III - Spécifications fonctionnelles

Fonction 1 : Modification Locale de la Trajectoire

Exigence ER-1 - Tâche PGE R2 - Use Case 4

Description

La fonction de modification locale de la trajectoire permet au robot Tiago de réagir de manière proactive aux obstacles détectés lors de la navigation autonome. Elle intervient après la planification de trajectoire initiale et assure que le robot évite les obstacles statiques et dynamiques en ajustant sa trajectoire en moins d'une seconde.

Entrées

- **Trajectoire Planifiée :** La trajectoire générée lors de la planification de trajectoire initiale est fournie en entrée. Cette trajectoire représente le chemin prévu du robot de la position de départ à la destination.
- **Données de Capteurs :** Les données des capteurs (Lidar, Surround View) sont fournies en entrée et fournissent des informations sur les obstacles statiques et dynamiques à proximité du robot à une fréquence idéale de 10hz.

Exigences

1. **Détection d'Obstacles :** La fonction doit surveiller en temps réel les données des capteurs pour détecter et différencier les obstacles statiques et dynamiques sur la trajectoire planifiée.
2. **Réaction aux Obstacles :** En cas de détection d'obstacles, la fonction doit être capable d'ajuster la trajectoire du robot en temps réel pour éviter les collisions. Les ajustements de trajectoire doivent être effectués de manière à minimiser les retards. Dans le cas où l'évitement est impossible, le robot s'arrête et envoie une image au logiciel superviseur.
3. **Priorité des Obstacles :** La fonction doit prendre en compte la priorité des obstacles en fonction de leur type (humain, robot, objet statique) pour déterminer la nature des ajustements de trajectoire.

Sorties

- **Trajectoire Modifiée :** En fonction des ajustements nécessaires pour éviter les obstacles, la fonction génère une nouvelle trajectoire, modifiée par rapport à la trajectoire initiale, qui guide le robot en toute sécurité vers sa destination tout en évitant les obstacles.

Contraintes

- La réaction aux obstacles doit être effectuée en moins **d'une seconde** pour garantir la sécurité du robot et des utilisateurs.
- Les ajustements de trajectoire doivent être effectués en minimisant les retards et en tenant compte des contraintes cinématiques du robot Tiago.

Scénarios d'Utilisation

1. **Navigation Autonome avec Obstacles Dynamiques :** Lorsque le robot se déplace le long de la trajectoire planifiée et détecte des obstacles dynamiques sur son chemin, la fonction de modification locale de la trajectoire ajuste la trajectoire pour éviter les obstacles sans interrompre la navigation.
2. **Évitement des Obstacles Statiques :** En cas de détection d'obstacles statiques non pris en compte lors de la planification de trajectoire initiale, la fonction ajuste la trajectoire pour les éviter en temps réel.

Fonction 1 : Mise en service du système Surround View

Exigence ER-6 ER-7 - Tâche PGE V1-1 T3 - Use Case 3

Description

Cette fonction vise à installer et calibrer le système Surround View sur le robot TIAGo. Le système Surround View utilise plusieurs caméras pour fournir une vision panoramique de l'environnement du robot. L'objectif est de s'assurer que le système est correctement installé et calibré pour garantir des images de haute qualité et une perception précise de l'environnement.

Entrées

- **Système Surround View :** L'ensemble du matériel nécessaire pour le système, comprenant plusieurs caméras et composants électroniques.
- **Robot TIAGo :** Le robot qui sera équipé du système Surround View.

Exigences

1. **Angles morts :** Le processus d'installation doit prendre en compte les angles morts du système afin de les minimiser.
2. **Champ de vision global :** L'installation doit faire en sorte que le champ de vision global du système Surround View recouvre efficacement l'environnement dans lequel le robot évolue.
3. **Recouvrement des champs de vision :** Un recouvrement adéquat entre les champs de vision des différentes caméras doit être assuré pour éviter les zones non couvertes.

Sorties

- **Système Surround View installé et calibré :** À la fin de la fonction, le système doit être correctement installé et calibré, prêt à être utilisé pour la navigation et la perception de l'environnement.
- **Images issues du système Surround View :** Des images de l'environnement capturées par le système Surround View doivent être générées.

