SOLUTION ROBOTIQUE SOFTWARE

MC_RTC

Multi-contact Real-time Control Framework

Mise à jour Septembre 2025

mc_rtc (*Multi-Contact Real Time Controller*) est un **framework de contrôle robotique temps réel** pour développer des contrôleurs complexes sur robots humanoïdes, manipulateurs et systèmes multi-robots. Son **architecture modulaire** intègre GUI client/serveur, machines à états finis (FSM), contrôle par optimisation quadratique (QP), visualisation, logging et debugging. **Compatible simulateurs et robots réels**, *mc_rtc* facilite la **transition prototypage-industrie** via son approche "écrivez une fois, exécutez partout", son API C++/Python extensible et ses composants configurables en YAML.

DOMAINES D'APPLICATION

À quoi cela sert ?

- Développer rapidement des contrôleurs robotiques avancés sans expertise approfondie en optimisation.
- Assurer une transition fluide de la simulation au robot réel avec un code unique et portable.
- Améliorer la robustesse et la sécurité grâce au contrôle par optimisation sous contraintes.
- Accélérer le prototypage et réduire les coûts de développement en robotique.

Exemples de cas d'usage

- Robotique humanoïde et locomotion bipède/quadrupède
- Manipulation industrielle et collaboration homme-robot
- Assemblage et tâches multi-contacts de précision
- Coordination multi-robot en environnement industriel
- Téléopération de robots complexes

LES +

- **Architecture modulaire et intuitive** : développement simplifié via machines à états finis (FSM), interface graphique client/serveur et système de plugins.
- **Contrôle temps réel performant** : optimisation quadratique (QP) rapide gérant dynamique multi-contacts et contraintes complexes.
- **Portabilité maximale** : code unique exécutable en simulation et sur robots réels, compatible avec multiples simulateurs (Choreonoid, Gazebo, RViz).
- **Extensibilité** : API C++/Python pour créer vos propres composants et contrôleurs, support multi-robot natif. Intégrez vos robots simplement.
- **Fiabilité académique** : développé par des laboratoires de référence (CNRS-UM LIRMM, CNRS-AIST JRL), utilisé en recherche et industrie.
- **Open Source & communauté active** : **licence BSD2** permissive, documentation complète, outils de debugging/logging intégrés, écosystème en croissance.

ENVIRONNEMENT & EXIGENCES TECHNIQUES

- **C++** (17), **Python** (2 & 3), **YAML**, compatible avec **ROS** 1 & 2
- **30+ robots réels** : humanoïdes (HRP, TALOS, Atlas), manipulateurs (Panda, Fanuc, UR), quadrupèdes (Unitree), etc…
- **Simulateurs** : Gazebo, V-REP, MuJoCo, IsaacSim, AGX Dynamics, etc…
- **OS** : Linux (Ubuntu/Debian), macOS, Windows
- **Installation**: APT, VCPKG, CMake
- **Matériel requis**: station de travail classique

CONTACTS Recherche

Arnaud TANGUY
Université de Montpellier / LIRMM
Mail : arnaud.tanguy@lirmm.fr

Thomas Duvinage
Advanced Institute of Science and Technology (AIST), Tsukuba, Japan
Mail : *thomas.duvinage@aist.go.jp*

CONTACTS Collaboration

Véronique Ribière
Mail : veronique.ribiere@cnrs.fr

OPPORTUNITES DE COLLABORATION

- ✔ Preuves de concept, adaptations end-to-end robotiques sur mesure
- ✔ Co-développement ou transfert technologique
- ✔ Collaboration via projets collaboratifs

MOTS CLE

Contrôle robotique temps réel, optimisation quadratique (QP), multi-contacts, machines à états finis (FSM), multi-robot, interface graphique, C++/Python, simulation robotique, robotique humanoïde, manipulation industrielle, ROS, open-source

DOCUMENTATION

Site Web
Tutoriel
Démo en ligne

MATURITE TECNOLOGIQUE

Projet mature, utilisé extensivement pour de nombreux gros projets: H2020 COMANOID, DARPA Robotics Callenge, ANA Avatar Xprize

Accessible publiquement sur GitHub
Développement actif