**Backlog de sprint #005**

Produit : Wall-E

Conçu par : Los Santos Customs

**Équipe :** François-Xavier Allaire

Yannick André Ouamba

Claude Ajavon

Williams Roberge-Bolduc

**Échéancier :**

Début du Sprint : 23 Avril 2020

Fin du Sprint : 14 Mai 2020

**Backlog de sprint**

Sprint 5 :

Williams et Yannick ont continué le développement de la conduite automatique. Cependant , ce système nous cause toujours autant de problèmes.

L’écran au LED a été implémenté dans le code et on peut lui envoyer des messages, il suffit simplement à le connecter.

Nous avons ajouté un système permettant le rafraîchissement de l’interface en temps réel avec les données reçues par la voiture.

Le module GPS a été débranché, puisque sa précision était beaucoup trop basse et que son utilisation nous mettait beaucoup trop de bâtons dans les roues.

Claude a ajouté une Classe de tests pour les méthodes de notre modèles, pour éviter les b ugs lors de calculs.

Problèmes du Sprint 5 :

* L’auto driving a été le principal problème du Sprint 1. La voiture a apporté des comportements non-voulus.
* Contourner les obstacles a été le plus grand défi de ce sprint, et on a manqué de temps pour finaliser cette section du code.
* La voiture peut se rendre à une destination voulue s’il n’y a pas d’obstacles, mais certains obstacles nous causent des problèmes.
* La voiture reçoit les bonnes données, mais ses virages et sa détection ne sont pas assez précis.