Contraintes

- L'installation doit être effectuée de manière à garantir une stabilité maximale du système, minimisant les vibrations ou les mouvements indésirables susceptibles d'affecter la qualité des images.
- Protection Anti-choc : Le système doit être protégé contre les chocs potentiels pendant le fonctionnement du robot, assurant ainsi la durabilité et la calibration du matériel.

Fonction 2 : Création d'une base de données

Exigence ER-2 - Tâche PGE V1-2 - Use Case 3

Description

Cette fonction concerne la création d'une base de données annotée à partir des images capturées par le système Surround View. L'objectif est de générer une base de données représentative de l'utilisation future du système, avec un équilibrage approprié des classes pour soutenir les tâches d'apprentissage automatique.

Entrées

- **Images issues du système Surround View :** Ensemble d'images capturées par le système Surround View.

Exigences

1. **Volume de données :** La base de données doit stocker un volume adapté d'images annotées, nécessaire pour l'apprentissage d'un modèle de classification.
2. **Représentative de l'utilisation future :** La base de données doit être représentative des scénarios d'utilisation futurs du système, couvrant une diversité de situations probables.
3. **Équilibrage des classes :** Les différentes classes d'objets ou de scénarios doivent être équilibrées dans la base de données pour éviter tout biais d'apprentissage.

Sorties

❖ **Base de données annotée :** La base de données résultante doit contenir des images annotées, prêtes à être utilisées pour l'entraînement et l'évaluation du modèle d'apprentissage automatique.

Contraintes

❖ Les images utilisées pour créer la base de données doivent provenir du système Surround View pour conserver les distorsions spécifiques à ce système entre les phases d'apprentissage, d'évaluation et d'utilisation.

❖ Une copie de sauvegarde de la base de données doit être maintenue pour garantir la disponibilité et éviter la perte de données en cas de défaillance du système principal.

Fonction 3 : Sélection et entraînement du modèle

Exigence ER-2 - Tâche PGE V1-3 - Use Case 3

Description

Sélectionner ou créer un modèle qui peut classifier les différences entre agents (peut-être des robots ou des utilisateurs) et obstacles statiques. Entraîner le modèle sélectionné avec une base de données pour qu'il puisse précisément identifier et distinguer les agents dans le contexte pour lequel il est conçu.

Entrées

❖ **Etude comparative :** L'état de l'art inclut une analyse comparative des algorithmes de vision par ordinateur et des techniques d'apprentissage en profondeur spécialisées dans la classification des agents et des obstacles. Cette étude visa à sélectionner la méthode la plus efficace et la plus appropriée pour notre contexte spécifique de navigation robotique. Les critères d'évaluation incluent la précision, la vitesse de traitement, la complexité du modèle, et l'adaptabilité aux environnements changeants.

❖ **Base de données :** La base de données exhaustive doit comprendre des images annotées de divers agents (humains) et d'obstacles statiques (murs, barrières, chariots, caisses). La base de données sera utilisée pour entraîner et valider le modèle de détection. Une attention particulière sera accordée à la diversité des scénarios pour assurer une généralisation robuste du modèle. Des modifications du modèle de détection pourront être effectuées pour améliorer la précision et la vitesse de traitement en fonction des résultats des tests initiaux.

Exigences

1. **Détection d'Obstacles :** la plateforme mobile doit être capable de détecter des objets d'intérêt dans son environnement qui seraient susceptibles d'interférer avec la trajectoire planifiée par cette dernière. Le robot doit ainsi pouvoir reconnaître deux différentes classes d'objets via le système Surround View : Les humains, représentant les obstacles dynamiques, et les chariots/caisses/outils représentant les obstacles statiques. Ainsi, le robot doit être capable de détecter ces objets et de déterminer si ces derniers sont susceptibles de rentrer en collision avec la plateforme.

Sorties

❖ **Modèle:** Le modèle produit des étiquettes précises pour les obstacles et agents, avec leurs positions et probabilités de détection, facilitant une réaction immédiate et une navigation sûre pour les robots en environnement variable.

Contraintes

❖ Le temps de traitement d'une image doit être inférieur à 10ms.

Fonction 4 : Intégration du modèle

Exigence ER-2 EE-2 - Tâche PGE V1-4 - Use Case 8, 7

Description

L'intégration du modèle en temps réel vise à implémenter le modèle de détection de manière à ce qu'il soit capable de fonctionner avec un haut degré de précision et d'efficacité dans un environnement temps réel (i.e. un temps de réaction inférieur à 100 ms). Le système doit pouvoir reconnaître deux différentes classes d'objets via le système Surround View : Les humains, représentant les obstacles dynamiques, et les chariots/caisses/outils, représentant les obstacles statiques.

