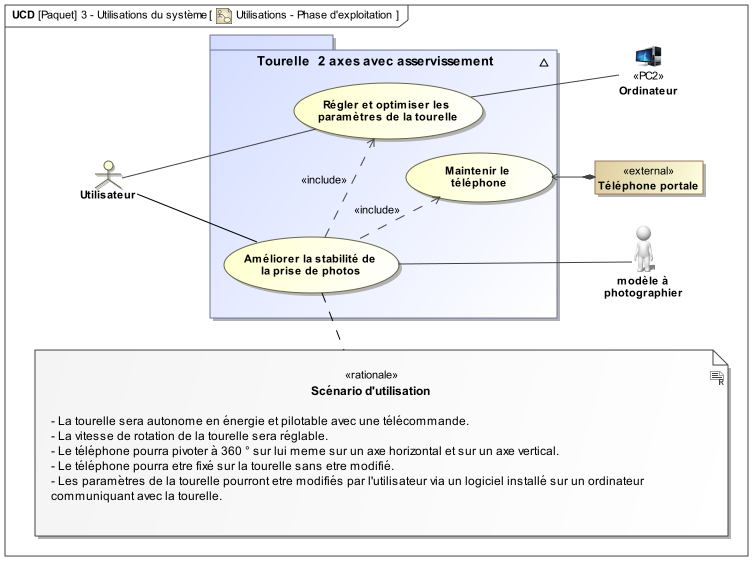
**Tourelle Motorisée**

**2 axes**

|  |  |  |
| --- | --- | --- |
|  |  |  |



L’objectif de la tourelle est de réaliser des photos panoramiques. La qualité de ces photos passe par une maitrise de la vitesse de rotation de l’axe de « pan ».

|  |
| --- |
| **Problématique :**  On s’intéresse à la commande. Le problème est le suivant :  **Comment modéliser et implanter un correcteur numérique permettant de respecter le cahier des charges ?**  Le but de ce TP est d’essayer de répondre à cette question.  **Dans un premier temps, on cherchera à vérifier le comportement de la tourelle. On cherchera ensuite à synthétiser un correcteur et à l’implanter.** |

# Découverte – Manipulation – Observation – Description

|  |
| --- |
| **Objectif : S’approprier le fonctionnement de la tourelle – 25 minutes** |

*Cette première partie nécessite la lecture préalable des fiches 1, 2 et 3.*

|  |
| --- |
| **Activité 1**  En utilisant la fiche 1, mettre en service la tourelle 2 axes à partir du pupitre.  Proposer une description des composants du système. |

|  |
| --- |
| **Activité 2**  Dans cette activité, on pourra utiliser Kp=2, Ki=0 et Kd=0 et en réaliser des échelons de 1 rad.s-1  Proposer un protocole expérimental permettant de vérifier que les exigences 1.4.1 et 1.4.2 sont validées (Fiche 3 « Ingénierie Systèmes »). |

|  |
| --- |
| **Objectif 2: valider les choix technologiques du constructeur – Durée estimée : 5 min** |

|  |
| --- |
| **Activité 3**  Quel capteur permet de mesurer la vitesse de la tourelle ? La résolution de ce capteur est-elle en adéquation avec l’écart statique souhaité ? Par quelle fonction de transfert pourrait-on modéliser le capteur ? |

# Modélisation du système

|  |
| --- |
| **Objectif 3 : Modéliser le comportement du système– Durée : 20 minutes** |

On donne la structure de l’asservissement de la tourelle dans le fichier Tourelle2Axes\_Sujet.slx.

|  |
| --- |
| **Activité 4**  Commenter la structure du modèle au vue de votre connaissance du système (en particulier la structure du correcteur et l’existence du bloc « Transport delay » ou retard.  Réaliser une simulation et confronter les résultats issus de la réponse temporelle à ceux obtenus dans la partie précédente. |

|  |
| --- |
| **Activité 5**  Analyser les résultats issus du diagramme de Bode. Que dire des marges de stabilité ?  Proposer un protocole expérimental permettant d’obtenir un diagramme de Bode sur le système réel. |

|  |
| --- |
| **Activité 6**  Proposer des solutions permettant d’améliorer les performances du système. |

