{i-1}) + K_i \int_{\tau=t_{i-1}}^{t_i} e(\tau) d\tau$$

En prenant la méthode des trapèzes par exemple pour l'intégrale, on obtient :

$$u_i = u_{i-1} + r_0 e_i + r_1 e_{i-1} \quad \text{avec} \quad r_0 = K_p + K_i \frac{T_e}{2} + \frac{K_d}{T_e} \quad \text{et} \quad r_1 = -K_p + K_i \frac{T_e}{2}$$

Une autre méthode consiste à dériver la relation du correcteur  $u(t) = K_p e(t) + K_i \int_{\tau=0}^t e(\tau) d\tau$

$$\frac{du(t)}{dt} = K_p \frac{de(t)}{dt} + K_i e(t)$$

On utilise à nouveau un schéma pour les dérivées :

$$\frac{u_i - u_{i-1}}{T_e} = K_p \frac{e_i - e_{i-1}}{T_e} + K_i e_i$$

Ainsi  $u_i = u_{i-1} + K_p(e_i - e_{i-1}) + K_i e_i T_e$

$$u_i = u_{i-1} + r_0 e_i + r_1 e_{i-1} \quad \text{avec} \quad r_0 = K_p + K_i T_e + \frac{K_d}{T_e} \quad \text{et} \quad r_1 = -K_p - 2 \frac{K_d}{T_e}$$

On obtient donc le même type d'écriture avec des coefficients différents devant  $e_i$  et  $e_{i-1}$ .

Ainsi quel que soit les schémas retenus, on peut toujours écrire la relation de récurrence sous la forme

$$u_i = u_{i-1} + r_0 e_i + r_1 e_{i-1}$$

## Implantation du correcteur

Le programme s'implante de la manière suivante :

```
r0=XXX
r1=XXX
while(True) :
    eps=Tc-T
    commande=commande_prec+r0*eps+r1*eps_prec
    eps_prec_prec=eps_prec
    eps_prec=eps
    commande_prec=commande
    if (commande > 255) :
        commande=255
    elif (commande<0) :
        commande=0
    resistance.write(commande)
```