Entrées

❖ **Modèle opérationnel :** Le modèle fonctionne sur image

❖ **Flux vidéo :** Flux vidéo en continue issu des caméras

Exigences

1. **Rapidité d'exécution :** Cette intégration nécessite une attention particulière aux performances, à la fiabilité et à l'interopérabilité du modèle avec les systèmes existants.

Sorties

❖ **Modèle intégré :** Modèle qui analyse en continu le flux vidéo renvoyant les informations sur les obstacles.

Contraintes

❖ Le temps de traitement d'une image doit être inférieur à 10 ms.

Fonction 1000 : Acquérir les données de l'environnement robotique

Exigence ER-1 - Tâche PGE R1-1 - Use Case 4

Description

La fonction R1-1 vise à acquérir des données sur l'environnement à l'aide des capteurs du robot, tels que le Lidar et les capteurs ultrasons. Ces données brutes seront ensuite transformées dans le repère monde de la carte, fournissant ainsi des informations exploitables pour la planification de trajectoire.

Entrées

1. Robot TIAGo : Le robot équipé de capteurs, notamment le Lidar et les capteurs ultrasons.

Exigences

1. **Fiabilité des données** : Les données acquises doivent être fiables et refléter avec précision l'environnement réel du robot.
2. **Fréquence d'acquisition** : La fréquence d'acquisition des données doit être suffisante pour permettre une réactivité appropriée du robot aux changements environnementaux.

3. **Compatibilité avec la carte** : La transformation des données doit garantir la compatibilité avec la carte de l'environnement existante, assurant ainsi une intégration fluide avec le système de navigation.
4. **Maintenance des capteurs** : Un mécanisme de surveillance de l'état des capteurs doit être mis en place, signalant toute défaillance ou besoin de maintenance.

Sorties

1. **Données brutes capteurs exploitables** : Les données transformées dans le repère monde de la carte, prêtes à être utilisées pour la planification de trajectoire.

Contraintes

❖ **État des capteurs** : Les capteurs du robot doivent être opérationnels pour garantir la collecte de données fiables.

Scénarios d'Utilisation

1. **Acquisition en temps réel** : Lorsque le robot TIAGo est en fonctionnement, la fonction R1-1 est activée pour acquérir continuellement des données sur l'environnement à l'aide des capteurs.
2. **Intégration avec la planification de trajectoire** : Les données brutes capteurs exploitables sont utilisées comme entrée pour le traitement et filtrage des données, contribuant ainsi à une navigation autonome et sécurisée du robot dans son environnement connu.

Fonction 1000 : Traiter les données de l'environnement robotique

Exigence ER-1 - Tâche PGE R1-2 - Use Case 4

Description

La fonction R1-2 a pour objectif de traiter les données brutes acquises à partir des capteurs du robot, en filtrant ces données pour éliminer le bruit et en détectant les obstacles présents dans l'environnement. Les résultats de ce traitement seront utilisés pour contribuer à la planification de trajectoire autonome du robot.

Entrées

1. **Données brutes capteurs exploitables R1-1** : Les données transformées dans le repère monde de la carte, issues de la fonction R1-1, prêtes à être traitées.
2. **Carte de l'environnement statique R0-2** : La représentation cartographique statique de l'environnement connu par le robot.

Exigences

1. **Efficacité du filtre** : Le filtre appliqué aux données brutes doit être efficace pour éliminer le bruit sans compromettre la précision des informations.
2. **Corrélation avec la carte** : La détection d'obstacles doit être cohérente avec la carte de l'environnement statique, assurant ainsi la pertinence des informations dans le contexte de la planification de trajectoire.
3. **Adaptabilité aux changements** : Le traitement des données doit être suffisamment adaptable pour prendre en compte les changements dynamiques dans l'environnement.

Sorties

1. **Données brutes capteurs exploitables** : Les données transformées dans le repère monde de la carte, prêtes à être utilisées pour la planification de trajectoire.

Contraintes

❖ **Précision de la détection** : La détection d'obstacles doit être précise pour garantir la sécurité du robot et éviter les collisions.