# Conception du correcteur du système

|  |
| --- |
| **Objectif : analyses l’action proportionnelle – Durée : 45 minutes** |

|  |
| --- |
| **Activité 7**   * Quelle est l’influence de la correction proportionnelle sur le temps de réponse du système ? * Quelle est l’influence de la correction proportionnelle sur la précision ? * Un correcteur proportionnel est-il suffisant pour valider les exigences 1.4.1, 1.4.2 et 1.8 du cahier des charges ? * Déterminer le gain permettant d’obtenir le système le plus rapide, avec un dépassement inférieur à 5%. * Confronter les résultats issus de l’expérimentation et ceux issus du modèle. |

|  |
| --- |
| **Activité 8**   * Quelle est l’influence de la correction intégrale sur le comportement du système ? * Déterminer expérimentalement le gain intégral permettant d’avoir un écart statique nul et un dépassement minimal. * Confronter les résultats issus de l’expérimentation et ceux issus du modèle. |

|  |
| --- |
| **Activité 9**  On souhaite implanter un correcteur PI de la forme . Donner l’équation de récurrence de ce correcteur.  En utilisant la fiche 4 implanter ce correcteur dans le logiciel.  Conclure. |

# Synthèse

|  |
| --- |
| **Objectif : *exposer le travail effectué – 10 minutes*** |

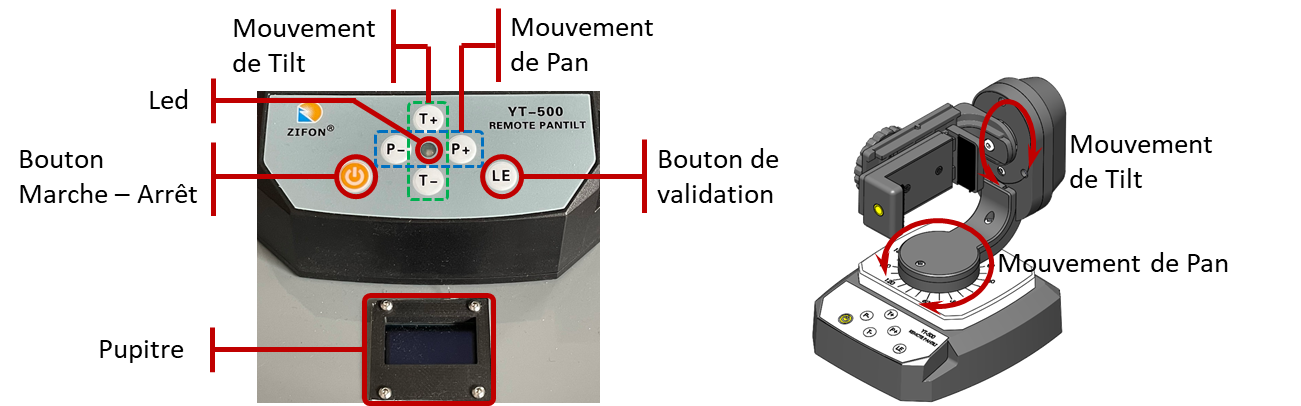
|  |
| --- |
| **Activité 10**  Proposer un poster présentant une synthèse de votre travail. Sur ce poster devront apparaitre les éléments clé des différents temps forts abordés précédemment ainsi que la démarche scientifique mise en œuvre pour répondre à la problématique. Les outils de communication nécessaires à sa rédaction sont laissés à votre initiative. |

# Fiche 1 – Description générale

## Présentation du système

|  |  |
| --- | --- |
| L’utilisation de photographies panoramiques 360° est en pleine croissance pour les photographes professionnels dans divers domaines, tels que l’immobilier, l’industrie, internet, les réseaux sociaux.  Pour répondre au besoin, les constructeurs de smartphones ont développé des fonctionnalités permettant la réalisation de photos panoramiques sur 360°. Lors de la réalisation de celles-ci, le smartphone doit suivre une ligne horizontale avec un déplacement horizontal le plus régulier possible et à une vitesse stable.  La tourelle motorisée 2 axes proposée permet de réaliser cela avec facilité et sans intervention humaine (commande à distance). |  |