|  |  |
| --- | --- |
| **6 : Conduite automatique de la voiture** | |
| Acteur ou rôle : | Client |
| Scénario ou story : | En tant que client, je veux utiliser la fonction de conduite automatique de la voiture afin de la déplacer à un certain endroit sans avoir à la contrôler |
| Détail ou description : | 1. Obtenir la position GPS de la voiture sous forme de coordonnées X et Y   1.1. Qui et temps  1.1.1. Y-A  1.1.2. (1h)  1.2. Préconditions  1.2.1. Avoir accès à Arduino et à ses librairies  1.3. Règles d’affaires  1.3.1. Les tests doivent êtres faits au fur et à mesure que les méthodes sont créées  1.3.2. Avoir accès à la position de la voiture  1.4. Règles d’affaires alternatives  1.4.1. Demander de l’aide à un coéquipier et si cela persiste, un soutien continu  1.5. Tests d’acceptation de cet item  1.5.1. S’assurer que peu importe la position de la voiture, avoir accès à ses coordonnées  1.6. Post-conditions  1.6.1. Finaliser les points manquants une fois que la voiture envoie ses coordonnées   1. Obtenir la distance entre la voiture et les obstacles rapprochés en utilisant les capteurs de position   2.1. Qui et temps  2.1.1. Y-A  2.1.2. (4h)  2.2. Préconditions  2.2.1. Avoir accès à Arduino et à ses librairies ainsi que les capteurs à ultrason  2.3. Règles d’affaires  2.3.1. Les tests doivent êtres faits au fur et à mesure que les méthodes sont créées  2.3.2. Pouvoir déterminer quel capteur détecte un obstacle à risque  2.4. Règles d’affaires alternatives  2.4.1. Demander de l’aide à un coéquipier et si cela persiste, un soutien continu  2.5. Tests d’acceptation de cet item  2.5.1. S’assurer que peu importe la position de l’obstacle par rapport au capteur, qu’il soit détecté  2.6. Post-conditions  2.6.1. Finaliser les points manquants une fois que la voiture détecte les obstacles à risque   1. Déterminer une vitesse par défaut pour le déplacement automatique de la voiture   3.1. Qui et temps  3.1.1. Y-A  3.1.2. (2h)  3.2. Préconditions  3.2.1. Avoir accès à Arduino et à ses librairies  3.3. Règles d’affaires  3.3.1. Les tests doivent êtres faits au fur et à mesure que les méthodes sont créées  3.3.2. Avoir accès à la vitesse de la voiture et pouvoir le gérer  3.4. Règles d’affaires alternatives  3.4.1. Demander de l’aide à un coéquipier et si cela persiste, un soutien continu  3.5. Tests d’acceptation de cet item  3.5.1. S’assurer que peu importe la position de la voiture, avoir accès à sa vitesse  3.6. Post-conditions  3.6.1. Finaliser les points manquants une fois que la voiture envoie ses coordonnées   1. Élaborer un système de conduite automatique pour se déplacer d’un point de départ (0,0) jusqu’à une coordonnée (X,Y) demandée par l’utilisateur   4.1. Qui et temps  4.1.1. Y-A & W-RB  4.1.2. (7h)  4.2. Préconditions  4.2.1. Avoir accès à Arduino et à ses librairies ainsi que Java et ses librairies  4.3. Règles d’affaires  4.3.1. Les tests doivent êtres faits au fur et à mesure que les méthodes sont créées  4.3.2. Le véhicule se rend à la coordonnée prédéterminée  4.4. Règles d’affaires alternatives  4.4.1. Demander de l’aide à un coéquipier et si cela persiste, un soutien continu  4.5. Tests d’acceptation de cet item  4.5.1. S’assurer que peu importe la position de la voiture, avoir accès à ses coordonnées et déterminer s’il s’est rendu au point désiré  4.6. Post-conditions  4.6.1. Finaliser les points manquants une fois que la voiture se rend au point   1. Développer un menu dans l’interface graphique JavaFX afin de rendre l’option de conduite automatique plus intuitive   1.1. Qui et temps  1.1.1. F-X  1.1.2. (3h)  1.2. Préconditions  1.2.1. Avoir accès à JavaFx  1.3. Règles d’affaires  1.3.1. Les tests doivent êtres faits au fur et à mesure que les méthodes sont créées  1.3.2. Les tests doivent couvrir toutes les possibilités du code, ils doivent couvrir préférablement 100% des branches atteintes par le code  1.4. Règles d’affaires alternatives  1.4.1. Demander de l’aide à un coéquipier et si cela persiste, un soutien continu  1.5. Tests d’acceptation de cet item  1.5.1. S’assurer que le menu est facilement lisible, clair et simple  1.6. Post-conditions  1.6.1. Finaliser les points manquants une fois que l’interface est adéquat   1. Rédiger un menu d’aide à l’utilisateur pour expliquer comment fonctionne le système   6.1. Qui et temps  6.1.1. F-X  6.1.2. (4h)  6.2. Préconditions  6.2.1. Avoir accès à JavaFx  6.3. Règles d’affaires  6.3.1. Les tests doivent êtres faits au fur et à mesure que les méthodes sont créées  6.3.2. Les tests doivent couvrir toutes les possibilités du code, ils doivent couvrir préférablement 100% des branches atteintes par le code  6.4. Règles d’affaires alternatives  6.4.1. Demander de l’aide à un coéquipier et si cela persiste, un soutien continu  6.5. Tests d’acceptation de cet item  6.5.1. S’assurer que le menu est facilement lisible, clair et simple  6.6. Post-conditions  6.6.1. Finaliser les points manquants une fois que l’interface est adéquat   1. Permettre à la voiture de contourner un obstacle à l’aide d’une intelligence artificielle construite avec des classes métier Java   7.1. Qui et temps  7.1.1. Y-A & W-RB  7.1.2. (1h)  7.2. Préconditions  7.2.1. Avoir accès à Arduino et à ses librairies et à Java et à ses librairies  7.3. Règles d’affaires  7.3.1. Les tests doivent êtres faits au fur et à mesure que les méthodes sont créées  7.3.2. Détecter et éviter les obstacles à risque  7.4. Règles d’affaires alternatives  7.4.1. Demander de l’aide à un coéquipier et si cela persiste, un soutien continu  7.5. Tests d’acceptation de cet item  7.5.1. S’assurer que peu importe l’environnement de la voiture, qu’il puisse détecter un obstacle à risque  7.6. Post-conditions  7.6.1. Finaliser les points manquants une fois que la voiture évite les obstacles à risque   1. Prévoir une action réalisée par la voiture pour signifier qu’elle a bel et bien atteint sa position   1.1. Qui et temps  1.1.1. Y-A & F-X  1.1.2. (3h)  1.2. Préconditions  1.2.1. Avoir accès à Arduino et à ses librairies et à Java et à ses librairies  1.3. Règles d’affaires  1.3.1. Les tests doivent êtres faits au fur et à mesure que les méthodes sont créées  1.3.2. Avoir accès aux fonctionnalitées à la voitures  1.4. Règles d’affaires alternatives  1.4.1. Demander de l’aide à un coéquipier et si cela persiste, un soutien continu  1.5. Tests d’acceptation de cet item  1.5.1. S’assurer que lorsque la voiture atteint son objectif, une action est réalisée à cet effet  1.6. Post-conditions  1.6.1. Finaliser les points manquants une fois que la voiture réagit lorsqu’elle a atteint son objectif |
| Tests d’acceptation : | Utiliser la fonction Auto-Driving dans plusieurs scénarios avec différents obstacles, comme des murs, des boîtes, rampes,etc. |
| Complexité : | 9 |
| Effort : | 14j/homme |
| Commentaires : |  |

