Robots-chiens : Présentation de quelques modèles et applications pédagogiques

La Revue 3EI
Culture Sciences de l'Ingénieur

Édité le 06/10/2025

école normale supérieure paris-saclay

Hervé DISCOURS[1]

[1] *IUT de Cachan, 9 Av. de la Division Leclerc, 94230 Cachan*

Cette ressource fait partie du N°117 de La Revue 3EI du 4ème trimestre 2025.

Cet article passe tout d'abord en revue les principaux robots-chiens actuellement développés, du haut de gamme comme Spot de Boston Dynamics, en passant par les solutions accessibles pour l'éducation et la recherche, ou encore les projets DIY open-source.

Un modèle sera étudié plus en détail, le Go1 de Unitree. Ses différents moyens de contrôle/commande seront analysés, aussi bien à bas niveau (SDK), que haut niveau (MQTT).

Pour terminer, différentes applications pédagogiques seront proposées sous forme de TP pour découvrir les moyens de commande par MQTT, avec Node-RED ou simplement Mosquitto. Quelques idées de projet seront détaillées, soit avec une vision « Automatisme Industriel » en utilisant un automate programmable (Siemens S7), ou soit avec une vision « Système Embarqué » en utilisant le SDK du constructeur (Langage C/C++, Python).

1- Introduction

Les robots quadrupèdes, communément appelés « robots-chiens », constituent aujourd'hui un domaine très dynamique de la recherche en robotique mobile. Inspirés à la fois de la morphologie et de la locomotion animale, ces systèmes cherchent à reproduire la stabilité, la souplesse et l'adaptabilité propres aux mammifères quadrupèdes. L'appellation « robot-chien » tient à leur ressemblance visuelle et fonctionnelle avec les canidés, mais elle recouvre en réalité une variété de plateformes robotiques conçues pour des usages divers, allant de l'exploration industrielle à l'assistance humaine.

C'est au cours de la dernière décennie que des avancées majeures ont été réalisées grâce à la miniaturisation des constituants et à l'amélioration des systèmes de contrôle dynamique. Des entreprises pionnières comme Boston Dynamics, ou encore la société chinoise Unitree Robotics, ont contribué à démocratiser ces plateformes, autrefois confinées aux laboratoires de recherche.

Au-delà de la prouesse technologique, l'intérêt pour les robots-chiens réside dans leur capacité à évoluer dans des environnements complexes où les robots à roues ou à chenilles peinent à se déplacer. Qu'il s'agisse de franchir des obstacles, de maintenir l'équilibre sur des terrains accidentés ou d'interagir avec des opérateurs humains, ces robots se distinguent par leur polyvalence et leur robustesse. Ils sont aujourd'hui étudiés et déployés dans des contextes variés : sécurité, exploration de zones sinistrées, logistique industrielle ou encore interaction sociale.

2 - Quelques modèles du marché

2.1 - SPOT de Boston Dynamics

Pionnier dans le domaine, Boston Dynamics est une entreprise américaine active depuis1992. De nombreuses réalisations ont rendu célèbre la marque, au travers de vidéos futuristes publiées sur les réseaux sociaux [Vidéo 1]. SPOT est la série quadrupède des robots développés par la marque (Figure 1).

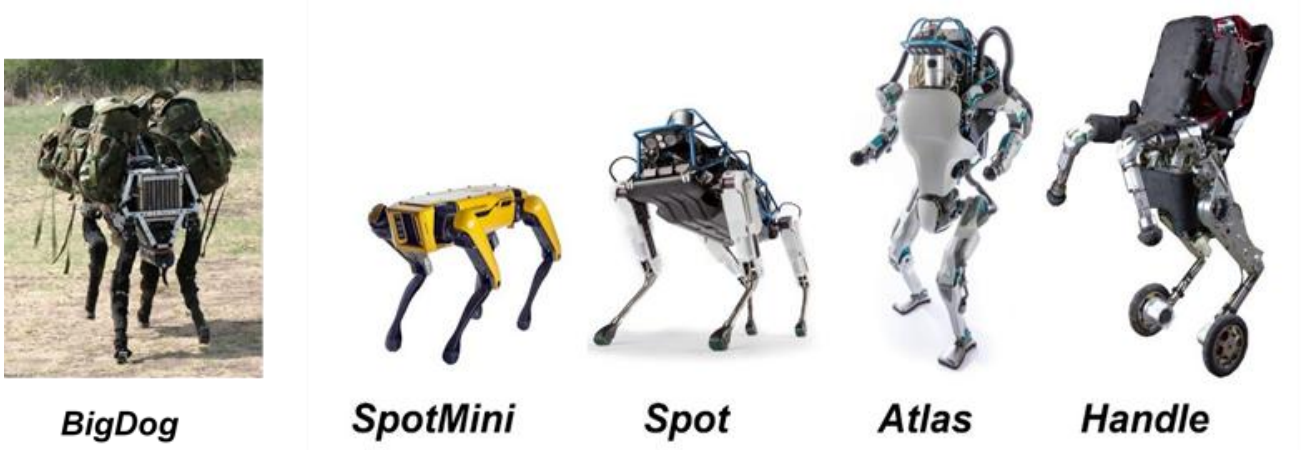

Figure 1 : Quelques robots célèbres de Boston Dynamics

SpotMini, initialement dévoilé en 2016, il a été commercialisé à partir de 2019. Malgré son prix élevé, à partir de 75 000 $, ses caractéristiques de mobilité, de charge utile, de stabilité et de capacités d'interaction avec l'environnement, ont fait de lui un succès commercial. Avec son « pelage », jaune et noir, SpotMini mesure plus d'un mètre de long, il peut porter jusqu'à 14 kg sur son dos, avec une vitesse de 5,7 km/h et une autonomie de 1H30 (Figure 2).

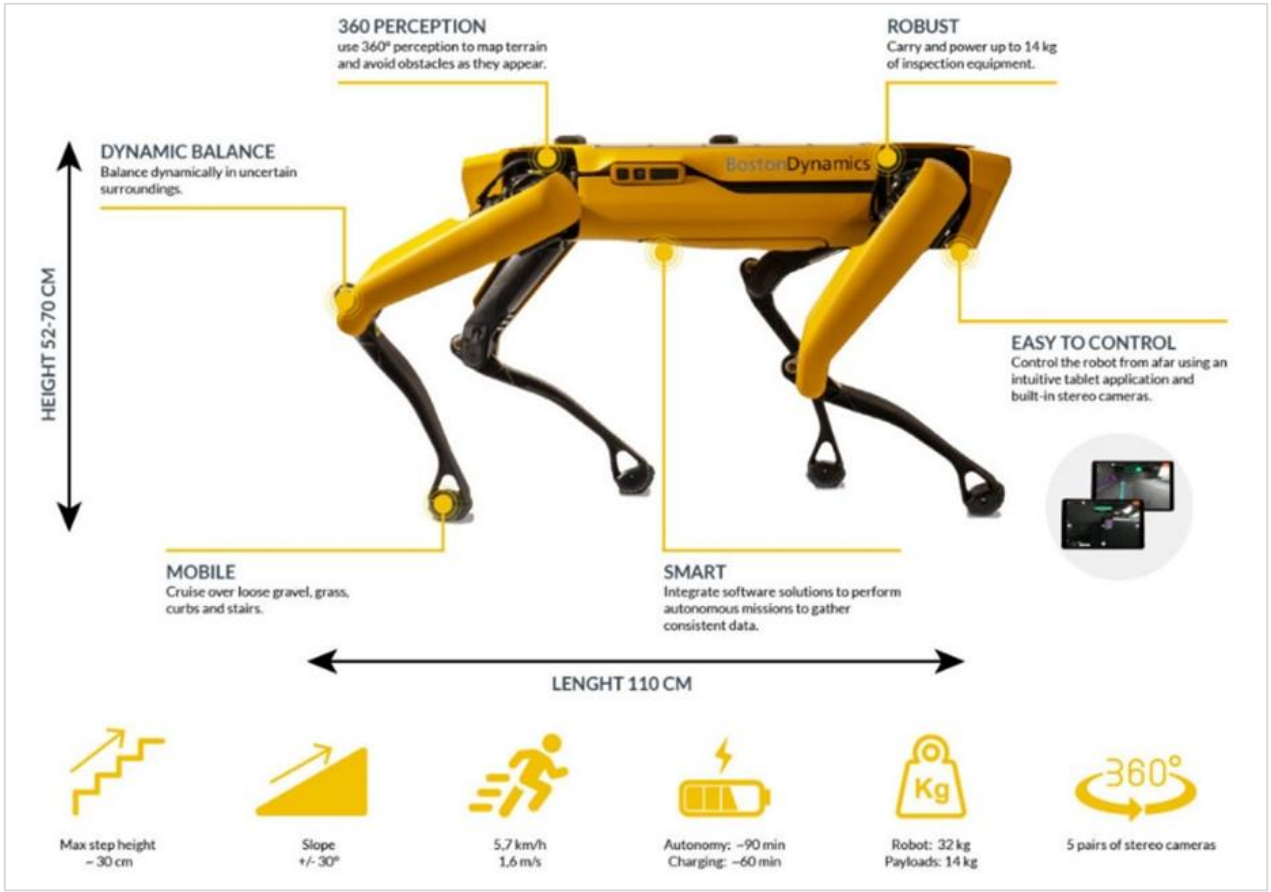

Figure 2 : Présentation de SpotMini

Au niveau contrôle/commande, comme beaucoup de ses concurrents, SpotMini peut fonctionner de trois façons.

Pilotage manuel

Télécommande / tablette (Figure 3) : Le robot peut être contrôlé à distance via une télécommande ou une interface sur tablette.

Contrôle précis : On peut lui demander de marcher, tourner, monter des escaliers ou manipuler des objets avec son bras robotisé (accessoire à acheter en plus de la version de base)

Situations adaptées : Idéal lorsque l'opérateur veut avoir un contrôle complet dans des environnements complexes ou inhabituels.

Navigation autonome

Planification de trajectoire (Figure 3) : Grâce à ses capteurs (caméras stéréo, LIDAR, IMU), SpotMini peut cartographier son environnement et se déplacer de manière autonome.

Évitement d'obstacles : Il peut contourner les obstacles, monter et descendre des escaliers ou traverser des terrains irréguliers sans intervention humaine.

Mission préprogrammée : On peut définir un parcours ou des tâches spécifiques, et le robot s'exécute tout seul.

Fonctionnement coopératif

Dans certaines conditions, SpotMini peut fonctionner de façon coopérative avec d'autres robots, travaillant conjointement pour accomplir une tâche comme ouvrir une porte : l'un scanne ou analyse la scène, puis l'autre intervient avec un bras [Vidéo 2].

Néanmoins, Spot peut coopérer avec d'autres robots dans des scénarios planifiés ou de démonstration, ou via des plateformes logicielles qui permettent la coordination de flotte. Mais ce n'est pas encore un "mode coopératif universel" qui fonctionne dans tous les environnements sans configuration préalable, contrairement à des drones aériens qui réalisent des spectacles (type shows de lumières avec des centaines/milliers de drones) et sont conçus dès le départ pour coopérer de manière extrêmement synchronisée et autonome.