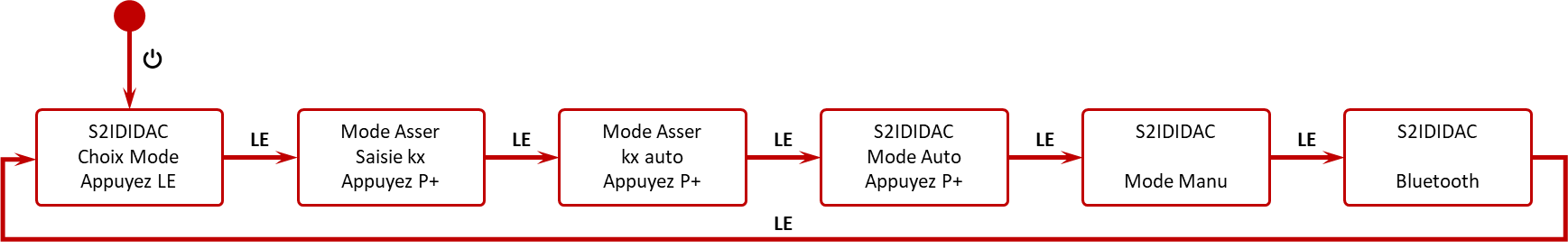
## Mettre en œuvre la tourelle 2 axes à partir du pupitre



## Mise en œuvre rapide

* Brancher le câble d’alimentation uniquement (pas le câble USB).
* Appuyer sur le bouton Marche – Arrêt.
* Appuyer 4 fois sur le bouton **LE** (jusqu’à ce que SD2IDIDAC – Mode Manu soit écrit sur le pupitre et que la led soit éclairée en vert).
* Appuyer sur les boutons de mouvement de Pan ou de Tilt pour déplacer la tourelle autour des deux axes.

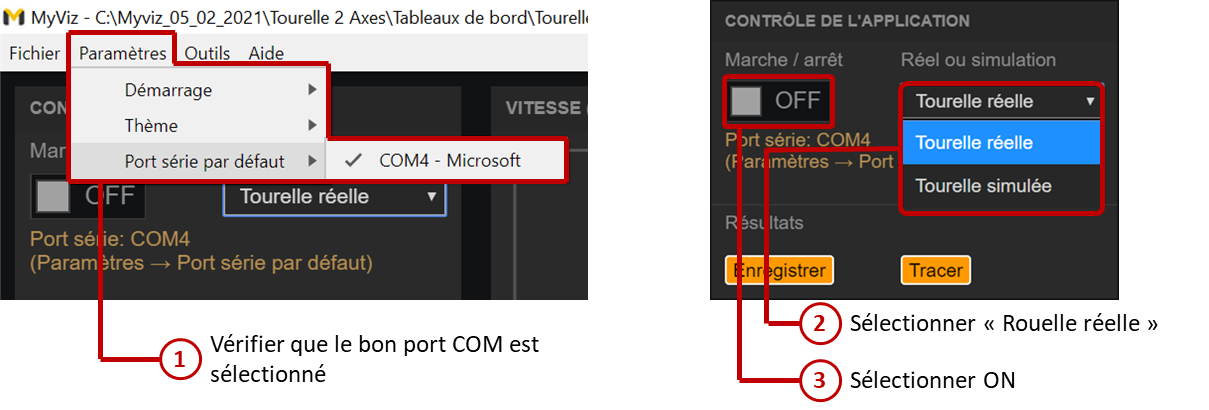
## Menus du pupitre



# Fiche 2 – Mettre en œuvre la tourelle 2 axes à partir de MyViz

## Mise en œuvre

* Vérifier que la tourelle est éteinte.
* Brancher le câble USB.
* Lancer MyViz.
* Ouvrir le Tableau de bord « Tourelle\_AsservissementMoteurEnVitesse ».
  1. Vérifier que le port COM est bien sélectionné.
  2. Sélectionner « Tourelle réelle ».
  3. Sélectionner ON.
  4. La tourelle se met en route seule et s’initialise (rotation dans le sens PAN -) et le pupitre affiche « Mode interface graphique ».