|  |  |
| --- | --- |
| **12 : Contrôle de la voiture et de la caméra** | |
| Acteur ou rôle : | Client |
| Scénario ou story : | En tant que client, je veux une voiture qui se contrôle bien et qui propose des contrôles solides et intuitifs |
| Détail ou description : | 1. Contrôle de vitesse précis   1.1. Qui et temps  1.1.1. Y-A  1.1.2. (3h)  1.2. Préconditions  1.2.1. Avoir accès à la voiture, à une manette de jeu et à l’application Java Wall-E  1.3. Règles d’affaires  1.3.1. Pouvoir contrôler la vitesse de la voiture en ajustant la pression que l’on met sur le ‘Joystick’  1.3.2.. Avoir une vitesse maximale et minimale par défaut sur la voiture  1.4. Règles d’affaires alternatives  1.4.1. Avoir une vitesse fixe sur la voiture, sans avoir de possibilité de l’ajuster  1.5. Tests d’acceptation de cet item  1.5.1. Vérifier si on est capables d’obtenir plusieurs vitesses différentes selon la pression que l’on applique sur une manette (Joystick)  1.6. Post-conditions  1.6.1. Essayer d’ajouter le virage intégré avec le mouvement, pour permettre à la voiture de réaliser plusieurs action en même temps   1. Virage intégré avec le mouvement   2.1. Qui et temps  2.1.1. Y-A  2.1.2. (2h)  2.2. Préconditions  2.2.1. Avoir accès à la voiture, à une manette de jeu et à l’application Java Wall-E  2.3. Règles d’affaires  2.3.1. Faire en sorte que le système puisse envoyer deux commandes en même temps pour permettre de tourner en même temps que d’avancer  2.4. Règles d’affaires alternatives  2.4.1. Laisser les deux fonctionnalités à part : ne pas permettre le déplacement synchronisé avec le virage  2.5. Tests d’acceptation de cet item  2.5.1. S’assurer que peu importe la vitesse de la voiture en avançant, elle peut tournier fluidement  2.6. Post-conditions  2.6.1. On peut maintenant mettre l’attention principale sur le système de conduite automatique de la voiture   1. Réitération du câblage de la voiture et réinstallation des modules   3.1. Qui et temps  3.1.1. Y-A  3.1.2. (h)  3.2. Préconditions  3.2.1. Avoir accès aux modules de la voiture ainsi qu’à leurs fils de branchement  3.3. Règles d’affaires  3.3.1. Fixation des capteurs de distance à leurs endroits respectifs  3.3.2. Branchement de la caméra de façon à ce que le fil n’entre pas en collision  3.4. Règles d’affaires alternatives  3.4.1. Demander de l’aide à un coéquipier et si cela persiste, un soutien continu  3.5. Tests d’acceptation de cet item  3.5.1. Mettre en ordre les fils sur la voiture afin de limiter l’obstruction occasionné  3.6. Post-conditions  3.6.1. Finaliser les points manquants une fois que la voiture se rend au point   1. Fixation des différents modules   4.1. Qui et temps  4.1.1. Y-A  4.1.2. (3h)  4.2. Préconditions  4.2.1. Avoir accès aux modules et à la voiture  4.3. Règles d’affaires  4.3.1. Tout doit être connecté et prêt à paramétrer  4.3.2. Les modules doivent être fixés adéquatement  4.4. Règles d’affaires alternatives  4.4.1. Demander de l’aide à un coéquipier et si cela persiste, un soutien continu  4.5. Tests d’acceptation de cet item  4.5.1. S’assurer que toutes les modules sont fonctionnels et bien fixés  4.6. Post-conditions  4.6.1. Finaliser les points manquants une fois que l’interface est adéquat   1. Contrôle intégré de la caméra   1.1. Qui et temps  1.1.1. Y-A  1.1.2. (3h)  1.2. Préconditions  1.2.1. Avoir accès à Arduino  1.3. Règles d’affaires  1.3.1. Le comportement de la caméra doit être contrôlé entièrement par l’utilisateur  1.4. Règles d’affaires alternatives  1.4.1. Demander de l’aide à un coéquipier et si cela persiste, un soutien continu  1.5. Tests d’acceptation de cet item  1.5.1. S’assurer que le comportement de la caméra est selon l’ordre de l’utilisateur  1.5.2. La caméra s’oriente fluidement  1.6. Post-conditions  1.6.1. Finaliser les points manquants une fois que la caméra se contrôle facilement |
| Tests d’acceptation : | S’assurer que la voiture puisse tourner en avançant, et que les modules soit tous fixés et stables |
| Complexité : | 1 |
| Effort : | 14j/homme |
| Commentaires : |  |