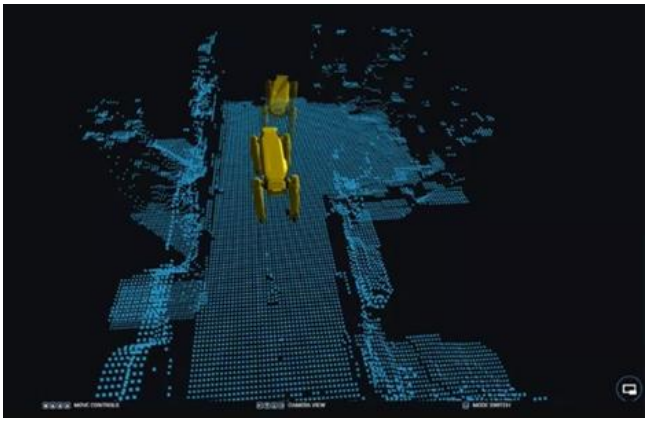

Figure 3 : Pilotage manuel ou Navigation autonome de SpotMini

Comme tous les robots chiens actuels, SpotMini peut être utilisé pour des missions dans des environnements dangereux (Figure 4). A titre d'exemple, il a été utilisé dans le cadre de la pandémie de Covid-19, ou lors d'inspections sur des plateformes pétrolières en mer du Nord, ou encore pour cartographier la distribution de la radioactivité à l'intérieur du New Safe Confinement (NSC), la structure en béton recouvrant le réacteur 4, site de la catastrophe nucléaire de 1986 à Tchernobyl.

Figure 4 : Un robot-chien n'a peur de rien

Différentes versions de Spot ont aussi été testés dans des contextes militaires ou de surveillance. Aux États-Unis, mais aussi en France (depuis 2021 à Saint-Cyr Coëtquidan), plusieurs armées ont expérimenté Spot dans des exercices d'observation, transport de charges, détection de mines et coordination avec drones aériens et systèmes de commandement [Vidéo 3]. On peut également citer l'utilisation en patrouille de surveillance de la résidence de Donald Trump à Mar-a-Lago. (Figure 5).

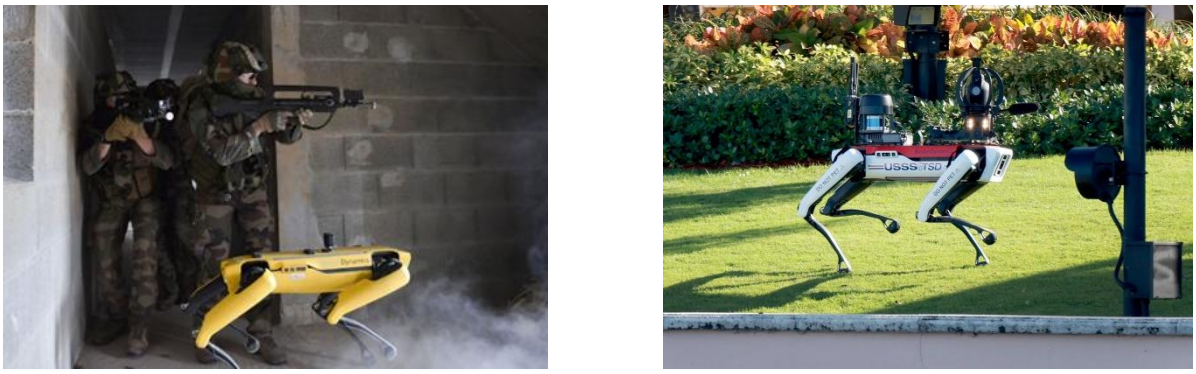

Figure 5 : Missions de défense et surveillance

Permettant de réduire l'exposition des soldats dans des zones hostiles ou inconnues, un robot-chien peut franchir des terrains accidentés, ce qui peut compléter le rôle des véhicules terrestres ou drones. Pour des considérations éthiques et réglementaires, Boston Dynamics a publiquement refusé d'armer Spot, ou de lui faire utiliser des armes létales. Ce n'est pas le cas de nombreux concurrents, comme le SPUR de Ghost Robotics, qui est équipé d'un fusil semi-automatique, de précision et de gros calibre (Figure 6). Et sans parler de toutes les transformations faites « maison » (Figure 7), ou mêmes des adaptations vendues dans le commerce, comme le « Thermonator » développé par la société THROWFLAME et équipé d'un lance-flammes longue distance [Vidéo 5].

Figure 6 : SPUR (Special Purpose Unmanned Rifle)

Figure 7 : UniTree Go1 with a Gun

2.2 - Unitree Robotics

Après le succès de l'américain SPOT, d'autres acteurs se sont imposés sur le marché, comme la société chinoise Unitree Robotics, qui a proposé des modèles plus accessibles financièrement, à partir de 3000 $, tels que le Go1 ou le Go2, destinés autant à la recherche académique qu'à des usages commerciaux. Depuis 2018, la gamme des produits Unitree se distingue par sa diversité, allant des modèles grand public, aux plateformes professionnelles destinées à des applications industrielles avancées (Figure 8).

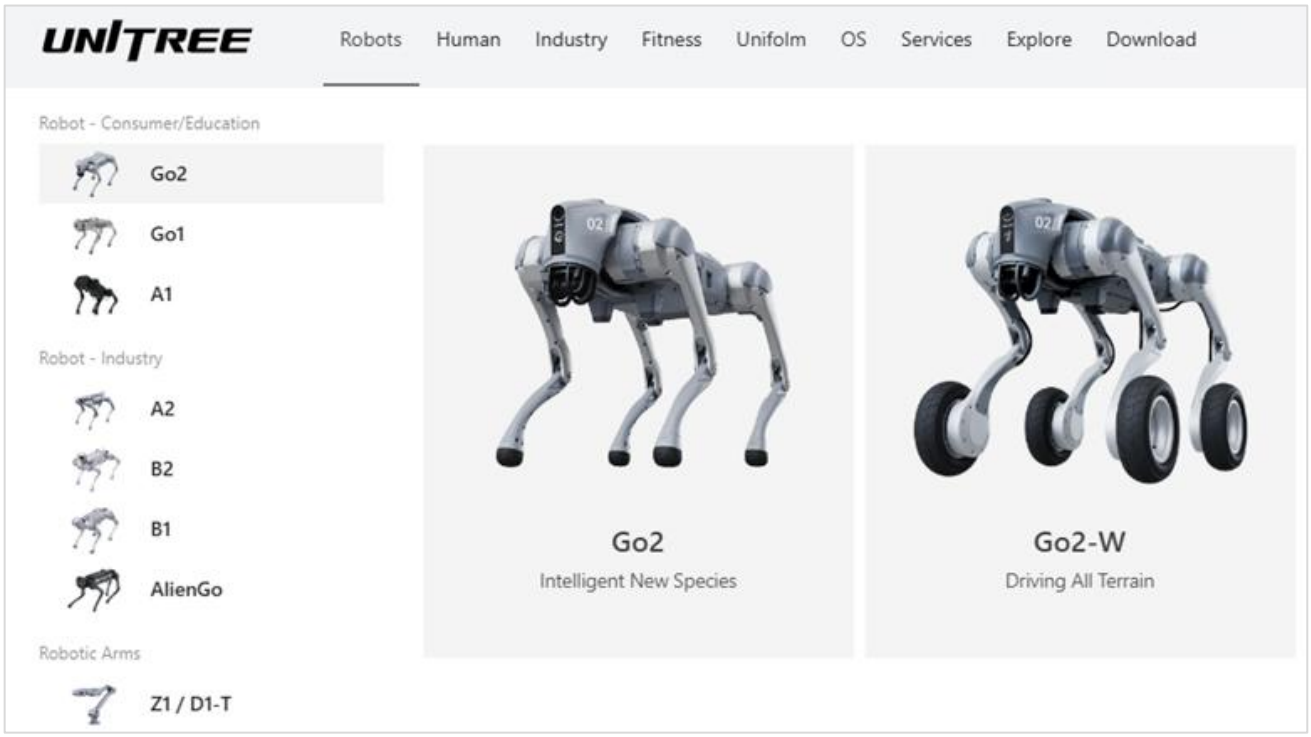

Figure 8 : Robots chiens Unitree en 2025

Les modèles Go1 et Go2 sont bien adaptés pour les établissements de formation, offrant des plateformes accessibles pour l'apprentissage de la robotique et le développement de projets innovants. Ils sont chacun proposés en plusieurs versions (AIR, PRO, X, EDU…) avec des équipements basiques, jusqu'à très complet, pour des prix allant de 1600 $ à plus de 15000 $.

	Go1 [AIR]	Go1 [PRO]	A1	Go2 [AIR]	Go2 [PRO]
Longueur (cm)	59	59	62	70	70
Vitesse (km/h)	9	12,6	11,8	9	12,6
Poids (kg)	12	12	12	15	15
Charge utile (kg)	3	3	5	7	8
Autonomie	~1 H	~1 H	1~2,5 H	1~2 H	1~2 H
Prix HT ($)	2700	3500	10 000	1600	2800

2.3 - DEEP Robotics

Plus récemment apparue sur le marché (2018~2020), l'entreprise chinoise DEEP Robotics propose plusieurs robots-chiens ayant de très bons rapports qualité/prix : X20, X30 et Lite3.

Avec des performances très proche du Go2 de Unitree, le Lite3 est proposé en plusieurs versions, avec là aussi différents niveaux d'accessoires (Figure 9). Comme son concurrent, le code source est « ouvert » et disponible : {https://github.com/DeepRoboticsLab}.

Figure 9 : DEEP Robotics Lite3 Pro

Le Lite3 est fourni avec une intégration ROS prête à l'emploi, ce qui permet :

- De piloter le robot directement via les nœuds ROS.
- De récupérer en temps réel les données des capteurs (caméras, IMU, LiDAR, ultrason, etc.).
- De développer des algorithmes de navigation, de planification de trajectoire ou de perception en utilisant des bibliothèques ROS existantes (SLAM, MoveIt, Nav2, etc.).

Le robot est livré avec un SDK (Software Development Kit) qui expose ses principales fonctions via une API, ce qui permet d'accéder directement :

- Aux commandes de locomotion (vitesse, orientation, allures prédéfinies comme : trot, pas, saut, etc.).
- Aux capteurs embarqués (caméras, IMU, LiDAR selon la version).
- Aux diagnostics (batterie, température, état des articulations).

Le SDK est compatible avec plusieurs langages, typiquement Python et C++, ce qui permet aussi bien un prototypage rapide qu'un développement robuste. On peut l'intégrer avec des frameworks d'IA (TensorFlow, PyTorch) pour entraîner des comportements basés sur l'apprentissage machine, comme la reconnaissance d'objets ou le suivi de personnes.

DEEP Robotics fournit un modèle URDF complet du Lite3. [Unified Robot Description Format] est un format de description des robots utilisé dans ROS et dans des simulateurs comme Gazebo, Webots ou Isaac Sim. Ce modèle permet de :

- Simuler le robot dans un environnement virtuel (avant même de l'avoir physiquement).
- Tester des algorithmes de locomotion, de navigation, de SLAM ou d'évitement d'obstacles en conditions simulées.
- Former des modèles d'apprentissage par renforcement (reinforcement learning) dans un environnement simulé, puis transférer ces comportements dans le robot réel (méthode appelée sim-to-real).

2.4 - Petoi avec « Bittle » et « Nybble »

Fondée en 2017, et basée aux États-Unis et en Chine, Petoi est une petite start-up dont l'objectif est de rendre accessibles des robots quadrupèdes à un large public, pour un usage éducatif. Les

produits principaux sont Nybble (un mini robot-chat en structure bois) et Bittle (un mini robot-chien), tous deux pesant moins de 300 grammes (Figure 10).