**Remarque : une visualisation 3D de la tourelle se lance également lors de la mise en marche de la tourelle.**

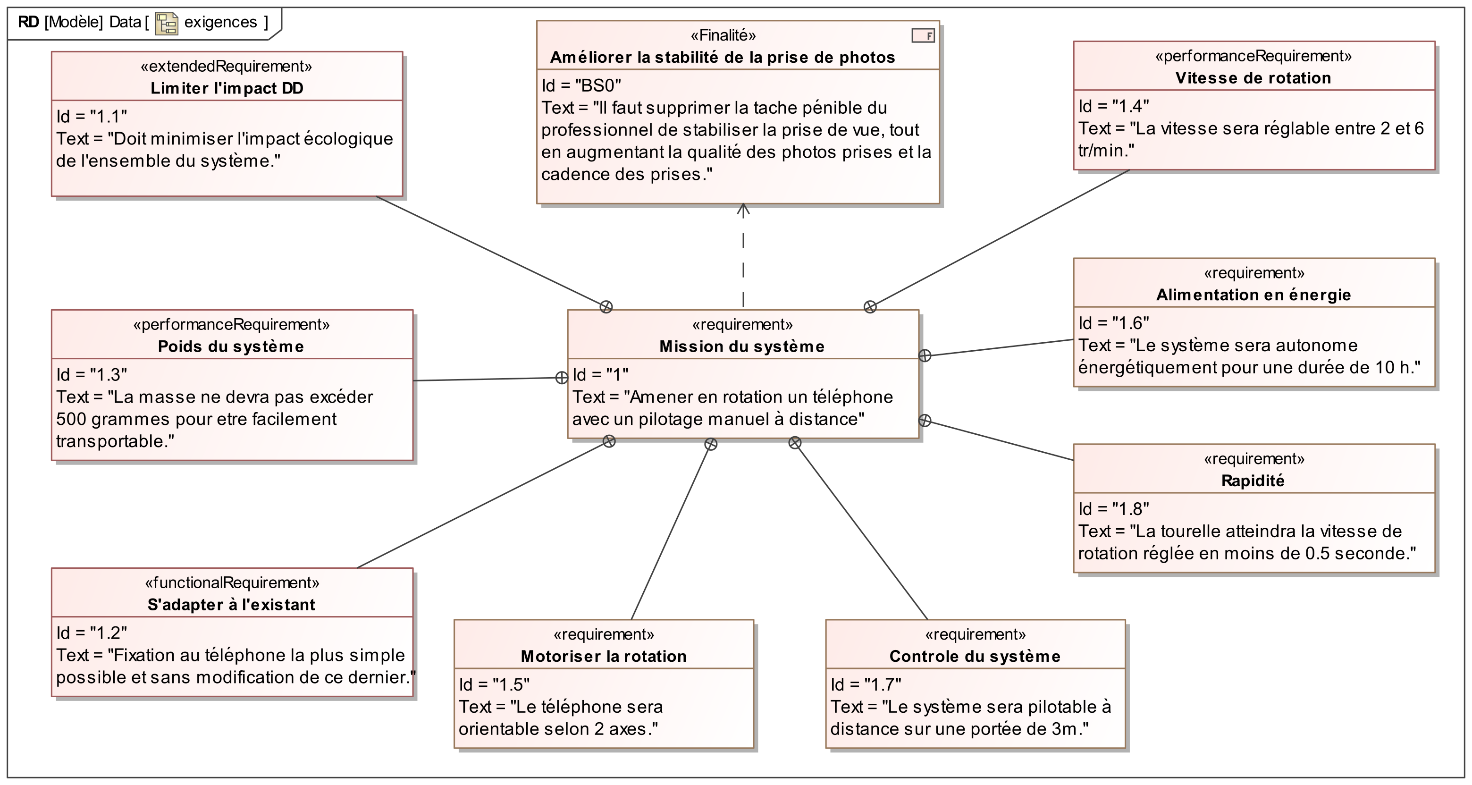
## Réaliser expérimentalement un échelon de vitesse

