1. Présentation du système
   1. Introduction du gyropode :
      * Sa mission et ses objectifs : véhicule à moteur electrique servant a deplacer une personne seule (charge utile max XXX) sur un terrain type route à une vitesse max de XX) en sécurité.

Quantifier (charge utile, vitesse, temps d’arret, capacité des moteurs, etc…)

(ici mettre un ou deux schemas. Un de haut niveau qui montre le gyropode et definir le vocabulaire (le conducteur, les moteurs, la route, le guidon, la batterie, les capteurs, les roues, etc.) qui sera ensuite repris dans tout le texte. Un second plus detaillé (modèle en éclaté ?) qui introduit les elements utiles pour comprendre l’analyse technique

* + - analyse technique (explication d’angle utilisateur, freinage, exigences) comment il doit faire pour avancer, reculer, freiner, parler de l’acceleration, de l’angle, du couple, …. Rentrer dans le detail techno du fonctionnement (ici plutôt reprendre l’esprit du projet mecatronique, avec les lois d’asservissement, les boucles de courant, etc.). rentrer dans la description techno
    - scenarios et modes d’utilisation : le conducteur est debout sur la plateforme, il tient le guidon, pour avancer il se penche en avant, etc.. l’objectif pour le système est de stabiliser le gyropode (expliquer stabiliser) et de le faire se deplacer, de jour ( ya des phares ?un detecteur de luminosité ?) selon les consignes du conducteur, en garantissant la securité du conducteur (chute, vitesse max,…)

reprendre ce qu’on fait les etudiants de projet tut (mode demarrage, mode deplacement, mode arret, mode arret d’urgence, mode panne de batterie ?, etc.…). Penser a evoquer les contraintes de temps.

Ici on peut par exemple mettre un diagramme UML de use case.

C’est ici à la fin que vous allez mentionner que selon les modeles et les options on peut envisager de le verrouiller/deverrouiller a distance, le localiser, etc.) (GPS, Bluetooth)

* + - exigences fonctionnelles :
      * le gyropode surveille l’état de la batterie. Il recuperer la valeur du capteur machin toutes les XX secondes et signale a xx en deca d’un seuil, par exemple…
      * le gyropode produit une accelation de XX en reponse d’une modification de l’angle a de XXX
      * le gyropode (etc… voir votre travail plus celui fait par le projet tut)
    - Architecture physique (schema qui montre les deux blocs, les E et S des deux blocs avec l’exterieur et les échanges entre les deux) (Moteurs, capteurs, microcontrôleur (STM 32), Raspberry Pi (Superviseur) . les fonctions de ‘bas niveau’, proches du materiel (batterie, moteurs, etc…) sont controlées par le stm32 , celle de haut niveau (XXX) sont gérées par une carte rasp. Les deux controleurs sont reliés par une liaison xxx. Decrire les données qui transitent.  
      Commande du système (retour d’états –commandes, boucle de courant)  
      Architecture logique (ce qui est fait par STM 32 et RPI)

1. Contrôle en temps réel du gyro

Ici on rentre dans ce qu’il y a a faire : vous etes chargés de concevoir, d’implanter et de mettre au point un logiciel de contrôle qui va realiser les fonctions de haut niveau decrites precedemment.

Faire le lien avec ce qui precede : Faire un zoom sur le bloc d’archi de haut niveau, et reprenant les E et les sorties (nom, format), ‘ouvrir’ la boite de haut niveau pour montrer qu’on a un logiciel de contrôle (qu’on peut appeler superviseur si vous voulez) et xenomai, avec une partie TR et linux.

Reprendre ici les fonctions que ce bloc doit assurer, ou mettre des \* dans les exigences qui precedent et qui correpondent aux fonctions assurées ici. (bien souligner si executions périodiques, sporadiques, les contraintes de sequencement dans les traitements, de synchro..)

* 1. Présentation Xenomai cobalt (Co-noyau), schéma (ES de RPI)
  2. Distinction des fonctions logicielles temps réel et non temps réel pour determiner ce qui va tourner sur quoi (on ne parle pas encore des fonctions qui aident au codage et a la mise au point, interfaces graphique et log)
  3. Transmission de données (partie bas, protocole utilisé, specification des fonctions fournies
  4. Conception de l’architecture logicielle du superviseur
     + Exigences de fonctionnement --🡪 Threads

Indiquer la methode : vous devez, a partir des fonctions (section 2.0), definir un ensemble de threads (ou taches) permettant la realisation de l’ensemble de ces fonctions. Guider le decoupage : independance, parallelisme, contraintes d’activations, perdiodicité ou pas, etc…

Parler des moyens de synchro, de protection en exclu mut et de com

Donnner un exemple sur le sujet avec 2 ou 3 fonctions et comment elles sont instanciées par des taches qui communiquent entre elles. Utiliser le formalisme AADl (reprendre en annexe du tp les 2 pages synthetique de presentation de AADL du TP robot)

* 1. Validation de la conception : a partir des exigences fonctionnelles, dériver quelques scenarios de test (donner des exemples), et les executer. Verifier que le resultat obtenu correspond a celui attendu (prendre une exigence fonctinnelles et une temporelle) . poursuivre pour couvrir l’ensemble des cas d’utilisation.
  2. Correction : AADL complet (justification et conception) (ne sera pas fourni aux etudiants)

1. Prise en main du TP (cette partie ne sera distribuée qu’en TP, ce qui precede en TD)
   1. Présentation du matériel (Simulateur, LED)
   2. Interface graphique (entrées de la plaquette : batterie, potentiometre, , sorties, log, … (refaire un schema pour visualiser l’ensemble des elements de la plateforme, des photos, des copies d’ecran…)
   3. Code (commenté) à trous, compilable des la premiere seance. Ne pas mettre par exemple la tache batterie
   4. Spécification des threads avec commentaires (chaque tache, vide, contient un en tete sous forme de commentaire qui precise les fonctions qu’elle instancie, les objets qu’elle manipule (semaphores, etcc), les entrées et sorties et les interfaces vers les autres taches, les contraintes de temps, les contraintes de sequencement)
   5. Indiquer les regles de codage (nom des taches, nom des semaphores, etc., commentaires, log, …). Rappel du cote ‘OO’ : a quel niveau ? bonnes pratiques
   6. Guide de séance de TP (comment programmer et utiliser les outils, log)
   7. Indiquer peut-etre comment valider le respect des contraintes temporelles (outils informatiques ?), les outils de debug ?..
   8. Planning d’avancement prévu ? passage sur la maquette de gyro apres le simu ? qu’est ce que ca peut changer ?
   9. Grille d’évaluation : ?

Annexes techniques :

* Code a trou
* AADL
* Programmation OO, librairies fournies