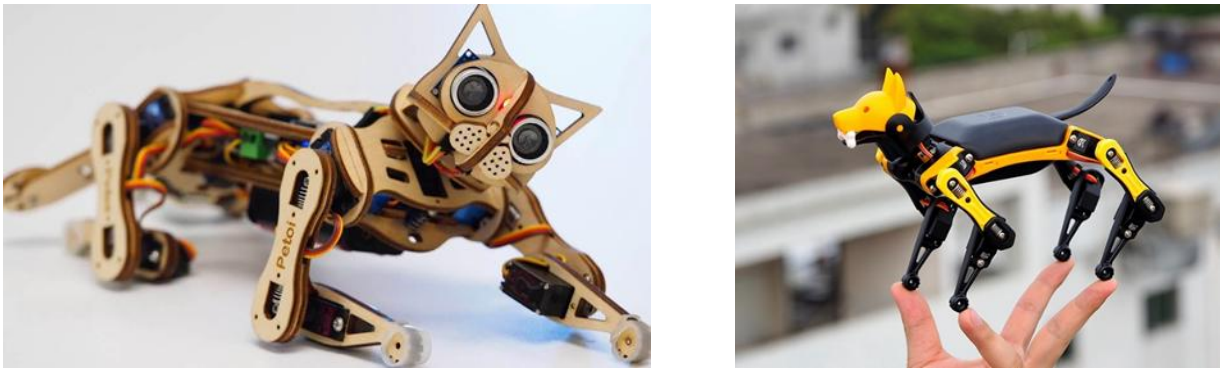

Figure 10 : Nybble et Bittle de PETOI

Pour le contrôle/commande, les deux mini robots, utilisent des cartes de développement « grand public », basés sur ESP32 ou ATmega328 (Arduino) (Figure 11). Le fait que matériel et logiciel soient ouverts, rend accessible leur utilisation pour des lycéens, étudiants, amateurs, clubs de robotique. Ils ne cherchent pas à rivaliser sur les performances extrêmes, mais sur l'accessibilité, la modularité, et la communauté. Une version puzzle 3D est également disponible sur les deux modèles (**Erreur ! Source du renvoi introuvable.**).

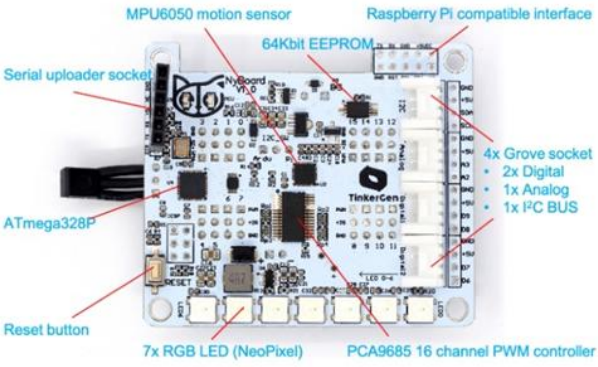

Figure 11 : Architecture avec ATmega

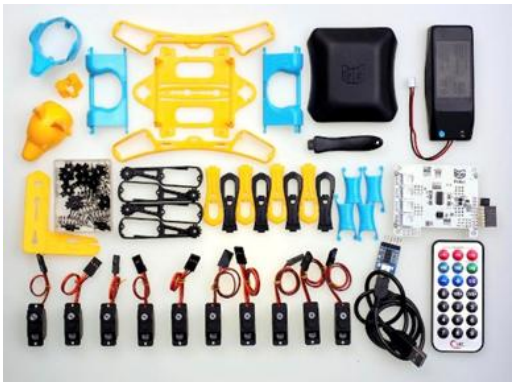

Figure 12 : Kit complet de Bittle

2.5 - Les modèles DIY

Avec la démocratisation des moyens de prototypage (logiciels de modélisation 3D, imprimantes 3D, cartes de développement open source…), de nombreux amateurs se lancent dans la fabrication de robots quadrupèdes « faits maison ».

Voici quelques projets qui ont rencontré un franc succès sur YouTube ou ailleurs. Ce qui les rend notables ou populaires est lié à :

Originalité mécanique ou conceptuelle

- Par exemple, éviter les engrenages / poulies classiques, articulations ingénieuses….
- Ou des versions modifiées de robots célèbres (Spot, etc.) conçues par des amateurs.

Travail visuel / démonstration

- Voir le robot marcher, danser, faire quelque chose de reconnaissable. Pas juste de la théorie ou du code.
- Explications détaillées et pédagogiques.
- Les vidéos "full build" + résultats finaux (marche, allures, équilibre…).

Accessibilité & documentation

- Fichiers CAD partagés, liste de pièces, tutoriels étape par étape, ce qui permet à d'autres de le reproduire.
- Codes réalisés sur des kits grand public (Arduino, ESP32, Raspberry…) et disponibles en open source sur Github.

Le « TOPS » de Aaed Musa

Conçu par Aaed Musa, un créateur passionné de robotique, TOPS est un robot quadrupède à 12 degrés de liberté (DOF), entièrement imprimé en 3D et open-source (Figure 13). Son nom est un jeu de mots sur le célèbre robot Spot de Boston Dynamics, en inversant les lettres de "SPOT". Ce robot est contrôlé via un système basé une carte Teensy 4.1, avec des bibliothèques spécifiques fournies sur GitHub, compatibles avec le framework Arduino. Les 12 actionneurs QDD (Quasi-Direct Drive) sont « faits maison » avec moteur Brusless et réducteur planétaire imprimé en 3D [Vidéo 6].

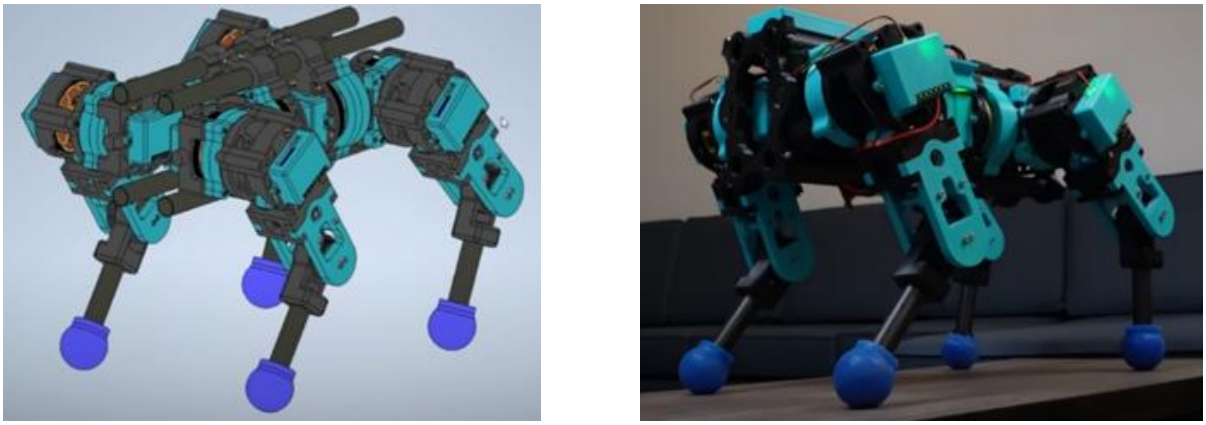

Figure 13 : Le « TOPS » de Aaed Musa

Le « openDog » de James Bruton

Le robot quadrupède développé par James Bruton (Figure 14) en est à sa version « openDog V3 ». La structure est composée de pièces imprimées en 3D, renforcée avec des pieds en TPU. Les moteurs sont de type Brusless, aveccontroleurs ODrive, le tout commandé par une carte Teensy 4.1 (Processeur ARM Cortex-M7 à 600 MHz).

Le projet est conçu pour être compatible avec le système ROS (Robot Operating System), facilitant l'intégration de fonctionnalités avancées telles que la reconnaissance gestuelle, la navigation autonome et l'interaction avec l'environnement [Vidéo 7].

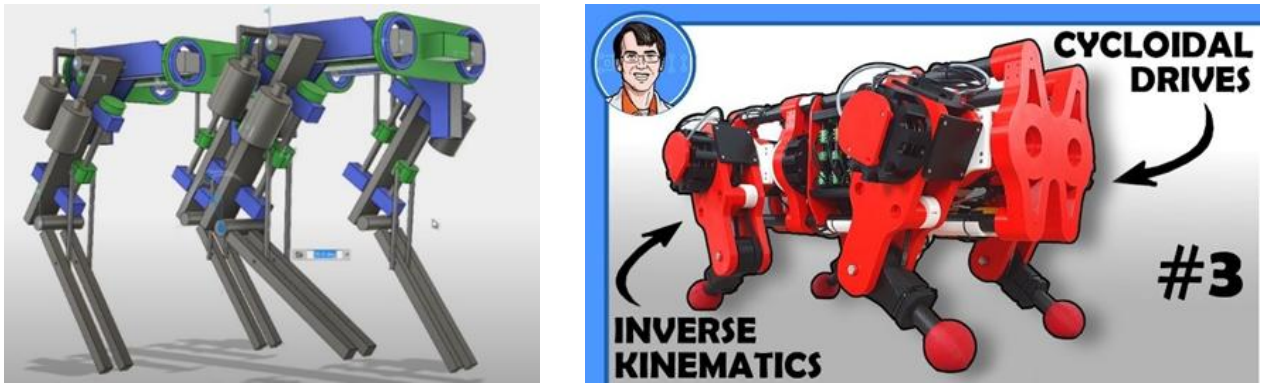

Figure 14 : Le « openDog » de James Bruton

Le « spotMicro » de [Advanced Hoppy Lab] et [SpotMicroAI]

SpotMicro est une version revisitée du robot quadrupède open-source initialement conçu par Deok-yeon Kim, qui se veut être visuellement, une copie miniature du robot de Boston Dynamics (Figure 15) [Vidéo 8]. La partie commande est réalisée avec un Raspberry Pi, et une programmation en Python avec intégration de ROS pour des fonctionnalités avancées.

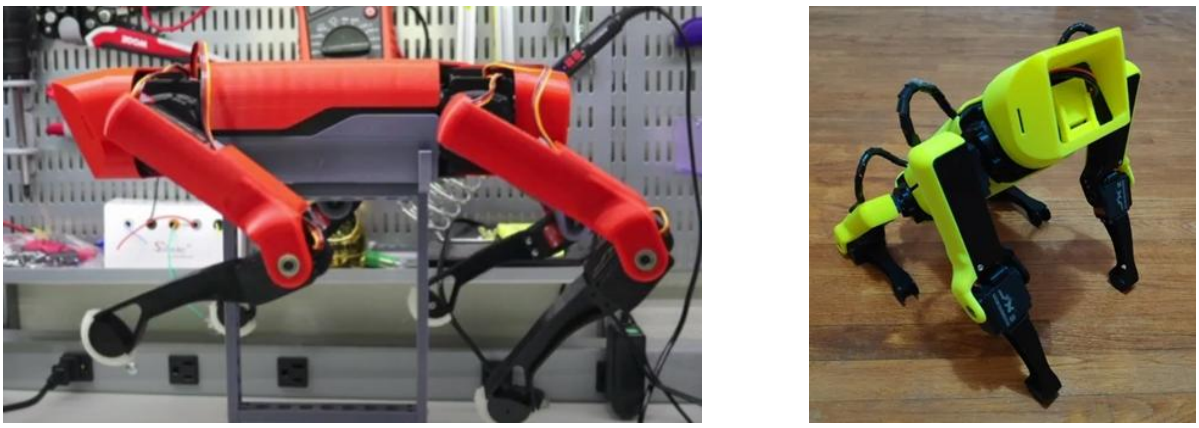

Figure 15 : Le « spotMicro » de [Advanced Hoppy Lab]

Le « robot dog » de [Maker 101]

Le robot-chien de Maker 101 est conçu pour être un projet d'introduction à la robotique, combinant mécanique, électronique et programmation simple. Il est basé sur une structure imprimée en 3D, contrôlée par un microcontrôleur Arduino Nano et équipée de servomoteurs standards pour les mouvements (Figure 16).

