

---

## 1. Introduction et Problématique Centrale

L'optimisation du Raspberry Pi 5 (RPI 5) pour la robotique humanoïde est un impératif technique, car la stabilité dynamique d'un bipède exige une boucle de contrôle (Whole-Body Control - WBC) fonctionnant à une fréquence minimale de 1 kHz, soit une période de calcul de 1 000  $\mu$ s. À cette fréquence, chaque microseconde de latence est critique, et l'architecture spécifique du RPi 5 impose des mesures radicales pour garantir le déterminisme temporel.

Voici une analyse détaillée des défis et des solutions d'optimisation pour transformer ce processeur en un contrôleur temps réel dur.

## 2. Architecture Matérielle et Communication (Le Système Nerveux)

### 1. Problème Structurel : L'Architecture Désagrégée et le RP1

Bien que le RPi 5 soit doté d'un processeur Broadcom BCM2712 (Cortex-A76) nettement plus puissant que ses prédécesseurs, il introduit une complexité architecturale préjudiciable au temps réel.

- L'indirection du Southbridge RP1 : Contrairement aux modèles précédents, les périphériques (GPIO, SPI, I2C) ne sont plus intégrés au SoC principal mais déportés sur une puce distincte, le RP1, connectée via un lien PCIe 2.0 x4. Cette séparation physique crée un temps d'arbitrage incompressible pour chaque transaction.
- Le "Temps Mort" structurel : Des analyses instrumentales révèlent un temps mort d'environ 12  $\mu$ s entre l'activation de la ligne Chip Select et le début effectif du signal d'horloge, contre seulement 1  $\mu$ s sur le RPi 4.
- Impact arithmétique : Pour un robot à 12 degrés de liberté nécessitant 24 transactions par cycle (lecture et écriture), ce délai s'accumule pour atteindre 288  $\mu$ s, consommant près de 30 % du budget temps total avant même le moindre calcul algorithmique.
- Le Paradoxe de la performance : Sous charge, le RPi 5 peut présenter des pics de latence atteignant 800  $\mu$ s, là où le RPi 4 plafonnait à 200  $\mu$ s, ce qui est catastrophique pour une boucle de 1 000  $\mu$ s.

### 2. Système Nerveux : Communication et Performance

Le développement de la robotique humanoïde dynamique impose une transition d'une architecture de communication classique vers un "système nerveux" à haute bande passante. Les sources soulignent une rupture technologique majeure : l'abandon du bus SPI au profit du couple PCIe et CAN-FD pour garantir le déterminisme et la réactivité nécessaires à l'équilibre bipède.

#### 2.1. L'Obsolescence du SPI sur Raspberry Pi 5

L'utilisation du bus SPI (Serial Peripheral Interface) sur le Raspberry Pi 5 constitue désormais un goulot d'étranglement critique pour le contrôle à haute fréquence (1 kHz et plus).

- L'Effet Chiplet et le Southbridge RP1 : Contrairement aux générations précédentes, le SoC BCM2712 du RPi 5 déporte les fonctions d'entrées/sorties (dont le SPI) vers une puce distincte, le RP1, reliée par un lien PCIe 2.0 x4. Cette indirection matérielle introduit un temps d'arbitrage incompressible.
- Le "Temps Mort" (Dead Time) : Des mesures instrumentales révèlent un délai d'environ 12  $\mu$ s entre l'activation du signal Chip Select (CS) et le début effectif de l'horloge, ainsi qu'entre les paquets. Sur un RPi 4, ce délai n'était que de 1  $\mu$ s.
- Saturation du Budget Temps : Pour un robot à 12 degrés de liberté (DoF), 24 transactions (lecture/écriture par cycle) génèrent un retard cumulé de 288  $\mu$ s. Cela consomme près de 30 % du temps CPU d'une boucle de 1 ms avant même le traitement de la moindre donnée utile, plafonnant la fréquence de contrôle stable à 500 Hz, ce qui est insuffisant pour la locomotion dynamique.

## 2.2. La Révolution PCI Express (PCIe) et le DMA

Pour briser ce plafond de latence, l'exploitation du connecteur PCIe 2.0 x1 exposé par le RPi 5 est indispensable.