Scénarios d'Utilisation

1. **Réaction aux obstacles** : Les résultats de la détection d'obstacles sont utilisés pour ajuster la trajectoire du robot, assurant ainsi son évitement en temps réel.
2. **Intégration avec la planification de trajectoire** : Les informations de détection d'obstacles contribuent à la génération d'une trajectoire sûre et optimale, en tenant compte des obstacles identifiés dans l'environnement.

Fonction 11 : Localiser dans l'environnement

Exigence ER-4, ER-5 - Tâche PGE R2-1,R0-2 - Use Case 4

Description

La fonction "Localiser le robot dans l'environnement" permet au robot Tiago d'estimer sa position dans son environnement en fonction des nouvelles données acquises après les avoir filtrées. Elle intervient tout du long de l'utilisation du robot.

Entrées

❖ Carte de l'environnement statique.

❖ Données brutes exploitables des capteurs de perception.

Exigences

1. **Précision de la localisation :** La localisation du robot doit permettre de bien le situer par rapport à l'environnement et implique donc un niveau de précision relativement élevé.

Sorties

❖ **État du robot :** L'état du robot constitué de sa position et de son orientation par rapport au repère de l'environnement.

Contraintes

❖ La localisation doit se faire en moins de 10 ms pour garantir la sécurité du robot et des utilisateurs.

Scénarios d'Utilisation

1. **Robot à l'arrêt :** Sa position est stable sur la carte.
2. **Exécution d'une commande (à la manette ou via une trajectoire) :** Le robot met à jour son état et met à jour sa position sur la carte de l'environnement.

Fonction 12 : Récupérer toutes les données et planifier la trajectoire du robot

Exigence ER2-2 - Tâche PGE R2 - Use Case 4

Description

La fonction permet au robot Tiago d'effectuer une planification de trajectoire autonome permettant d'éviter uniquement les obstacles connus de l'environnement , ce qui est essentiel pour une navigation efficace et sécurisée dans un environnement connu d'avance. Cette fonction implique la capacité du robot à planifier et exécuter la trajectoire depuis le point A au point B à l'aide de la cartographie existante de l'environnement.

Entrées

❖ **Données de capteurs :** Les données des capteurs (Lidar, Surround View) sont fournies en entrée et fournissent des informations sur les obstacles statiques et dynamiques à proximité du robot.

❖ **Carte de l'Environnement :** À partir d'une représentation cartographique de l'environnement existant, le robot pourra se localiser dans l'environnement.

Exigences

1. **Algotirthme adapté :** L'algorithme doit être adapté à l'environnement.
2. **Exigence 2 :** La trajectoire planifiée ne doit entrer en contact avec aucun obstacle connu de l'environnement et doit être la plus optimale depuis le point de départ jusqu'au point d'arrivée.

Sorties

❖ **Trajectoire initiale :** La trajectoire initiale guide le robot de la position de départ (point A) à la destination (point B) tout en aillant le chemin optimal vers sa destination.

Contraintes

❖ **Contraintes Matérielles :** Les capteurs du robot doivent être en état de fonctionnement et fournir des données fiables. Les capacités de calcul du robot doivent être suffisantes pour exécuter les algorithmes de planification.

❖ **Contraintes Environnementales :** L'environnement est supposé fixe et connu. Toute modification de ce dernier devra obligatoirement être suivi d'une mise à jour de la carte.

❖ **Contraintes Cinématiques du Robot :** La planification de trajectoire doit respecter les contraintes cinématiques du robot, telles que sa vitesse maximale, son rayon de braquage, etc.

❖ **Sécurité :** La priorité doit être donnée à la sécurité. La trajectoire planifiée doit garantir l'évitement d'obstacles pour éviter les collisions.

❖ **Contraintes de Conception :** Les contraintes de conception du robot, telles que la consommation d'énergie, la taille et le poids du chargement, peuvent influencer l'exécution des trajectoires.

Scénarios d'Utilisation

1. **Navigation d'un point A à un point B :**

L'utilisateur va demander au robot d'aller à un point dans l'environnement connu du robot. Puis en prenant en compte de sa position actuelle, il va générer une trajectoire permettant au robot de se déplacer de son point de départ (point A) à sa destination (point B), grâce à la carte de l'environnement existante et connu par le robot.