Un tutoriel détaillé est disponible sur YouTube [Vidéo 9], expliquant étape par étape la construction et le contrôle du robot-chien. Ce projet est destiné aux débutants qui souhaitent se lancer dans la robotique sans investissements majeurs, tout en apprenant les bases de la mécanique, de l'électronique et de la programmation.

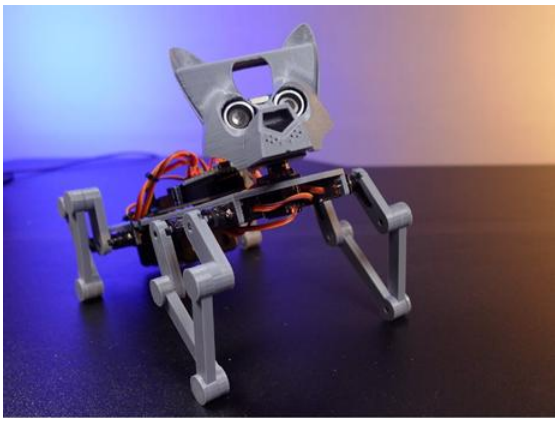

Figure 16 : Le robot dog de [Maker 101]

Le « mjbots quad » de Josh Pieper

C'est l'un des projets DIY / open source les plus aboutis à ce jour, mêlant des parties mécaniques, électroniques avancées et logiciel de contrôle dynamique. L'objectif du projet était de construire un robot quadrupède dynamique, performant, inspiré de robots comme le Mini-Cheetah, mais entièrement open source [Vidéo 10].

Le Mini Cheetah est un robot quadrupède développé par le MIT Biomimetic Robotics Lab. (Figure 18). C'est un robot de recherche qui a marqué les esprits car il était à la fois petit, agile, robuste

et relativement “abordable” comparé aux robots quadrupèdes commerciaux comme ceux de Boston Dynamics. Présenté en 2019, il apparait dans des vidéos où il réalise des backflips, court dans l’herbe, ou résiste aux coups de pied [Vidéo 11]. Il a servi de base pour tester de nouveaux algorithmes de Machine Learning pour la locomotion quadrupède.

La version A1 du « mjbots quad » (Figure 17) utilise des servos brushless avec des contrôleurs personnalisés. La structure est composée de pièces métalliques usinées spécialement pour le projet. Le robot utilise un Raspberry Pi 3B+ comme unité centrale pour le contrôle global. Le logiciel de contrôle dynamique est très performant, alliant contrôle de couple et position, avec gestion des gaits (allures), du séquencement et de la dynamique de marche.

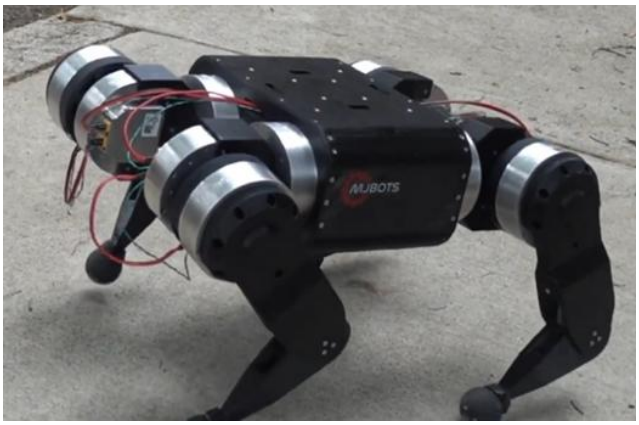

Figure 17 : Le « mjbots quad » A1

Figure 18 : Backflip de Mini Cheetah

Le projet « Antartic » de [Gonzalo Espinoza]

Gonzalo Espinoza Graham est à l’origine du projet « Antartic », nom donné car son objectif initial était de construire un robot, de l'envoyer en Antarctique et de diffuser sa survie en direct. Gonzalo adopte une approche "open-source" totale, partageant chaque aspect du projet, y compris les erreurs et les leçons apprises. Les vidéos durent plusieurs heures chacune et montrent chaque étape de la CAO, de l'impression 3D, de l'assemblage, de la programmation du firmware, des modèles d'entraînement... (Figure 19) [Vidéo 12]. Le projet a débuté le 8 septembre 2024 et est toujours en cours.

Le robot est programmé par apprentissage automatique (ML) et inspiré des travaux de Disney sur les personnages robotiques bipèdes {Disney’s Bipedal Robotic Character paper}. Le prototypage initial a commencé avec un Arduino Uno classique, pour tester les servomoteurs, la cinématique des pattes, et la lecture des capteurs (IMU, etc.).

Le robot (géométrie, masses, moteurs, contraintes) est tout d’abord modélisé dans un moteur de simulation (PyBullet ou MuJoCo, les deux très populaires pour l’entraînement de robots quadrupèdes). Le robot est donc simulé via son modèle URDF, ce qui permet de tester rapidement des milliers de combinaisons de mouvements.

Le modèle (réseau de neurones) assurant la commande de mobilité est entraîné en Python sur PC, avec algorithmes intégrés dans des frameworks Python comme Stable Baselines3 ou RLlib.

Framework logiciel :

- Côté ML pur : PyTorch ou TensorFlow.
- Côté interface avec la simulation : Gymnasium (anciennement OpenAI Gym) pour définir les environnements.

L'apprentissage est réalisé par renforcement (Reinforcement Learning, RL). Le robot apprend en maximisant une récompense : avancer sans tomber, garder l'équilibre, économiser de l'énergie. Les algorithmes typiques dans ce cadre, sont :

- PPO (Proximal Policy Optimization),
- DDPG (Deep Deterministic Policy Gradient),
- SAC (Soft Actor-Critic).

Une fois entraîné, le réseau est compressé (quantization, pruning) et exporté dans un format léger : TensorFlow Lite ou onnx. Sur le robot, le microcontrôleur (Arduino) ne fait pas l'entraînement, il fait seulement l'inférence : lire les capteurs (IMU, positions servos), puis appliquer le modèle ML, afin de générer les consignes moteur.

Figure 19 : Tutorial complet de Gonzalo Espinoza

Et tous les autres qui ont moins de visibilité

Il convient de préciser qu'il demeure impossible d'être exhaustif dans le recensement des différents modèles de robots quadrupèdes actuellement disponibles ou en développement. Le domaine est en effet marqué par une très forte dynamique d'innovation, où de grands industriels, des start-ups, des laboratoires de recherche et même des communautés de makers indépendants proposent régulièrement de nouvelles plateformes, souvent sous forme de prototypes, de concepts ou de séries limitées. De plus, de nombreux projets restent confidentiels, non publiés ou abandonnés avant commercialisation, tandis que d'autres ne sont documentés qu'à travers des canaux informels (vidéos en ligne, dépôts GitHub, forums spécialisés). Dans ces conditions, tout état de l'art à un instant « t » ne peut offrir qu'une vision partielle et nécessairement sélective de l'écosystème, en privilégiant les contributions les plus représentatives ou les mieux documentées.

2.6 - Kawasaki CORLEO

Au départ, on aurait pu penser à une « fake news », mais Kawasaki a bien présenté sur son site officiel, un « véhicule de mobilité personnelle tout-terrain à quatre pattes », nommé CORLEO (Core Logic Enhanced Organism). Il a été présenté à la presse à l'exposition 2025 de Osaka-Kansai. Il est encore à l'état de concept, montré sous forme d'une maquette ou de simulations pour l'exposition d'Osaka, avec vidéos d'animation [Vidéo 13]. Certains articles parlent d'un horizon très lointain (2050) comme éventuelle date de mise sur le marché.

Kawasaki propose donc un concept révolutionnaire de moto trail qui troque les roues pour 4 pattes robotisées, inspirées des mouvements félins. Pensé pour explorer les terrains les plus hostiles, ce véhicule futuriste fonctionne à l'hydrogène et promet une expérience de pilotage ultra-immersive (Figure 20).

Figure 20 : Le Kawasaki CORLEO

Ce robot, que certains présentent comme un cheval mécanique, est propulsé par un moteur à hydrogène de 150 cc pour générer l'électricité nécessaire aux unités motrices des quatre pattes. On lit des estimations non confirmées de 8 à 12 heures d'usage, avec une puissance de 20 CV pour l'ensemble de la propulsion, à savoir quatre jambes motorisées, chacune indépendante, avec sabots en caoutchouc antidérapants, "divisés" gauche/droite pour une meilleure adhérence (Figure 21).

Au niveau du contrôle, notons la présence de systèmes d'IA pour le contrôle de l'équilibre, adaptation de la démarche (gait adjustment), la reconnaissance d'obstacles, l'adaptation de l'itinéraire.

Il y a également la détection du déplacement du centre de gravité du pilote, via des capteurs, pour ajuster la posture du pilote et pour rester ergonomique même en montée ou sur terrain accidenté. Ajoutons un affichage tête haute (HUD) présentant toutes les données comme niveau d'hydrogène, stabilité, informations de navigation et la projection au sol de repères lumineux la nuit, pour guider le pilote.

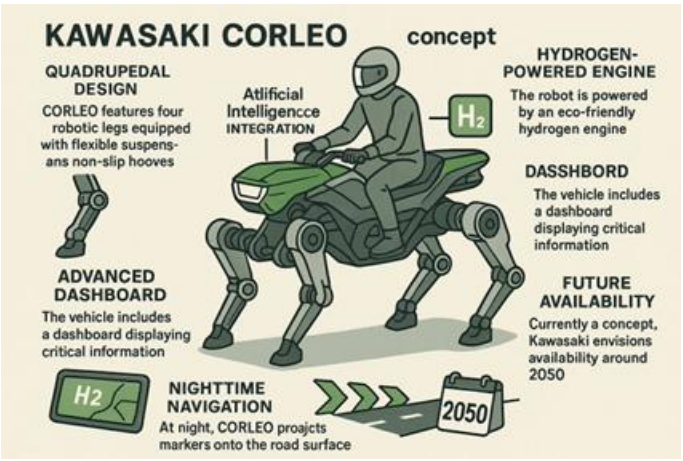

Figure 21 : Constitution du CORLEO

3 - Robot-chien Go1 et applications pédagogiques

3.1 - Présentation du robot

Premier grand succès commercial pour Unitree, Go1 est commercialisé depuis 2021, avec plus de 20000 unités vendues en 2023. Depuis 2025 la gamme s'est étoffée avec la venue de Go2, qui en fait le successeur potentiel de Go1. Mais le Go1 reste pertinent pour un usage plus simple, pour des applications industrielles moins intensives. De plus les deux robots sont chacun disponibles en plusieurs versions pour couvrir de nombreux cas d'usage.

