

Introspection et Débogage avec `pal_statistics` dans ROS 2 Control

Maximilien Naveau & Pierre Fernbach*

Durée

Présentation standard (≈ 20 minutes), 10 à 15 minutes de présentation + 5 à 10 minutes de questions/réponses.

Résumé (≤ 100 mots)

Cette présentation expose l'intégration de `pal_statistics` au sein de ROS 2 Control pour offrir une introspection **en temps réel** des entrées et sorties des contrôleurs. Cette fonctionnalité permet aux développeurs de surveiller les signaux internes **sans intervention utilisateur**, améliorant considérablement les workflows de débogage et de réglage. Au-delà de ROS 2 Control, `pal_statistics` peut être utilisé indépendamment pour exposer des métriques que des outils comme `PlotJuggler` peuvent analyser facilement. Nous présentons une démonstration—implémentée dans `ros2_control_demos`—où nous exécutons `ros2_control_example_1` et montrons que chaque contrôleur expose ses entrées/sorties. Un nœud additionnel est lancé pour illustrer la fonctionnalité indépendante de `pal_statistics` et son utilisation avec `PlotJuggler`.

Description détaillée (≤ 1000 mots)

Le débogage et l'introspection sont indispensables pour concevoir des systèmes robotiques fiables fonctionnant en temps réel. Dans ce contexte, il est essentiel que les outils et méthodes utilisés n'interfèrent pas avec la boucle de contrôle du robot, dont la fréquence d'exécution peut atteindre plusieurs kHz, typiquement jusqu'à 2kHz. Cette cadence élevée génère une quantité considérable de données de log, rendant crucial l'usage d'outils capables de filtrer et d'analyser efficacement ces informations tout en préservant la qualité du contrôle. Notons que l'utilisation des topics ROS 2 classiques n'est pas adaptée à ces fréquences élevées due au fait qu'ils ne sont pas temps réel.

De plus, les contrôleurs dans ROS 2 Control exposent généralement uniquement leurs interfaces de commande et d'état, tandis que les calculs intermédiaires internes restent souvent inaccessibles aux développeurs. Cette limitation complique l'identification des goulets d'étranglement, des problèmes numériques ou des erreurs de configuration lors du développement.

Pour répondre à ces enjeux, PAL Robotics a développé `pal_statistics`, une bibliothèque légère conçue pour publier efficacement des métriques et statistiques arbitraires, même à des fréquences élevées comme 2kHz. Le framework ROS 2 Control intègre désormais `pal_statistics` directement dans sa couche d'interface contrôleur, permettant à chaque contrôleur d'exposer instantanément ses entrées, sorties et états internes, sans ajout de code utilisateur.

Cette intégration apporte plusieurs bénéfices :

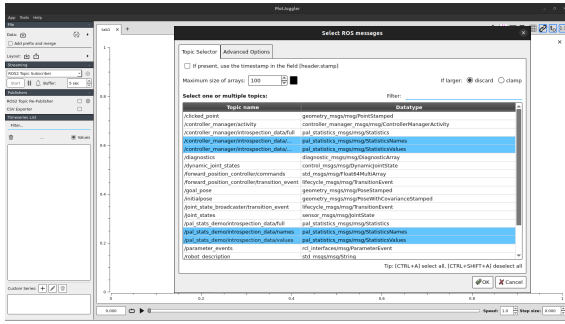
*firstname.lastname@pal-robotics.com, PAL FRANCE

- **Introspection automatique** : Chaque contrôleur publie en continu ses entrées et sorties par défaut, même à haute fréquence.
- **Transparence des métriques internes** : Les développeurs accèdent facilement aux métriques internes, ce qui accélère et clarifie le débogage.
- **Outils de visualisation adaptés** : Des outils comme PlotJuggler peuvent analyser et afficher ces métriques en temps réel grâce aux dernières évolutions ($\geq 3.10.11$).
- **Utilisation flexible** : `pal_statistics` peut également être utilisé indépendamment de ROS 2 Control dans n'importe quel nœud ROS 2, assurant une introspection cohérente sur l'ensemble des sous-systèmes robotiques.

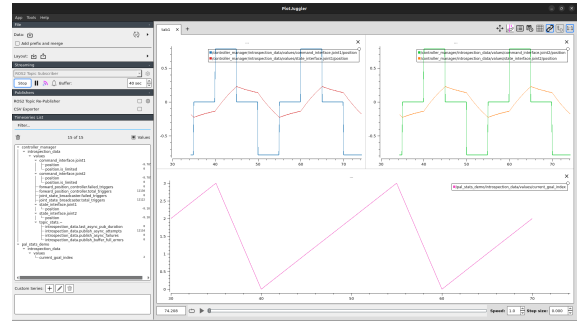
Cas d'utilisation exemple

Pour la démonstration en direct, nous utiliserons `ros2_control_example_1` pour exécuter le contrôleur de trajectoire articulaire sur le `rrbot`. Avec PlotJuggler, nous montrerons comment les signaux d'état d'entrée/sortie sont visualisés pour chaque contrôleur en temps réel. Pour illustrer davantage la flexibilité de `pal_statistics`, nous ajouterons un simple nœud Python qui utilise `pal_statistics` indépendamment de ROS 2 Control, interagissant avec la configuration `example_1`. Cela montrera comment les métriques peuvent être exposées et visualisées en dehors du framework contrôleur, offrant un workflow d'introspection cohérent.

Petit exemple Figure 1 de ce que l'on peut voir une fois les contrôleurs lancés.



(a) Sélection dans PlotJuggler des deux topics d'introspection.



(b) Tracé des signaux d'entrée/sortie des contrôleurs en haut, et tracé du nœud Python en bas.

FIGURE 1 – Exemple d'utilisation de PlotJuggler pour l'introspection en temps réel des contrôleurs ROS 2.

Points clés pour l'audience

- Comprendre comment `pal_statistics` s'intègre dans ROS 2 Control pour l'introspection automatique.
- Découvrir comment utiliser `pal_statistics` indépendamment dans des nœuds ROS 2 personnalisés.
- Apprendre à visualiser et à déboguer les internes des contrôleurs avec PlotJuggler.

Ressources

- `pal_statistics` : github.com/pal-robotics/pal_statistics
- ROS 2 Control : github.com/ros-controls/ros2_control
- PlotJuggler Changelog 3.10.11 : github.com/facontidavide/PlotJuggler/CHANGELOG.rst
- Dépôt de démonstration : github.com/ros-controls/ros2_control_demos
- Intégration ROS 2 Control : `hardware_interface/include/introspection.hpp`

- Intégration ROS 2 Control : `controller_interface/controller_interface_base.cpp`
- Dépôt de la démonstration prévue : github.com/MaximilienNaveau/roscon-fr-2025