- Accès Direct à la Mémoire (DMA) : Contrairement au SPI qui sollicite le CPU pour chaque octet, les contrôleurs CAN-FD basés sur PCIe utilisent le DMA pour écrire les données capteurs directement dans la RAM du système hôte sans intervention du processeur.
- Réduction Drastique de la Latence : La latence d'initiation d'une transaction sur PCIe chute à moins de 1  $\mu$ s. De plus, l'utilisation du "coalescing" d'interruptions réduit la surcharge liée aux commutations de contexte.
- Efficacité du Bus : Alors que les adaptateurs SPI ou USB saturent souvent à 20-50 % avec une gigue importante, un lien PCIe permet de maintenir une charge de bus de 95 % à 98 % sans perte de trames.

## 2.3. CAN-FD : Densification et Commande Unifiée

Le protocole CAN-FD (Flexible Data-rate) complète cette architecture en tant que bus de terrain robuste pour la commande distribuée vers les membres.

- Densification de la Charge Utile : Le passage de 8 octets (CAN 2.0) à 64 octets par trame permet d'implémenter des trames de commande unifiées. Une seule trame peut désormais encapsuler la position cible, la vitesse, le couple et les gains (KP, KD).

- Réduction de l'Overhead : Cette capacité divise par deux ou trois le nombre de transactions requises par cycle de contrôle. En regroupant les données, on réduit le nombre d'arbitrages sur le bus et on garantit la cohérence atomique des commandes appliquées à chaque articulation.
- Accélération du Débit : Le CAN-FD maintient une phase d'arbitrage robuste mais bascule sur un débit de 5 à 8 Mbps pour la phase de données, réduisant radicalement le temps d'occupation du bus pour chaque message.

En somme, si le bus SPI est un sentier de randonnée encombré où chaque marcheur doit s'arrêter à chaque barrière, l'architecture PCIe/CAN-FD est une autoroute automatisée où les données circulent en convois massifs sans jamais marquer l'arrêt. [Source : Analogie basée sur les concepts de latence transactionnelle et de densification des sources 8, 197, 199].

### 3. Optimisation Logicielle (Kernel & OS)

#### 2. Optimisation du Noyau (Kernel Tuning)

L'application du patch PREEMPT\_RT est la première étape pour transformer Linux en système temps réel, mais elle est insuffisante sur le RPi 5 sans un partitionnement strict des ressources via le fichier cmdline.txt.

- Isolation (isolcpus=2,3) : Ce paramètre physique exclut les coeurs 2 et 3 de l'ordonnanceur général de Linux. Cela empêche le système de déplacer des processus non critiques (WiFi, SSH, mises à jour) sur ces coeurs, évitant ainsi l'invalidation des caches L1/L2 qui provoque des délais de rechargement mémoire imprévisibles.
- Mode "Full Tickless" (nohz\_full=2,3) : Le noyau Linux génère normalement une interruption d'horloge périodique (le "tick") pour gérer le temps. Le mode Tickless désactive cette interruption sur les coeurs isolés, éliminant une source majeure de gigue (jitter) et permettant une exécution "bare-metal" du code de contrôle.
- Callbacks RCU (rcu\_nocbs=2,3) : Les opérations de maintenance mémoire du noyau (Read-Copy-Update) sont déportées vers les coeurs 0 et 1. Cette mesure évite les blocages ("stalls") imprévisibles lors des accès mémoire intensifs du solveur.
- Gouverneur de fréquence (cpufreq performance) : Il est crucial de bloquer la fréquence CPU à son maximum (2,4 GHz). Cela élimine la latence liée aux changements de vitesse d'horloge (DVFS) et à la synchronisation des boucles à verrouillage de phase (PLL).

#### 3. Gestion des Interruptions (IRQ Affinity)

Par défaut, le contrôleur d'interruptions distribue les signaux matériels sur n'importe quel coeur disponible, ce qui sature le système de requêtes contradictoires. Une stratégie de ségrégation physique est nécessaire :

- Coeurs 0 et 1 (Intendance) : Ils absorbent tout le "bruit" du système, incluant les interruptions réseau, USB et le logging.

