Corrige

Exercice 1. 5 pts

Caractéristique	Roues	Chenilles	Pattes	Drone
Autonomie suffisante pour missions longues (> 2 H)	**Oui**	**Oui**	**Non**	**Non**
Faible consommation d'énergie	**Oui**	**Non**	**Non**	**Non**
Rotation sur place	**Oui**	**Oui**	**Oui**	**Oui**
Complexité mécanique faible	**Oui**	**Non**	**Non**	**Non**
Faible coût de fabrication	**Oui**	**Non**	**Non**	**Non**

Exercice 2. 8 pts

1. **carte d'occupation (occupancy grid)** 1.5pt

2.

0 : cellule libre, aucun obstacle détecté → zone accessible à la navigation.

100 : cellule occupée, obstacle détecté → zone à éviter.

–1 : cellule inconnue, jamais observée par les capteurs → incertitude, non navigable tant qu'elle n'est pas explorée. 0.5x 3 = 1.5 pt

3. **Les zones sont en dehors du champ de vision du LIDAR (ex. : derrière des murs ou objets), ou bien elles n'ont pas encore été explorées du trajet suivi par le robot.** 1pt

4.
Toutes les cellules traversées jusqu'à 2 m sont mises à 0 (libres). 1pt

La cellule impactée à 2 m est mise à 100 (occupée).

5. **Elle est marquée comme libre (0), car le faisceau n'a rencontré aucun obstacle dans sa portée maximale.** 1 pt

6.

- **Les obstacles peuvent être mal localisés.** 1 pt
- **Les distances peuvent être fausses, entraînant des erreurs dans la représentation de l'environnement.**
- **Cela peut aussi fausser la localisation du robot (si elle repose sur la carte).**

7.

L'odométrie (position estimée du robot) et un algorithme de localisation sont indispensables pour que le robot puisse se localiser sur la carte.

Le robot a besoin, en complément des données du LIDAR, d'une estimation initiale de sa position (pose), et idéalement de données d'odométrie pour améliorer la précision de la localisation à l'aide d'algorithmes de SLAM ou de localisation probabiliste. 1 pt

Exercice 3. **7 pts**

1. **$ rosrun robot_script/robot.py** 1pt

2. **/gazebo_gui , /gazebo, /Obs_Det, /rosout** 1pt

3. roscore est le **serveur maître de ROS**. Il coordonne la communication entre les différents nœuds d'un système ROS. 1 pt
4. Gazebo est un **simulateur 3D** de robots et d'environnements, intégré à ROS. 1pt
5. 3pt

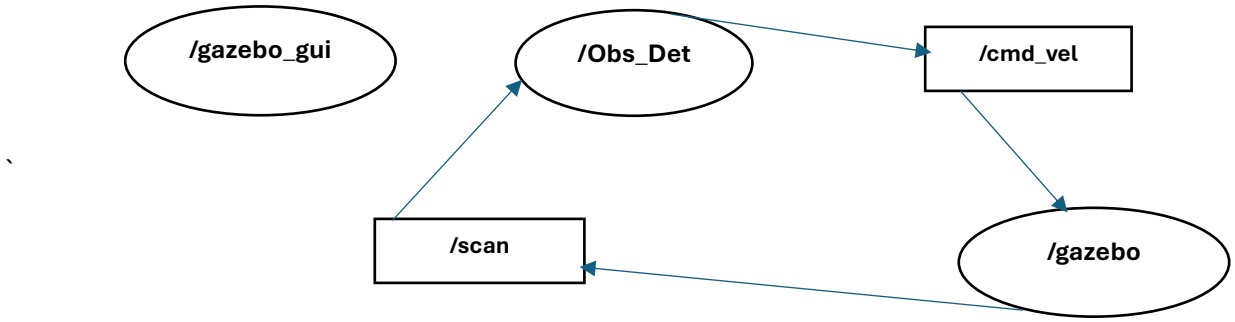
/gazebo_gui : Ce nœud gère l'**interface graphique** de Gazebo (affichage 3D, interactions visuelles).

/gazebo : C'est le **nœud principal du simulateur Gazebo**, qui gère la simulation physique, les plugins et l'interaction avec ROS.

/Obs_Det : Il s'agit d'un nœud utilisateur (script Python) de **détection d'obstacles**, abonné aux capteurs et publiant des commandes.

/scan : Ce topic transporte les **données du LIDAR** vers les nœuds abonnés.

/cmd_vel : Ce topic est utilisé pour envoyer des **commandes de vitesse linéaire et angulaire** au robot.



`