Pour la suite de cet article, c'est le modèle Go1 EDU qui sera étudié. Mesurant 70 cm, il pèse environ 12 kg avec batterie, et sa charge utile est de 5 kg. La batterie Li-ion de 6000 mAh assure une autonomie comprise entre 1 et 2 heures suivant le mode d'utilisation [Vidéo 14].

La partie mécanique comporte 12 servomoteurs. Donc 3 par patte (Figure 22). Il s'agit de moteurs synchrones, à aimants, avec réducteur mécanique 1/6.33, et contrôleur électronique intégré. Les consignes (Angle, couple, vitesse) sont transmises via une liaison, série, RS485, à 4 Mégabits, par seconde (Figure 23).

Figure 22 : Articulations de Go1

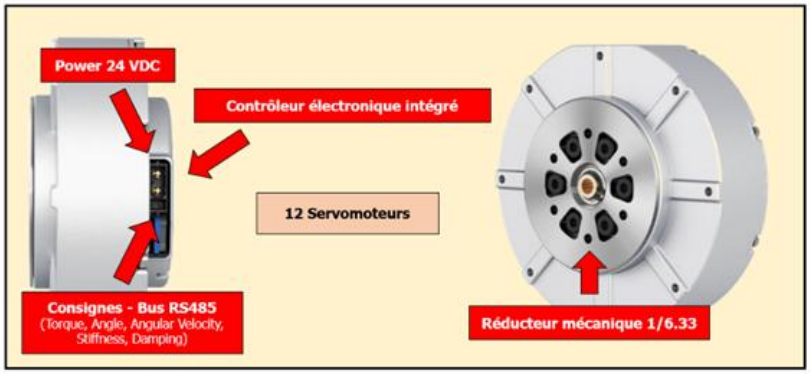

Figure 23 : Servomoteurs de Go1

Les servomoteurs sont commandés à partir d'une carte de contrôle, avec microcontrôleur STM-32, intégrant une centrale inertielle pour la mesure des vitesses, des accélérations et de l'orientation sur les 3 axes. Une liaison radio permet d'effectuer, avec une télécommande spéciale, un arrêt d'urgence.

Toujours dans Go1, trois cartes Jetson Nvidia, pour le traitement des images issues des 5 caméras. Et tout ce beau monde, communique en Ethernet, via un switch. Ce réseau est accessible à partir d'un port lié au connecteur RJ45, sur le dos du chien.

Mais l'élément essentiel est la Raspberry, qui centralise tout. Elle communique aussi avec un module WiFi qui permet une liaison sans fil depuis un ordinateur ou un smartphone équipé de l'application proposée par le fabricant (Figure 24).

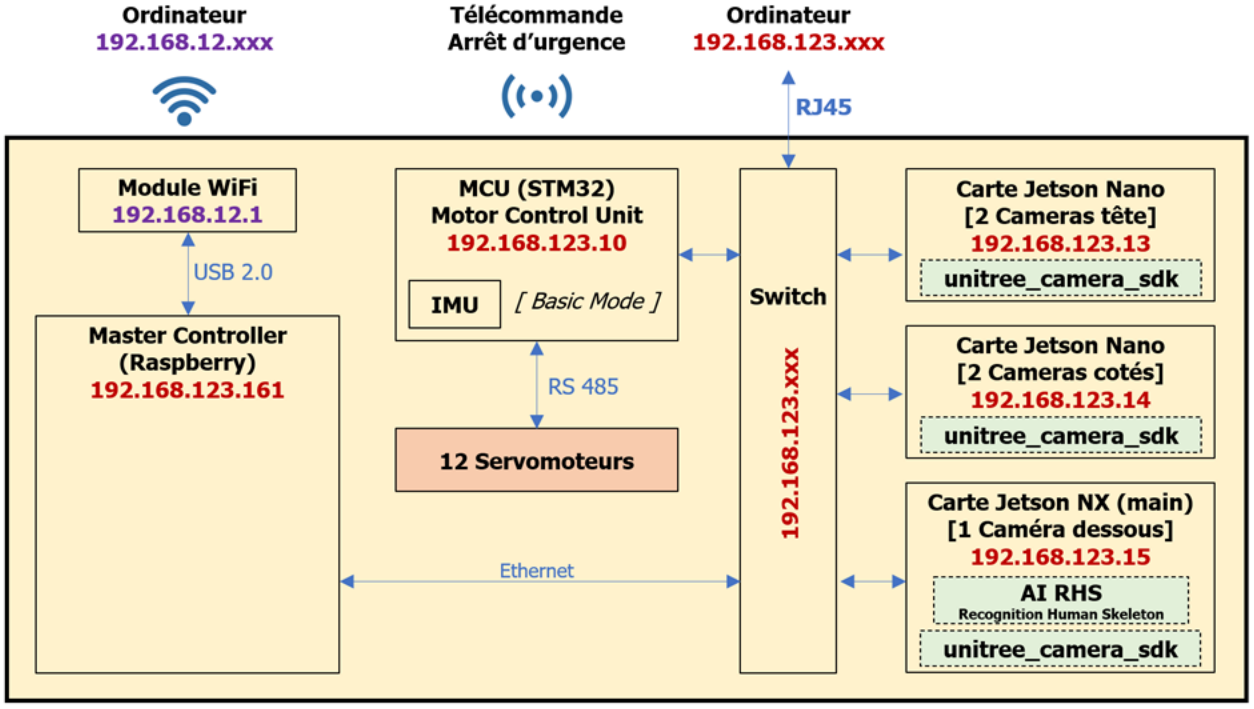

Figure 24 : Architecture interne de Go1 EDU

Serveur Web de Go1

Au niveau logiciel, dans la Raspberry, on trouve un serveur web embarqué qui permet de commander le robot via un simple navigateur web. Il faut juste entrer l'adresse de la Raspberry :

- 192.168.123.161. en filaire,
- 192.168.12.1. en WiFi (Figure 25).

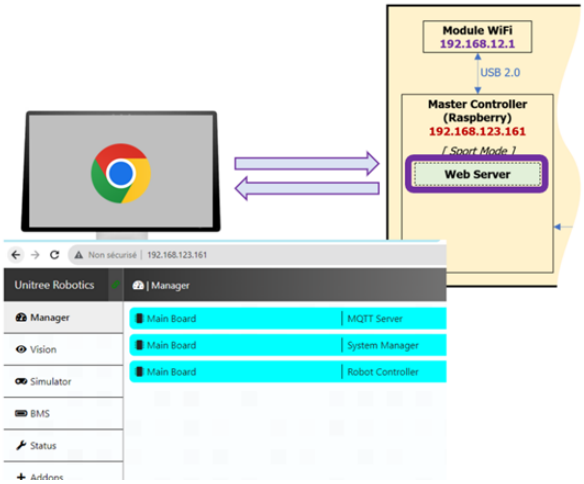

Figure 25 : Commande de Go1 par son serveur Web

Le SDK de Unitree

Autre avantage avec le robot Unitree, un SDK (Software Development Kit) est fourni par le constructeur. Le programme est compilé et exécuté en local dans le Raspberry. L'ordinateur, quant à lui, nous permet de choisir quel programme ou fonction il faut compiler et exécuter. Les ordres envoyés sont des commandes sous Linux, car c'est le système d'exploitation du Raspberry. Les programmes sont ceux fournis dans le SDK, donc la mémoire du Raspberry. Il y a de nombreux exemples d'applications, mais on peut bien sûr chercher sur Internet/GitHub d'autres fichiers sources pour les exécuter. Ils sont généralement disponibles en C++, JavaSript ou Python (Figure 26).

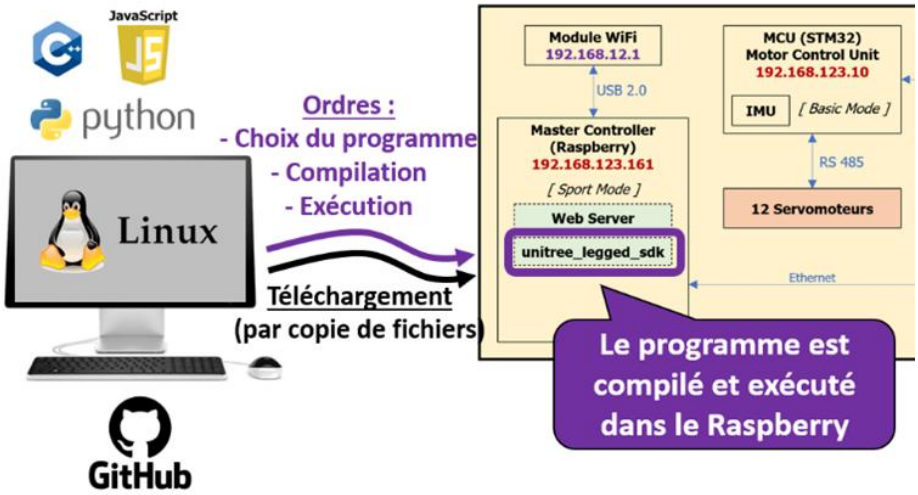

Figure 26 : Utilisation du sdk Unitree

Go1 est compatible ROS

Le Go1 est conçu pour être contrôlé via ROS (Robot Operating System) ce qui permet d'écrire des programmes de navigation, de suivi, ou d'actions complexes.

Pour faire simple :

- ROS tourne dans le robot Go1 sur son ordinateur embarqué (Raspberry).
- ROS crée et gère tous les topics et nœuds du système :
    - Communication entre les différents composants du robot (capteurs, moteurs, caméras).
    - Les topics : Flux d'informations (ex. /imu pour les données d'accéléromètre, /cmd_vel pour la vitesse).
    - Les services et actions : Commandes ponctuelles ou actions longues (ex. démarrer une séquence de mouvement).