Fonction 13 : Exécuter la trajectoire

Étape 1 : Calculer les commandes à envoyer aux actionneurs

Exigence 6 - Tâche PGE R2-4 - Use Case 1

Description

Le contrôleur robotique reçoit les états futurs échantillonnés sur la trajectoire fournis en entrée. Le contrôleur calcule la commande à envoyer au actionneur pour effectuer un déplacement de l'état courant à l'état suivant.

Entrées

❖ **Trajectoire Planifiée :** La trajectoire générée lors de la planification de trajectoire est fournie en entrée. Cette trajectoire représente le chemin prévu du robot de sa position actuelle à sa destination.

❖ **Données de Capteurs :** Les données des codeurs des roues du robot sont récupérées en entrée et fournissent des informations sur l'actionnement des moteurs du robot.

❖ **État du robot :** Position et vitesse du robot.

Exigences

1. **Les contrôleurs doivent être stables.**
2. **Les contrôleurs doivent respecter les contraintes temps réel.**
3. **Commande échantillonnée respectant les contraintes temps réel :** Le calcul de la commande ne doit pas être bloquant pour le système robotique.

Sorties

❖ **Contrôleurs fonctionnels :** Le robot exécute la commande envoyée et respecte la consigne en position et en vitesse.
❖ **Commande envoyée aux actionneurs :** Position et vitesse des deux actionneurs (roue) du robot.

Contraintes

❖ Le contrôleur doit être correctement paramétré pour minimiser l'accumulation des erreurs.

❖ Le contrôleur doit respecter les contraintes temps réel.

❖ La commande doit être calculée en "temps réel" (renseigner la fréquence ici ?).

❖ La commande doit respecter les limites matérielles (vitesse maximale des actionneurs) et les limites imposées par l'équipe projet pour garantir la sécurité du matériel (vitesse maximale du robot en fonctionnement).

Scénarios d'Utilisation

1. **Exécution d'une commande en position :** Avancer en ligne droite d'un mètre.
2. **Exécution d'une commande en vitesse :** Avancer en ligne droite à 0,5 m/s pendant 2s.
3. **Exécution d'une trajectoire simple :** Le robot calcule la commande à chaque instant pour exécuter la trajectoire fournie en entrée.

Étape 2 : Vérifier l'exécution de la commande (sécurité)

Description

La fonction a pour objectif de vérifier l'exécution de la commande envoyée au robot.

Entrées

❖ **Odométrie :** Avec les codeurs montés sur les actionneurs du robot, il est possible de réaliser une estimation du déplacement réalisé par le robot.
❖ **Nouvelle localisation du robot pour estimer le déplacement réalisé par le robot.**
❖ **Commande :** La commande envoyée au robot pour la comparer avec la trajectoire réalisée par le robot.

Exigences

1. **Générer un message erreur :** Si la consigne n'est pas suivie par le robot, celui-ci doit s'arrêter et signaler un défaut de fonctionnement.

Sorties

❖ **Commande corrigée :** Si le robot n'exécute pas et/ou mal la consigne, le contrôleur robotique corrige la commande.
❖ **Message d'erreur :** Si le robot est en défaut, signaler l'erreur aux opérateurs.

Contraintes

❖ Une erreur sur l'exécution de la commande ne doit pas compromettre l'intégrité du robot, ni de l'environnement et ni des opérateurs.

Scénarios d'Utilisation

1. **La commande a bien été exécutée par le robot :** La commande est exécutée par le robot, il n'y a pas d'erreur ou de dérive. L'exécution continue
2. **La commande n'a pas, ou mal, été exécutée par le robot :** La commande n'est pas suivie par le robot, le contrôleur robotique corrige l'erreur.
3. **La commande n'a pas, ou mal, été exécutée par le robot :** La commande n'est pas suivie par le robot, le contrôleur robotique ne corrige pas l'erreur, le robot s'arrête et se met en défaut.

Fonction 1 : Modification Locale de la Trajectoire

Exigence ER-1 - Tâche PGE R2-3 - Use Case 4

Description

La fonction de modification locale de la trajectoire permet au robot Tiago de réagir de manière proactive aux obstacles détectés lors de la navigation autonome. Elle intervient après la planification de trajectoire initiale et assure que le robot évite les obstacles statiques et dynamiques en ajustant sa trajectoire en moins d'une seconde.

Entrées

❖ **Trajectoire Planifiée :** La trajectoire générée lors de la planification de trajectoire initiale est fournie en entrée. Cette trajectoire représente le chemin prévu du robot de la position de départ à la destination.