- Coeur 2 (Pré-traitement) : L'interruption critique provenant de la carte CAN-FD (signaler l'arrivée de données capteurs) doit lui être dédiée. Il prépare les données et les injecte dans la mémoire partagée.
- Coeur 3 (Sanctuaire) : Ce coeur est réservé exclusivement au fil de contrôle temps réel exécutant le solveur QP (Quadratic Programming) ou OCS2. En étant libéré des interruptions matérielles et des tâches système, il peut résoudre les équations de dynamique inverse avec une latence minimale et constante.

Synthèse de la configuration recommandée (cmdline.txt)

Pour stabiliser la boucle à 1 kHz, les paramètres suivants doivent être ajoutés : `isolcpus=2,3 nohz_full=2,3 rcu_nocbs=2,3 cpufreq.default_governor=performance`.

## 4. Sécurité Électrique et Puissance (Architecture 48V)

### 3. Sécurité de l'Architecture de Puissance (48V)

L'adoption d'une architecture 48V (généralement via des packs LiPo 13S) est impérative pour atteindre les densités de puissance nécessaires à la locomotion dynamique des robots humanoïdes. Cependant, ce passage transforme le robot en un système à haute énergie comparable aux véhicules électriques légers, où la faible résistance interne des batteries peut générer des courants de court-circuit de plusieurs centaines d'ampères, imposant des protocoles de sécurité matérielle rigoureux.

#### 3.1 Gestion des Courants d'Appel : Le Circuit de Pré-charge

La connexion directe d'une batterie 48V à un bus alimentant de nombreux variateurs de moteurs (comme les Robstride 04) provoque un arc électrique destructeur, appelé "inrush current".

- Physique du phénomène : Les contrôleurs moteurs possèdent des banques de condensateurs de filtrage massives (souvent >2000 F). Au moment de la mise sous tension, si les condensateurs sont déchargés, le système se comporte comme un court-circuit. Le courant instantané, limité uniquement par la résistance des câbles, peut atteindre environ 1000A.
- Solution technique : Un circuit de pré-charge est obligatoire pour éviter de souder définitivement les contacts des relais principaux par électro-érosion. Il se compose d'un relais secondaire monté en série avec une résistance de puissance (typiquement entre 20 et 50 Ohms, de type céramique ou bobinée) placée en parallèle du contacteur principal.
- Séquence de démarrage : Une logique de contrôle (via microcontrôleur ou relais temporisé) doit d'abord fermer le circuit via la résistance. Une fois que la tension du bus atteint 90 à 95% de la tension batterie, le contacteur principal est fermé sans arc, puis le relais de pré-charge est ouvert.

#### 3.2 Protection contre les Surtensions : Le Hacheur de Freinage (Braking Chopper)

En robotique dynamique, les moteurs sont des charges dites "4 quadrants" : ils consomment de l'énergie, mais en produisent également lors des phases de freinage ou d'impact.

- Énergie régénérative : Lors d'une décélération brutale ou d'une réception de saut, les moteurs agissent comme des générateurs et renvoient un courant inverse massif vers le bus DC (Back-EMF). Si la batterie ne peut absorber ce courant assez vite (ou si le système de gestion de batterie, BMS, coupe le circuit), la tension du bus grimpe instantanément.
- Seuil critique : Sur un bus 48V, la tension peut dépasser 60V, franchissant la tension de claquage des MOSFETs des variateurs et entraînant leur destruction immédiate.
- Fonctionnement du module : Le Braking Chopper surveille en permanence la tension. Dès qu'un seuil critique est atteint (généralement 52V), un transistor de puissance (IGBT ou MOSFET) dérive l'énergie excédentaire vers une résistance de dissipation externe qui transforme le surplus en chaleur, agissant comme une soupape de sécurité électrique.

### 3.3 Architecture d'Arrêt d'Urgence (E-Stop) et Intégrité du Circuit

La coupure d'urgence d'un courant continu (DC) de forte puissance est physiquement plus complexe que celle d'un courant alternatif (AC) car il n'y a pas de passage par zéro pour éteindre l'arc électrique.