Python ou C++ sert ensuite à écrire des programmes ou scripts qui pilotent le robot via ROS (Figure 27) (Figure 28). Il devient possible :

- De souscrire aux topics ROS pour lire les données des capteurs.
- De publier des commandes sur les topics ROS pour déplacer le robot.
- D'implémenter de la logique : suivi d'objet, navigation autonome, séquences de mouvements, analyse caméra…)

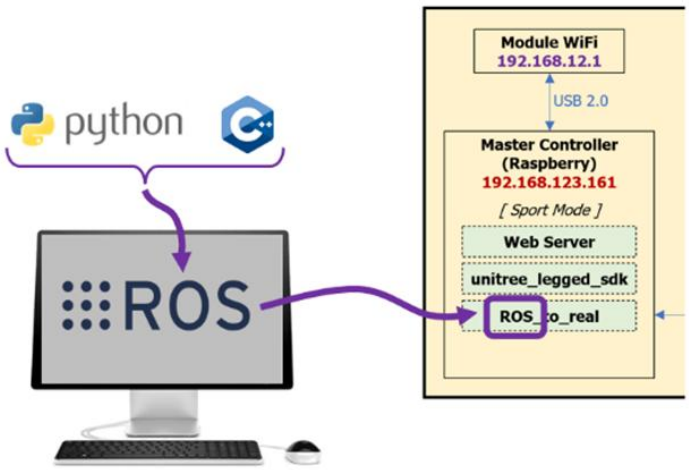

Figure 27 : Go1 est compatible ROS

```cpp
#include "ros/ros.h"
#include "geometry_msgs/Twist.h"

int main(int argc, char **argv) {
    ros::init(argc, argv, "go1_controller_cpp");
    ros::NodeHandle nh;

    ros::Publisher cmd_pub = nh.advertise<geometry_msgs::Twist>("/cmd_vel", 10);
    ros::Rate loop_rate(10);

    while (ros::ok()) {
        geometry_msgs::Twist msg;
        msg.linear.x = 0.5;  // avancer
        msg.angular.z = 0.0;
        cmd_pub.publish(msg);
        ros::spinOnce();
        loop_rate.sleep();
    }
    return 0;
}
```

```python
import rospy
from geometry_msgs.msg import Twist

rospy.init_node("go1_controller")
pub = rospy.Publisher("/cmd_vel", Twist, queue_size=10)

# Déplacer le robot en avant
msg = Twist()
msg.linear.x = 0.5
msg.angular.z = 0.0
pub.publish(msg)
```

Figure 28 : Exemple de programmes (C++ à gauche), (Python à droite)

Go intègre un broker MQTT

Enfin, une autre possibilité d'interagir avec le robot consiste à utiliser le protocole MQTT nativement présent dans la Raspberry de Go1. Pour rappel, le protocole MQTT (Message Queuing

Telemetry Transport) est un protocole léger de messagerie publish/subscribe utilisé principalement pour l'Internet des objets (IoT). Il permet à des appareils (capteurs, objets connectés, serveurs) d'échanger des messages via un broker central, avec un minimum de bande passante et de ressources [Vidéo 15].

Dans notre cas, le broker est déjà installé dans la Raspberry. Il suffit d'un client MQTT, par exemple implanté dans un ordinateur, pour souscrire à une information, par exemple pour connaitre quel est l'état de charge de la batterie. Le client peut également publier des données, par exemple pour réaliser une action (faire avancer le robot…).

Il existe de nombreux logiciels et bibliothèques qui permettent de créer des clients MQTT. Par exemple « Node-RED » avec interface graphique, « Mosquitto » en C/C++, ou encore « paho » en python (Figure 29). L'implémentation peut se faire sur des systèmes IoT (et grand public comme Arduino, ESP32, Raspberry…), ou sur des systèmes d'automatismes industriels professionnel (CPU S7 de Siemens, comme étudié dans cet article).

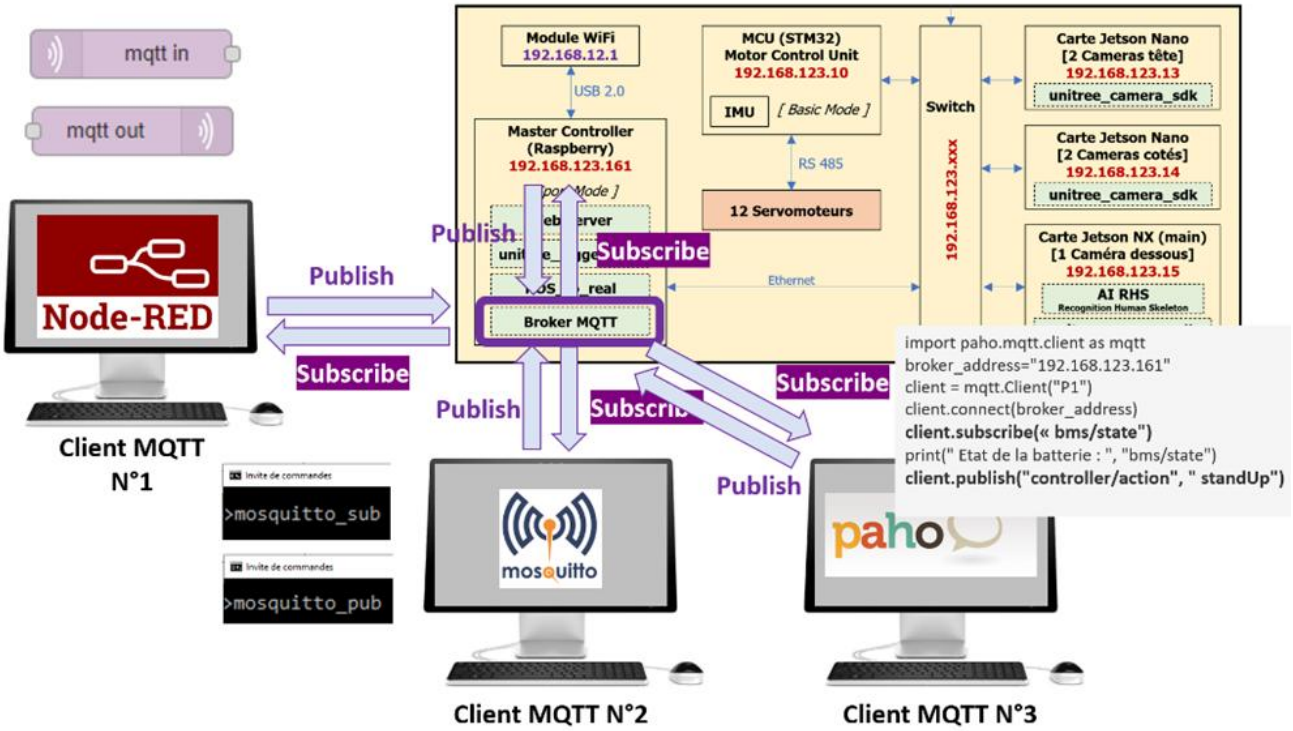

Figure 29 : Différents clients MQTT, pour le broker de Go1

3.2 - Commande du robot avec MQTT et Automate (Siemens)

Les CPU de la gamme S7-1200 et S7-1500 ne disposent pas directement d'un client MQTT intégré dans le firmware, mais il est possible d'inclure facilement des bibliothèques TIA Portal qui comportent des blocs fonctionnels MQTT. Dès la version V15, il est possible d'implémenter des clients MQTT dans des CPU 1200 (Figure 30). Dans les versions suivantes, les bibliothèques n'ont cessé de s'enrichir.

Figure 30 : MQTT Client Siemens TIA V15

Un automate [sur cet exemple Siemens, mais c'est aussi valable pour Schneider (Modicon M262, M241…), Beckhoff (TwinCAT 3), B&R Automation, Omron NX/NJ, ou encore Wago via Codesys], pouvant implémenter un client MQTT. Ce client peut assurer le contrôle du robot-chien Go1 (Figure 31) via le broker intégré dans la Raspberry, et prêt à l'emploi.

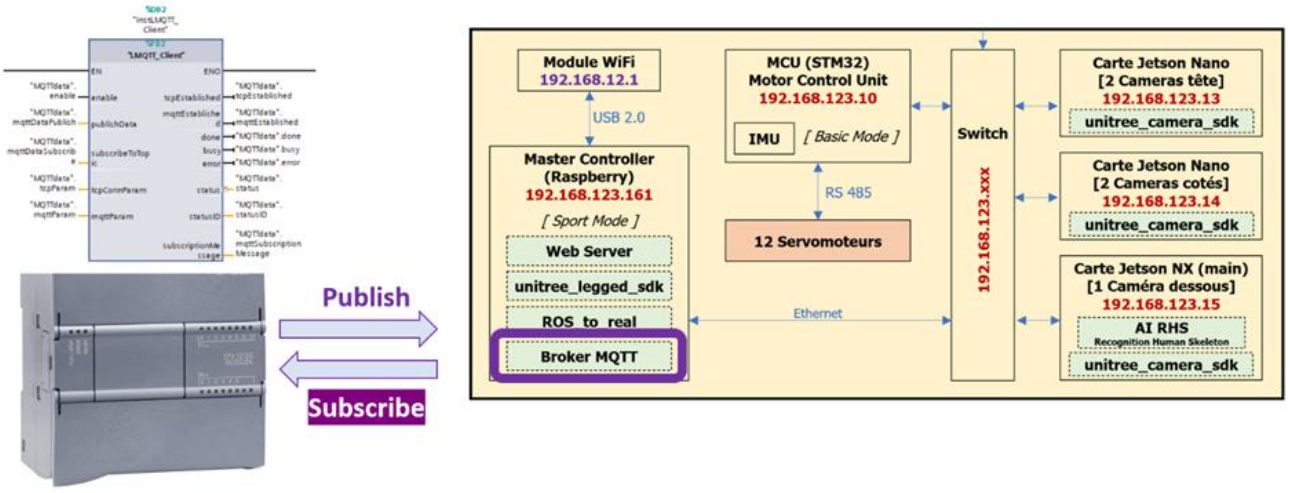

Figure 31 : Contrôle de Go1 avec un automate Siemens, via MQTT

3.3 - Exemples d'utilisation de Go1 en SAE de BUT2 ou BUT3

En IUT (Institut Universitaire de Technologie), une SAE signifie Situation d'Apprentissage et d'Évaluation. C'est un dispositif pédagogique nouveau dans la réforme des BUT (Bachelor Universitaire de Technologie), mis en place depuis 2021. Une SAE est généralement construite sous

forme de projet, d'étude de cas ou de problème à résoudre. Les étudiants mobilisent plusieurs ressources (cours théoriques, TD, TP) pour réussir la tâche demandée.

Travail préparatoire (activité de TP classique) :

TP1 : Identification des topics utiles pour commander le robot.

Le premier est appelé 'controller' avec un sous-topic 'action'. Il permet de commander, différents modes de fonctionnement. La syntaxe des différentes commandes (Debout, couché, marche, etc.) est détaillée dans différents fichiers source (Figure 32).

```
import { EventEmitter } from "events";
import { Go1MQTT } from "./mqtt/go1-mqtt";
import { Go1State, getGo1StateCopy } from "./mqtt/go1-state";
import { IClientOptions } from "mqtt";

export enum Go1Mode {
  dance1 = "dance1",
  dance2 = "dance2",
  straightHand1 = "straightHand1",
  damping = "damping",
  standUp = "standUp",
  standDown = "standDown",
  recoverStand = "recoverStand",
  stand = "stand",
  walk = "walk",
  run = "run",
  climb = "climb",
}

export class Go1 extends EventEmitter {
  mqtt: Go1MQTT;
  go1State: Go1State;

```

Figure 32 : Le topic 'controller/action'

Autre topic indispensable, le 'contrôleur/stick' (comprendre Joystick). Il doit contenir 4 valeurs de type float : Lx, Rx, Ry et Ly correspondant respectivement aux 4 mouvements possibles sur la manette.

- Lx, correspond au joystick gauche sur l'axe X. Sa valeur est :
  - 0 en position milieu (sans rien toucher).
  - +1 en position extrême à droite.
  - -1 en position extrême à gauche (Figure 33).

Sur le robot cette commande permet le décalage gauche/droite.

Figure 33 : Commande joystick Lx

- Rx, correspond au joystick droit sur l'axe horizontal. Les valeurs sont à nouveau (en float) comprises entre -1 et +1. Cette commande permet la rotation gauche/droite du robot-chien.
- Ry correspond au joystick droit sur l'axe vertical. Les valeurs sont elles aussi comprises entre -1 et +1.
- Ly, correspond au joystick gauche sur axe vertical avec des valeurs entre -1 et +1. Il permet de faire avancer ou reculer le robot.