❖ **Données de Capteurs :** Les données des capteurs (Lidar, Surround View) sont fournies en entrée et fournissent des informations sur les obstacles statiques et dynamiques à proximité du robot à une fréquence idéale de 10hz.

Exigences

1. **Détection d'Obstacles :** La fonction doit surveiller en temps réel les données des capteurs pour détecter et différencier les obstacles statiques et dynamiques sur la trajectoire planifiée.
2. **Réaction aux Obstacles :** En cas de détection d'obstacles, la fonction doit être capable d'ajuster la trajectoire du robot en temps réel pour éviter les collisions. Les ajustements de trajectoire doivent être effectués de manière à minimiser les retards. Dans le cas où l'évitement est impossible, le robot s'arrête et envoie une image au logiciel superviseur.
3. **Priorité des Obstacles :** La fonction doit prendre en compte la priorité des obstacles en fonction de leur type (humain, robot, objet statique) pour déterminer la nature des ajustements de trajectoire.

Sorties

❖ **Trajectoire Modifiée :** En fonction des ajustements nécessaires pour éviter les obstacles, la fonction génère une nouvelle trajectoire, modifiée par rapport à la trajectoire initiale, qui guide le robot en toute sécurité vers sa destination tout en évitant les obstacles.

Contraintes

❖ La réaction aux obstacles doit être effectuée en moins **d'une seconde** pour garantir la sécurité du robot et des utilisateurs.

❖ Les ajustements de trajectoire doivent être effectués en minimisant les retards et en tenant compte des contraintes cinématiques du robot Tiago.

Scénarios d'Utilisation

1. **Navigation Autonome avec Obstacles Dynamiques :** Lorsque le robot se déplace le long de la trajectoire planifiée et détecte des obstacles dynamiques sur son chemin, la fonction de modification locale de la trajectoire ajuste la trajectoire pour éviter les obstacles sans interrompre la navigation.
2. **Évitement des Obstacles Statiques :** En cas de détection d'obstacles statiques non pris en compte lors de la planification de trajectoire initiale, la fonction ajuste la trajectoire pour les éviter en temps réel.

Fonction 1: Alerter les agents

Exigence EI-1 - Informer les agents se trouvant sur la trajectoire du chariot de sa trajectoire

Exigence EE-2 - L'application doit fonctionner en temps réel

Exigence EE-3 - Codage en C++

Description

La fonction « Alerter les agents » représente l'interface externe de l'IHM de notre projet. Son objectif est d'assurer la communication entre les agents opérant dans l'environnement du robot et le robot lui-même. Cette fonction revêt une importance cruciale dans la garantie de la sécurité des agents, tout en permettant au robot d'opérer dans des conditions optimales.

Entrées

❖ **Trajectoire planifiée** : La trajectoire estimée lors de la planification de trajectoire du robot.
❖ **Position du robot** : La position en temps réel du robot.
❖ **Position du poste de l'agent** : La position du poste de travail de l'agent.

Exigences

❖ **Informer les agents** se trouvant sur la trajectoire du chariot de sa trajectoire
❖ **Langage de programmation** : Codage en C++
❖ **Temps réel** : L'application doit fonctionner en temps réel et mettre à jour les informations affichées à une fréquence de 10 Hz

Sortie

❖ **Logiciel Agent** : Affichage graphique des différentes données en entrée (position du poste de travail, position temps-réel du robot, trajectoire estimée du robot)

Contraintes

❖ **Environnement industriel bruyant** (usine, entrepôt)
❖ **Attention des agents** au travail non focalisée sur la surveillance du robot

Scénario d'utilisation

Le robot reçoit l'instruction de se déplacer du point A au point B de l'entrepôt. Il se met en mouvement, et tous les écrans aux postes de travail affichent instantanément la trajectoire estimée du robot, sa position en temps réel, ainsi que la position du poste de travail correspondant. Un opérateur remarque que le robot s'approche de son poste, visualisant cette information sur l'écran de son poste de travail. Il intensifie alors son attention et sa vigilance pour assurer un passage sans incident.