- Limites des relais standards : Les relais classiques ne peuvent pas couper un arc DC de 48V sous forte charge ; l'arc peut se maintenir, brûlant les contacts et gardant le robot sous tension malgré l'appui sur le bouton d'urgence.
- Contacteurs dédiés : Il est impératif d'utiliser des contacteurs DC industriels (ex: Albright SW80 ou Gigavac GV200) équipés de soufflages magnétiques. Ces dispositifs utilisent des aimants permanents pour "pousser" et allonger l'arc électrique via la force de Lorentz jusqu'à sa rupture physique.
- Redondance et STO : Conformément à la norme ISO 13849, le circuit doit être à double canal (redondant). En complément, les variateurs doivent supporter la fonction Safe Torque Off (STO), qui coupe matériellement l'alimentation des pilotes de grille des transistors de puissance, garantissant qu'aucun couple ne peut être produit, même en cas de plantage du logiciel de contrôle.
- Avertissement sur les SSR : Les relais statiques (SSR) sont à proscrire pour la coupure de sécurité principale car leur mode de défaillance typique est le court-circuit (restant fermé), ce qui est inacceptable pour un système de sécurité.

Pour stabiliser davantage le système lors des appels de courant violents (pics de 10kW lors d'un saut), l'ajout de supercondensateurs en parallèle peut servir de tampon, protégeant la batterie du stress thermique et évitant les chutes de tension (brownouts) qui pourraient réinitialiser les calculateurs embarqués.

## 5. Ingénierie Détaillée et Risques Techniques

### 5. Synthèse des Risques Techniques et Leçons (Reconstruction du Tableau)

Le tableau suivant réorganise les "Bitter Lessons" techniques dispersées dans le document original. Voici une extension détaillée des spécifications techniques de votre projet, structurée selon les sources pour offrir une base d'ingénierie robuste.

#### 1. Actionneurs (Thermique) : La Barrière de l'Endurance

Les actionneurs Quasi-Direct Drive (QDD) sont essentiels pour la transparence mécanique, mais leur faible réduction impose des courants élevés pour le maintien de posture.

- **Problème Physique** : La chaleur est générée par les pertes Joule. Comme le couple est proportionnel au courant, la chaleur augmente au carré du couple produit.
- **Réalité du Matériel** : Les aimants en Néodyme (NdFeB) de grade standard perdent leur force magnétique de façon irréversible dès 80-100°C, ce qui réduit définitivement les performances du moteur.
- **Règle d'Or** : Selon la loi d'Arrhenius, une augmentation de 10°C au-dessus de la température nominale divise par deux la durée de vie des isolants des bobinages.

#### 2. Matériaux : Résilience Structurelle et Thermique

L'utilisation de matériaux inadaptés à la chaleur peut transformer une structure rigide en un système mou et imprécis.

- **Limites du PLA** : Le PLA subit du fluage (déformation lente sous charge) dès 55-60°C. La carcasse d'un moteur en charge peut facilement atteindre cette température, risquant de bloquer les pales des ventilateurs ou de désaligner les arbres moteurs.
- **Solution PAHT-CF** : L'utilisation du PAHT-CF (Nylon haute température renforcé de carbone) est impérative pour les carénages et pièces structurelles grâce à sa Température de Déflexion sous Charge (HDT) de 194°C.
- **Gestion par PCM** : Les matériaux à changement de phase (PCM) comme la paraffine agissent comme des "condensateurs thermiques", stockant l'énergie lors des pics d'effort sans augmenter la température. Cependant, ils nécessitent des temps de repos pour se solidifier à nouveau et libérer cette chaleur.

#### 3. Refroidissement : Évacuation Active Obligatoire

Le refroidissement passif est insuffisant pour les densités de puissance requises par la marche dynamique.

- Ventilation Haute Pression : Dans un robot compact, les flux d'air sont entravés. Il faut privilégier des ventilateurs à haute pression statique (comme la série 9HV) capables de forcer l'air à travers des radiateurs denses, plutôt que des modèles à haut débit d'air à vide.
- Intégration Logicielle : Le refroidissement doit être couplé à une surveillance thermique active qui force le robot à changer de posture ou à s'asseoir si la bobine dépasse un seuil critique (ex. 80°C).