Le topic ‘face_light/color’ qui va commander la led multicolore dans la tête du robot n’est pas indispensable mais agréable à utiliser. On lui envoie 3 valeurs en Uint (0 à 255) correspondant à l’intensité de couleur codée en RGB.

Le topic en lecture ‘bms/state’ permet de connaître l’état du Battery Manager Système (pourcentage de charge de la batterie, valeur du courant consommé…).

TP2 : Commande du robot avec un client léger.

Suivant le niveau de connaissances des étudiants, il est possible d’utiliser Node-RED, ou simplement des lignes de commande avec Mosquitto (Figure 34) [Vidéo 16].

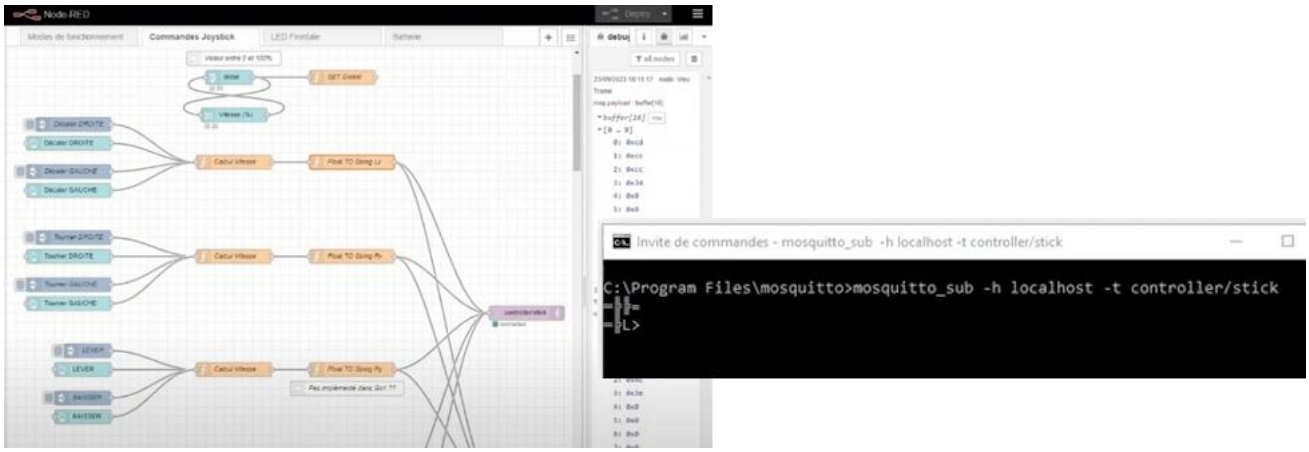

Figure 34 : Commandes MQTT avec Node-RED ou Mosquitto

TP3 : Commande du robot avec un automate

L’envoi et la réception des topics est maintenant réalisé par un client MQTT à implémenter dans un automate. Avec Siemens et une petite CPU S7-1200, il est possible d’utiliser un bloc fonction « LMqtt_Publisher » (Figure 35) [Vidéo 16].

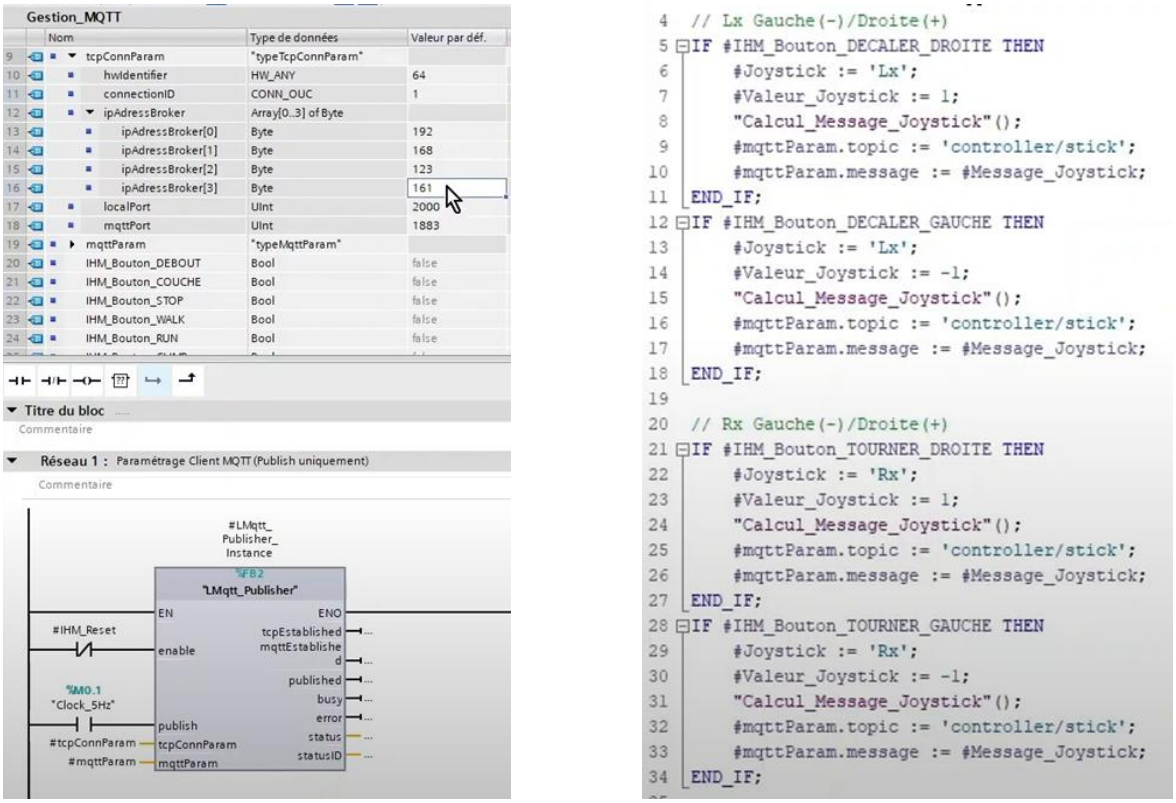

Figure 35 : Exemple de code pour CPU Siemens « LMqtt_Publisher »

Exemples de projets complets (SAE) :

1. Le robot-chien guide d'aveugle (Parcours All)

Sans pouvoir remplacer complètement l'interaction émotionnelle et intuitive d'un animal vivant, un robot-chien pourrait compléter certains aspects fonctionnels d'un chien guide pour personnes ayant une déficience visuelle : navigation, détection d'obstacles, sécurité…

L'objectif du projet est donc la modification du robot Go1 pour assurer cette fonction de guide. Les différents éléments composant le projet sont (Figure 36) :

- Une caméra « intelligente » (Cognex) pour détecter des particularités dans l'environnement (obstacles, panneaux de signalisation, feu tricolore…)
- Le robot-chien équipé d'une poignée haute pour guider le non-voyant.
- Une carte sonore pour informer de dangers ou situations particulières.
- L'automate monté sur une platine et fixée sur le dos du chien.
- Une télécommande sans fils pour que l'utilisateur puisse interagir avec le robot (Marche/arrêt, pause, changement de mode de fonctionnement…)

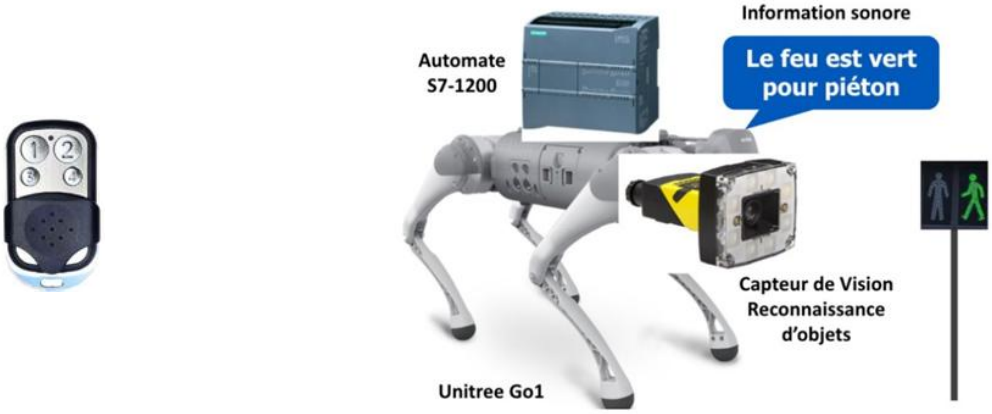

Figure 36 : Le robot-chien pour guide d'aveugle

2. Le robot-chien vigile (Parcours All)

Un robot-chien vigile peut améliorer la sécurité d'un espace en patrouillant, détectant des intrusions et en réduisant les risques pour les humains. Il peut patrouiller sans pause, sans fatigue, ni distractions. Équipé de capteurs adéquats, il peut identifier rapidement des intrus ou des comportements suspects (cf chapitre en début d'article avec la surveillance de la résidence de D. Trump).

L'objectif du projet est donc la modification du robot Go1 pour assurer cette fonction de surveillance. Les différents éléments composant le projet sont :

- Un lecteur de QR-codes (Cognex) pour scanner les badges et vérifier l'identité des personnes interceptées.
- Des capteurs ultrasons pour détecter une présence anormale dans l'environnement.
- Le robot-chien programmé pour réaliser un parcours en autonomie.
- Une carte sonore avec haut-parleur pour demander des informations (Arrêtez-vous, Scannez votre badge…)
- L'automate monté sur une platine et fixée sur le dos du chien.
- Un gyrophare et une alarme sonore pour signaler un danger, ou une tentative de vol du chien.

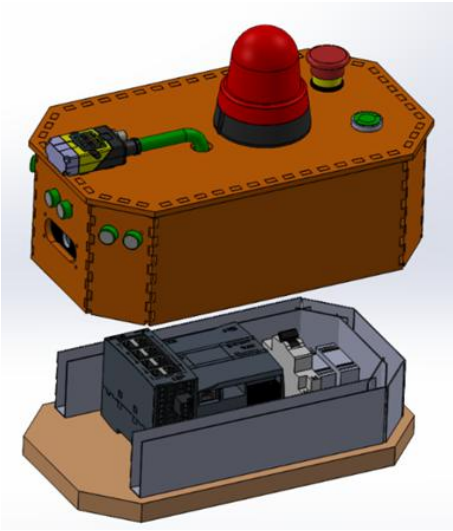

Figure 37 : La platine montée sur le dos du robot-chien vigile

3. Amélioration du robot d'origine (Parcours ESE)

Avec le SDK et/ou ROS sur Go1, il est possible d'aller bien plus loin que l'utilisation « standard » avec la télécommande ou l'application. Cela permet « d'ouvrir » le robot et de programmer ses comportements, ses capteurs et son contrôle. Voici quelles possibilités :

A. Contrôle bas niveau (SDK)

- Envoi directement de commandes aux moteurs (torque, vitesse, position).
- Modification des gains de contrôle (Kp, Kd) pour ajuster la rigidité ou la souplesse des mouvements.
- Lire les états en temps réel : positions articulaires, vitesses, courants moteurs, IMU, batterie…et les transmettre à une application « perso »
- Créer ses propres modes de locomotion : trot, pas, saut, montée/descente d'escaliers… avec des paramètres différents de ceux par défaut.

Exemples :

- Programmer un nouveau mode « sprint » en augmentant la fréquence de pas et la puissance ou couple des moteurs. Faire des mesures de vitesses et établir de nouveaux records (Figure 38).