Fonction 1 : Logiciel superviseur

Exigence EE-3 - Tâche PGE I1-2 - Codage en C++ Exigence EE-2 - Tâche PGE I1-3 - L'application doit fonctionner en temps réel Exigence EI-2 - Tâche PGE I1-3 - Implémenter une communication avec un opérateur distant

Description

La fonction « Logiciel superviseur » représente l'interface externe de l'IHM de notre projet. Son objectif est d'assurer la communication entre le superviseur et le robot. Cette fonction revêt une importance cruciale dans le suivit du bon déroulement du travail du robot. Grâce à l'interface le superviseur peut vérifier en temps réel le trajet ainsi que les obstacles repérés par le robot.

Entrées

- **Flux vidéo fourni par le système Surround View (SV) :** Ensemble d'images qui servira à créer la Bird View.
- **Trajectoire planifiée :** La trajectoire générée lors de la planification de trajectoire du robot est affichée sur le moniteur au poste du superviseur.
- **Position du robot :** La position en temps réel du robot est affichée sur le moniteur au poste du superviseur.
- **Position du poste du superviseur :** La position du poste de travail de l'agent est affichée sur le moniteur, aux côtés de la trajectoire et de la position du robot.
- **Position des obstacles :** La position des obstacles détectés est affichée sur le moniteur si ces derniers sont proches de la trajectoire.

Exigences

- **Langage de programmation :** Codage en C++
- **Actualisation des données :** L'application doit fonctionner et évoluer en fonction des informations reçu pour que l'affichage corresponde à ce qui est perçu par le robot.

Sortie

- **Logiciel superviseur :** Version plus complète du logiciel agent. On affiche graphique les différentes données en entrée (position du poste de travail, position temps-réel du robot, trajectoire estimée du robot, position des obstacles, bird view)

Contrainte

- **Attention du superviseur :** Il doit comprendre facilement que le robot est bloqué et pourquoi. Il doit donc être informé le plus vite et efficacement possible

Scénario d'utilisation

Le robot reçoit l'instruction, du superviseur, de se déplacer du point A au point B de l'entrepôt. Il se met en mouvement, et l'écran du superviseur affiche instantanément la trajectoire estimée du robot, sa position en temps réel, ainsi que la position du poste de travail. Il affiche ensuite les obstacles lorsque ces derniers sont détectés ainsi qu'une fonctionnalité avancée d'accès à une vue d'ensemble (bird view) pour chaque obstacle, permettant une évaluation détaillée de la situation. En cas d'obstacle dynamique, le module de traitement déclenche une alerte spécifique, visible sous la forme d'un rond orange sur l'IHM du superviseur. Pour les obstacles statiques inévitables, un message d'alerte distinct (alerte sonore) est envoyé, signalant au superviseur la nécessité d'une intervention humaine, matérialisée par un carré rouge. De plus, la bird view est automatiquement ouverte sur le moniteur.

Fonction 1 : Protocole communication Superviseur/Robot (SV)

Exigence EI-2 - Implémenter une communication avec un opérateur distant Exigence EI-3 - Le chariot envoie une image à l'opérateur, une image d'un obstacle statique imprévu Exigence EI-4 - Un opérateur donne les ordres de démarrage du robot

Description

Un protocole de communication Superviseur/Robot est un ensemble de règles qui définissent la structure, la signification, et les méthodes d'échange des informations entre le superviseur et le robot. Il spécifie la syntaxe, la sémantique, le mode de transmission, la gestion des erreurs, le contrôle de flux, la sécurité et les caractéristiques temporelles de la communication.

Entrée

- **Logiciel superviseur :** Le logiciel doit être fonctionnel afin d'y afficher de nouvelles informations.

Exigence

- **Communication :** Le formatage et la transmission ne doivent pas impacter significativement les temps de traitement. Surtout lorsqu'il s'agit d'une communication du logiciel vers le robot.

Sortie

- **Communication superviseur :** Traduire les informations du robot pour que le logiciel les comprenne et inversement.

Contrainte

- **Interférence :** Il ne faut pas que le poids des transmissions perturbe ou ralentisse le travail du superviseur.

IV - Spécifications de performances

contraintes temps réel :

- taux de rafraîchissement entre perception et action du système de 10 Hz
- la complexité des algorithmes (temps de calcul)
- taux de rafraîchissement de l'interface utilisateur

contraintes physiques du robot :

- charge utile maximale du robot
- vitesse maximale du robot
- (accélération maximale du robot)
- autonomie de la batterie

contraintes sur les trajectoires :

- optimisation de la trajectoire en tenant des comptes des obstacles et de leur nature en temps et/ou en distance (le + rapide possible et le + court possible)