#### 4. Électronique (Bus) : L'Intégrité du Signal CAN-FD

Le passage à des débits élevés (5-8 Mbps) transforme les câbles en lignes de transmission complexes où chaque erreur physique crée une "épilepsie robotique".

- Topologie Daisy Chain : La topologie en étoile est formellement interdite car elle crée des désadaptations d'impédance massives (jusqu'à 67 % de l'énergie réfléchi). Le câblage doit être une guirlande (Daisy Chain) continue.
- Terminaisons et Stubs : Chaque extrémité du bus doit posséder une résistance de 120 Ohms. Les dérivations (stubs) vers les moteurs doivent être limitées à 30 cm maximum pour éviter les échos destructeurs.
- Terminaison Split : Pour stabiliser le bus contre le bruit de mode commun, on utilise deux résistances de 60 Ohms avec un condensateur relié à la terre au point milieu.

#### 5. EMI (Interférences) : Blindage et Protections

L'onduleur de puissance commute des courants de 100A en nanosecondes, créant un environnement électromagnétique hostile.

- Blindage Hybride : Utilisez des câbles avec Tresse (cuivre) pour la résistance mécanique et les basses fréquences, couplée à un Feuillard (aluminium) pour bloquer les hautes fréquences du PWM.
- Reprise de Masse à 360° : La terminaison du blindage ne doit jamais se faire par un fil ("queue de cochon"), car cela crée une antenne rayonnante. Il est impératif d'utiliser des presse-étoupes EMC assurant un contact circulaire complet.
- Filtrage et Flyback : Une choke de mode commun (CMC) de 51  $\mu$ H est recommandée pour filtrer le bruit moteur sans déformer le signal CAN-FD. Une protection TVS est indispensable pour absorber les surtensions de l'effet Flyback (>100V) générées lors de coupures brusques du courant dans les inductances.

#### 6. Stratégie Projet : Software-Defined David

Face à la concurrence chinoise capable de vendre des robots (ex. Unitree G1) à un prix proche de votre coût de matériaux, la stratégie doit pivoter.

- Focus Logiciel : Ne visez pas l'excellence matérielle pure. La valeur réside dans la chaîne logicielle (stack) et l'autonomie cognitive.
- Budget Cible : Maintenez un budget entre 5k en utilisant des composants "sur étagère" (COTS) et des châssis modulaires.
- Ouverture : L'avantage concurrentiel est la transparence totale des boucles de contrôle, permettant des itérations rapides là où les systèmes industriels sont des "boîtes noires".

## 6. IA, Simulation et Contrôle

### 4. Le Cerveau : IA et Transfert « Sim-to-Real »

Le transfert de politiques apprises en simulation vers le robot réel est l'étape la plus critique, car les modèles physiques « idéaux » échouent souvent face aux non-linéarités non modélisées comme les frottements, les jeux mécaniques et les délais de communication. Pour combler ce « Reality Gap », une approche hybride combinant modélisation neuronale et environnements stochastiques est employée.

#### 4.1 Actuator Nets : La Modélisation Neuronale des Moteurs

L'approche conventionnelle ( $\tau = k_t \cdot I$ ) est jugée insuffisante pour les actionneurs réductés, car elle ignore des phénomènes physiques complexes tels que les frottements de Coulomb, visqueux et de Stribeck, ainsi que l'hystérésis du réducteur planétaire et les jeux mécaniques (backlash).

- Méthodologie de capture : Pour créer un modèle fidèle, on excite l'actionneur réel avec des signaux de fréquence variable, appelés signaux « chirp », et des échelons aléatoires. On enregistre alors précisément la réponse du moteur (couple réel produit, vitesse, erreur de position) en fonction des commandes envoyées.
- Architecture du réseau : Ces données servent à entraîner un réseau de neurones temporel (généralement un LSTM ou un MLP avec historique). Ce réseau apprend la fonction de transfert complexe de l'actionneur réel.
- Intégration et impact : Ce « réseau actionneur » est ensuite intégré directement dans le simulateur (comme Isaac Lab), remplaçant le modèle moteur par défaut. Cette technique permet de réduire l'erreur de suivi de plus de 20 % à moins de 5 %, facilitant souvent un transfert « zero-shot » (sans réajustement sur le robot réel). L'agent apprend ainsi une politique qui respecte nativement les limites physiques et les saturations des moteurs réels.