Figure 38 : Qui sera le plus rapide ?

- Programmer un nouveau mode « climb » en modifiant les angles des articulations et séquencement des consignes pour franchir un parcours très accidenté. Faire un concours du robot qui réussira le mieux le parcours, à l'image des compétitions officielles (Figure 39) [Vidéo 17].

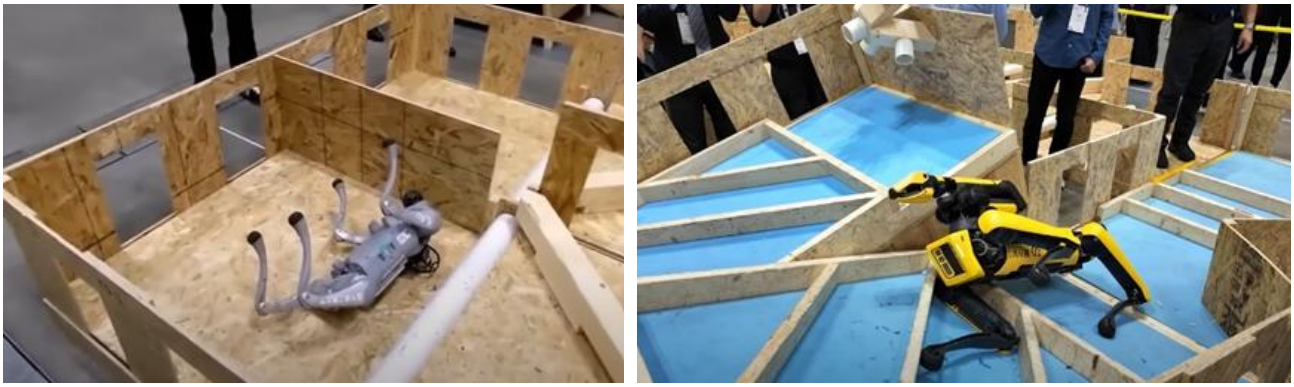

Figure 39 : IEEE International Conference on Robotics and Automation (ICRA)

B. Contrôle haut niveau (ROS / middleware)

- Envoi de commandes de déplacement globales : avance, tourne, va à tel point…
- Planification de trajectoire : définir une trajectoire que le robot doit suivre.
- Navigation autonome (SLAM) : utiliser ROS pour intégrer un LiDAR/caméra et permettre au robot de se localiser et d'éviter les obstacles.
- Perception avancée : reconnaissance d'objets/personnes, suivi de cibles, interaction avec l'environnement.

On notera que les nouvelles versions de Unitree Go2, n'intègre plus directement des commandes via MQTT (sauf pour la mise à jour du firmware par OTA). L'accès au système se fait désormais via ROS2 ou DDS (Figure 40). Le Data Distribution Service (DDS) est un middleware de communication temps réel. C'est un système standardisé (défini par l'OMG – Object Management Group) utilisé en robotique et dans l'industrie pour permettre à plusieurs programmes, capteurs et robots de communiquer efficacement entre eux sans avoir besoin de gérer les connexions directement.

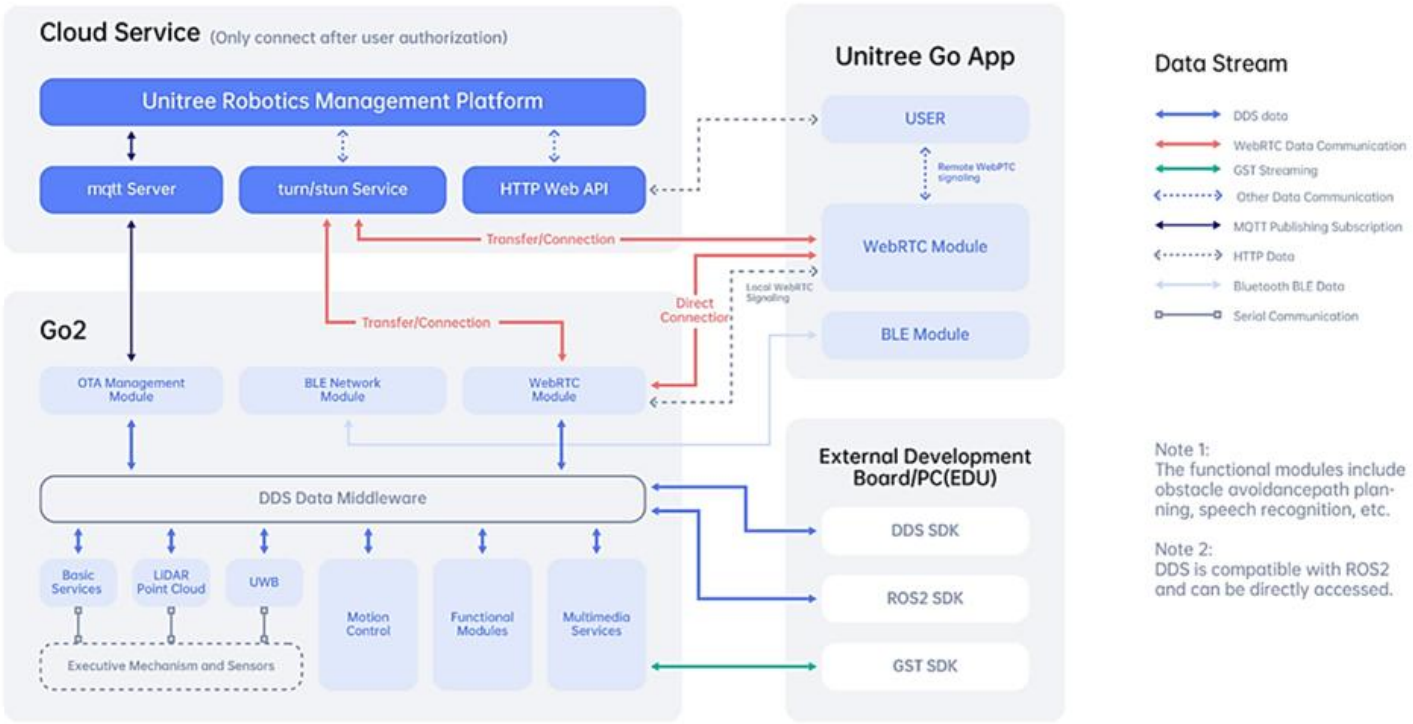

Figure 40 : Architecture de Unitree Go2

4 - Conclusion

Les robots quadrupèdes représentent aujourd'hui un domaine en pleine expansion, combinant avancées mécaniques, électroniques et informatiques pour offrir des machines capables de naviguer dans des environnements complexes, de coopérer avec des humains et d'exécuter des tâches variées. Du haut de gamme comme Spot de Boston Dynamics, en passant par les solutions accessibles pour l'éducation et la recherche, ou les projets DIY open-source, l'écosystème est à la fois riche et dynamique.

Sortis de la niche (jeu de mots) des systèmes techniques gadgets ou sans aucune utilité, les robots quadrupèdes sont désormais là et il serait malvenu de les ignorer. Véritables concentrés de technologie, ils dénotent cependant des systèmes généralement étudiés dans les établissements scolaires. D'ailleurs en IUT, sont-ils spécifiques aux enseignements du parcours ESE (Systèmes Embarqués) ou AII (Automatisme) ? Le débat est lancé… Et j'entends les EME (Energie) au fond de la salle : La gestion de l'énergie, l'autonomie du chien, la batterie, ce n'est pas un sujet ? Il mange des croquettes ?

A l'heure où nos étudiants doivent choisir une « spécialité » de plus en plus tôt, la réalité nous rappelle que notre environnement technologique est plus que jamais multidisciplinaire. Et bon nombre de nos étudiants, nous disent « Mais pourquoi je dois choisir un parcours ? Ils sont tous intéressants, j'ai envie de tous les faire ! ».

C'est peut-être là la plus grande leçon que nous donnent ces robots : ils ne sont pas uniquement mécaniques, électroniques, informatiques ou énergétiques, mais bien tout cela à la fois. Leur étude nous rappelle qu'au-delà des spécialités et des parcours, c'est la capacité à relier les disciplines qui permettra aux ingénieurs et techniciens de demain de comprendre, d'innover et d'imaginer les usages futurs de ces nouvelles technologies.

Références

[Vidéo 1] {Boston Dynamics} Do You Love Me?
https://www.youtube.com/watch?v=fn3KWM1kuAw

[Vidéo 2] {Boston Dynamics} Hey Buddy, Can You Give Me a Hand?
https://www.youtube.com/watch?v=fUyU3lKzoio

[Vidéo 3] {Cheddar} How Boston Dynamics' Spot enhances military safety and operations
https://www.youtube.com/watch?v=APSUrJVFvCQ

[Vidéo 4] {Axle Foley} UniTree Go1 with a Gun HD quality
https://www.youtube.com/watch?v=fH-awDeKl9Q

[Vidéo 5] {Throwflame} The Robot Dog With A Flamethrower | Thermonator
https://www.youtube.com/watch?v=rj9JSkSpRlM

[Vidéo 6] {Aaed Musa} I Built a Robot Dog and Made it Dance
https://www.youtube.com/watch?v=VhUvoV5XyRg

[Vidéo 7] {James Bruton} Playlist openDog V3
https://www.youtube.com/playlist?list=PLpwJoq86vov8uTgd8_WNgBHFpDYemO-OJ

[Vidéo 8] {Advanced Hobby Lab} Full build of SpotMicro
https://www.youtube.com/playlist?list=PLAPTbzQ9K5TCaQBTw4c2F2jYY1ZrWi0pl

[Vidéo 9] {Maker 101} Build a simple 3D dog robot and control it
https://www.youtube.com/watch?v=NlUgbXlzpAw

[Vidéo 10] {Josh Pieper} One year of building a robot dog
https://www.youtube.com/watch?v=ePdGshbKR-Q

[Vidéo 11] {Massachusetts Institute of Technology (MIT)} Backflipping MIT Mini Cheetah
https://www.youtube.com/watch?v=xNeZWP5Mx9s

[Vidéo 12] {Gonzalo Espinoza Graham} Building a Robot Dog in my cave
https://www.youtube.com/playlist?list=PL55VZ7oDEoRnXKSYTiGlNazkJCWyt8zJR

[Vidéo 13] {Kawasaki Group Channel} が提案する未来のパーソナルモビリティ「CORLEO」
https://www.youtube.com/watch?v=vQDhzbTz-9k

[Vidéo 14] {Hervé Discours} Unitree Go1 Essai Comparatif
https://www.youtube.com/watch?v=y18JpFnG5qQ

[Vidéo 15] {Hervé Discours} MQTT Initiation - Broker Mosquitto
https://www.youtube.com/watch?v=xpsZa3N3Ogo

[Vidéo 16] {Hervé Discours} Unitree Go1 Commande MQTT Node-Red / TIA Portal LMqtt Client PLC S7-1200 Siemens
https://www.youtube.com/watch?v=_3fwFAqhA84

[Vidéo 17] {IEEE Spectrum} Robot Dog Obstacle Course
https://www.youtube.com/watch?v=Zv0VkKflKaU
https://www.youtube.com/watch?v=WUkW-RRm7Yw

[Vidéo 18] {iFixit} Démontage de l'Unitree Go2 : l'analyse approfondie d'un expert en robotique
https://www.youtube.com/watch?v=YjVbW6Fc11Y