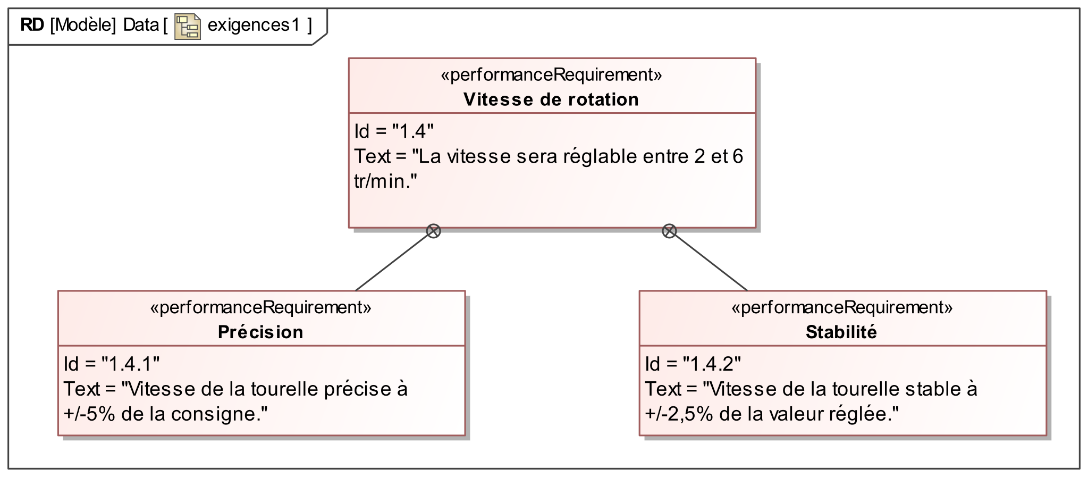
1. S’assurer que la partie précédente a été réalisée.



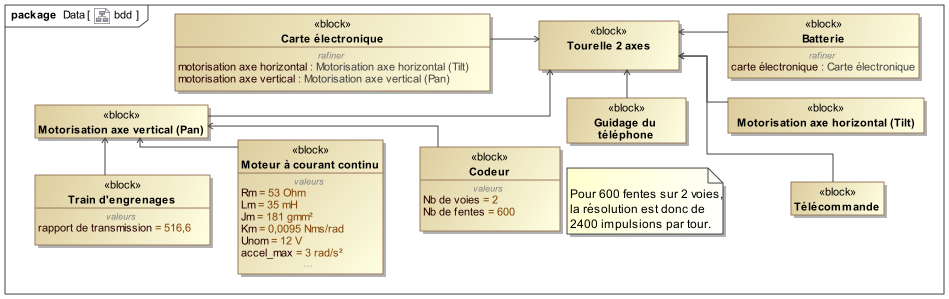
# Fiche 3 – Ingénierie Systèmes

## Diagramme des exigences





## Diagramme de définition des blocs



# Fiche 4 – Correcteurs numériques

Afin de reprogrammer le correcteur utiliser dans MyViz, il est possible d’éditer le fichier suivant.

C:\MyViz\MyVizFiles\MyViz.nw\python\scripts\TourelleDeuxAxes\correcteurs.py

**IMPORTANT**

**Avant toute modification du fichier correcteurs.py, réaliser une sauvegarde de ce fichier.**

Pour modifier le comportement du correcteur, il faut redéfini la fonction **PID**.

Il est conseillé de conserver le bloc d’instruction « application de la saturation sur la commande » pour conserver une limitation du niveau de la commande.

# Application de la saturation sur la commande

**if** **(**commande\_avant\_sat > umax**):**

commande = umax

**elif** **(**commande\_avant\_sat < umin**):**

commande = umin

**else** :

commande = commande\_avant\_sat

Ainsi, la variable à modifier à chaque pas de calcul sera la variable commande\_avant\_sat.

Il est aussi possible de stocker la valeur de l’erreur à l’instant t pour l’utiliser à l’instant suivant dans la variable

**self.erreurprec[iMoteur]**.

On peut stocker le terme intégral à l’instant présent pour l’utiliser l’instant suivant dans la variable

**correcteurs.I\_x[iMoteur]**.

Ainsi, la structure de la fonction PID sera la suivante.

**def** PID(self, iMoteur, ref, mes, Kp, Ki, Kd, Tf, umax, umin, dt2):

# Calcul de la commande avant saturation

erreur = ref - mes

# Calcul de la commande avant saturation

# A REMPLIR

# A REMPLIR

# A REMPLIR

# Application de la saturation sur la commande

**if** (commande\_avant\_sat > umax)**:**

commande = umax

**elif** (commande\_avant\_sat < umin)**:**

commande = umin

**else** :

commande = commande\_avant\_sat

# Terme intégral (sera utilisé lors du pas d'échantillonnage suivant)

correcteurs.I\_x[iMoteur] = # À déterminer si nécessaire

# Stockage de la mesure courante pour utilisation lors du pas d'échantillonnage suivant

self.erreurprec[iMoteur] = erreur

**return** commande