#### 4.2 Randomisation de Domaine (Domain Randomization)

Pour garantir la robustesse de la politique, l'environnement de simulation doit être conçu pour être beaucoup plus hostile et bruyé que la réalité. En perturbant massivement les paramètres physiques à chaque épisode d'entraînement, le réseau de neurones apprend une politique capable d'envelopper tout le spectre des conditions réelles.



- Variabilité des paramètres physiques : Les sources recommandent des plages de variations strictes pour forcer l'adaptabilité :
  - Masse des segments : Variation de  $\pm 15\%$  à  $20\%$  pour compenser les erreurs de câblage non modélisé ou les tolérances de fabrication.
  - Friction au sol : Simulation de coefficients allant de 0.4 à 1.2 (pour mimer des surfaces allant de la glace au tapis épais).
  - Friction des joints : Ajout de bruits de l'ordre de  $\pm 0.05$  N.m liés à l'usure, la graisse ou la température.
- Gestion des latences de communication : C'est un paramètre crucial. On simule des retards de commande allant de 0 à 20 ms. Cela permet à l'IA de rester stable malgré la gigue (jitter) du noyau Linux, les délais du bus CAN et la latence intrinsèque de l'ordinateur embarqué (Raspberry Pi ou Jetson).

#### 4.3 Perturbations et Apprentissage par Curriculum

L'acquisition de réflexes de récupération (comme se rattraper après un trébuchement) nécessite l'introduction de perturbations externes dynamiques durant l'apprentissage.

- Poussées aléatoires : Le simulateur applique des forces imprévisibles sur le centre de masse (CoM) du robot. Ces « coups » obligent l'agent à découvrir des stratégies d'équilibre actif.
- Curriculum progressif : Appliquer toutes les perturbations et le bruit maximal dès le début empêcherait la convergence de l'apprentissage. On utilise donc une approche par curriculum : l'environnement est initialement « propre », puis la difficulté (amplitude des poussées, niveau de friction, latences) augmente progressivement au fil des époques d'entraînement.
- Résultat : Cette montée en puissance assure une convergence vers une politique stable capable de marcher sur des terrains accidentés ou de résister à des interactions humaines imprévues dès son déploiement physique.

#### 7. Contrôle (IA) : Comblant le "Reality Gap"

Le transfert de la simulation (Isaac Lab) vers le robot réel échoue si les modèles ignorent les frottements et les jeux mécaniques.

- Actuator Net : Remplacez les modèles moteurs idéaux par des réseaux de neurones (MLP ou LSTM) entraînés sur des signaux réels ("chirp") pour capturer l'hystérésis et les frottements non linéaires. Cela réduit l'erreur de suivi de  $20\%$  à moins de  $5\%$ .
- Randomisation Massive : Pour un transfert "zero-shot", variez aléatoirement durant l'entraînement la masse des liens, la friction au sol (0.4 à 1.2) et surtout les latences de commande (0-20 ms) liées au noyau Linux et au bus CAN.

## 7. Synthèse et Conclusion

La stabilité et la survie opérationnelle d'un robot humanoïde dynamique ne résultent pas de la performance isolée d'un composant, mais de la Chaîne de Cohérence : une synchronisation systémique entre la puissance de calcul, le déterminisme logiciel et l'intelligence motrice. Cette architecture transforme une machine potentiellement fragile en un système résilient capable d'exécuter des mouvements explosifs sans défaillance thermique ou perte de communication.

### 1. La Couche Matérielle : L'Épine Dorsale PCIe et le Bus Réflexe CAN-FD

La couche matérielle constitue le "système nerveux" du robot. Pour un humanoïde bipède, qui est structurellement un pendule inversé instable, la latence entre la perception d'un déséquilibre et l'application d'un couple correctif est le facteur de survie n°1.