|  |  |
| --- | --- |
| **10 Affichage de données avec un écran LED** | |
| Acteur ou rôle : | Client |
| Scénario ou story : | En tant que client, je veux avoir la possibilité de personnaliser les visuels affichés sur l’écran LED afin de rendre le robot unique |
| Détail ou description : | 1. Faire en sorte que l’écran LED reçoit les messages   1.1. Qui et temps  1.1.1. Y-A  1.1.2. (3h)  1.2. Préconditions  1.2.1. Avoir accès à Arduino  1.3. Règles d’affaires  1.3.1. L’écran LED doit pouvoir recevoir des messages codés  1.4. Règles d’affaires alternatives  1.4.1. Demander de l’aide à un coéquipier et si cela persiste, un soutien continu  1.1. Tests d’acceptation de cet item  1.1.1. S’assurer que les messages soient bien reçus par l’écran LED et qu’il peut effectivement afficher des messages qui lui sont envoyés  1.6. Post-conditions  1.6.1. On peut maintenant connecter cette section au code Java   1. Connecter l’écran LED   1.1. Qui et temps  1.1.1. Y-A  1.1.2. (1h)  1.2. Préconditions  1.2.1. Avoir accès à La voiture  1.3. Règles d’affaires  1.3.1. L’écran LED doit pouvoir s’allumer en utilisant la voiture comme source électrique  1.4. Règles d’affaires alternatives  1.4.1. Demander de l’aide à un coéquipier et si cela persiste, un soutien continu  1.1. Tests d’acceptation de cet item  1.1.1. S’assurer que l’écran fonctionne bien, et qu’on peut s’y connecter par Wifi  1.6. Post-conditions  1.6.1. On peut maintenant essayer d’envoyer des messages à partir du code Java   1. Afficher Les messages à l’aide de l’écran LED   1.1. Qui et temps  1.1.1. Y-A  1.1.2. (1h)  1.2. Préconditions  1.2.1. Avoir accès à La voiture et à Arduino et au code java  1.3. Règles d’affaires  1.3.1. L’écran LED doit pouvoir afficher des messages textes tirés par le code java  1.4. Règles d’affaires alternatives  1.4.1. Demander de l’aide à un coéquipier et si cela persiste, un soutien continu  1.1. Tests d’acceptation de cet item  1.1.1. S’assurer que l’écran affiche bien les messages que l’on demande à partir du code Java  1.6. Post-conditions  1.6.1. On peut essayer de connecter un haut-Parleur(facultatif) |