- Le Goulot d'Étranglement du SPI : Sur l'architecture Raspberry Pi 5 (RPi5), l'utilisation du bus SPI pour contrôler les moteurs est proscrite pour la haute performance. L'architecture "chiplet" du SoC BCM2712 introduit un temps mort de 12  $\mu$ s par transaction. Pour un robot à 12 degrés de liberté effectuant 24 transactions par cycle, cela consomme près de 30 % du budget temps d'une boucle à 1 kHz avant même le transfert de la moindre donnée utile.

- La Supériorité du PCIe : La solution impérative réside dans l'exploitation de l'interface PCIe 2.0 x1 du RPi5. Contrairement au SPI, le PCIe permet l'Accès Direct à la Mémoire (DMA), réduisant la latence d'initiation à moins de 1  $\mu$ s et libérant le processeur de la surcharge de gestion des interruptions.

- La Densification CAN-FD : Le passage au protocole CAN-FD (5 à 8 Mbps) permet d'encapsuler l'ensemble des commandes (position, vitesse, couple, gains) dans une seule trame de 64 octets. Cela garantit la cohérence atomique des ordres envoyés à l'actionneur et divise par trois le nombre de transactions requises par rapport au CAN 2.0 classique.

### 2. La Couche Système (OS) : La Maîtrise du Déterminisme Temporel

Avoir un processeur rapide ne suffit pas si le système d'exploitation n'est pas déterministe. Une boucle de contrôle Whole-Body Control (WBC) doit s'exécuter avec une régularité de métronome à 1 kHz.

- Linux RT (PREEMPT\_RT) : L'application de ce patch est vitale pour rendre préemptibles les sections critiques du noyau. Sans lui, des pics de latence imprévisibles (> 800  $\mu$ s sur RPi5) causent des échéances manquées ("missed deadlines"), entraînant inévitablement une instabilité mécanique et la chute du robot.

- Isolation Stricte par isolcpus : Pour transformer Linux en un système quasi "bare-metal", il faut partitionner les ressources CPU. L'ajout du paramètre isolcpus=2,3 dans la configuration de démarrage retire les coeurs 2 et 3 de l'ordonnanceur général.

- La Chorégraphie des Coeurs : Une stratégie robuste consiste à "punaiser" (pinning) les tâches :

- Coeur 3 : Dédié exclusivement au fil de contrôle temps réel (solveur QP/OCS2).

- Coeur 2 : Dédié à la gestion des interruptions (IRQ) du contrôleur CAN-FD.

- Coeurs 0-1 : Gestion des tâches non critiques comme le WiFi, l'USB et les logs.

- Optimisations Agressives : L'utilisation des modes nohz\_full (désactivation du tick d'horloge périodique) et rcu\_nocbs sur les coeurs isolés élimine les dernières sources de gigue (jitter) logicielle.

### 3. La Couche IA : L'Intelligence Motrice via les Actuator Nets

Le transfert de la simulation vers le monde réel (Sim-to-Real) échoue généralement à cause du "Reality Gap" : les modèles physiques "idéaux" ne capturent pas les frottements, les jeux mécaniques (backlash) et l'hystérésis des moteurs réels.

- La Méthodologie Actuator Net : Au lieu d'équations physiques théoriques, on utilise un réseau de neurones (LSTM ou MLP) entraîné sur des données empiriques. On excite les actionneurs réels avec des signaux "chirp" (balayages de fréquence) pour enregistrer leur réponse exacte.

- Intégration dans Isaac Lab : Ce modèle neuronal est intégré directement dans le simulateur, remplaçant le modèle moteur par défaut. L'agent apprend ainsi une politique qui respecte nativement les limites thermiques, les saturations de courant et les non-linéarités des moteurs réels, réduisant l'erreur de suivi de position de 20 % à moins de 5 %.

- Randomisation de Domaine (Domain Randomization) : Pour garantir la robustesse, l'environnement d'entraînement est rendu volontairement plus hostile que la réalité. On perturbe massivement la masse des segments ( $\pm 15\text{-}20\%$ ), la friction au sol (0.4 à 1.2) et les latences de communication (0 à 20 ms). Cette approche par curriculum assure que la politique apprise pourra gérer n'importe quelle condition réelle dès son déploiement physique.