The Journal of Open Source Software

robot_collision_checking: A Lightweight ROS 2 Interface to FCL (Flexible Collision Library)

Mark Zolotas [1*¶], **Philip Long** [2*], **and Taskin Padir** [1,3]

1 Northeastern University, USA (at the time of this work) **2** Atlantic Technological University, Ireland **3** Amazon Robotics, USA ¶ Corresponding author * These authors contributed equally.

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Chris Vernon

Reviewers:

- @Aravind-Sundararajan
- @cadojo

Submitted: 15 August 2024
Published: unpublished

Summary

This paper presents `robot_collision_checking`, a C++ library that creates an easy interface to the Flexible Collision Library (FCL) (Pan et al., 2012). The package allows users to access collision and distance checking functionalities of FCL directly through a Robot Operating System (ROS) interface (Quigley et al., 2009), given that the robotics community widely relies on ROS as the standard for software development. We include ROS 1 and ROS 2 (Macenski et al., 2022) implementations of the core C++ library.

Collisions and distances can be calculated between a variety of collision objects, including solid primitives (spheres, box, cylinder), planes, meshes, voxel grids, and octrees (via the Octomap library (Hornung et al., 2013)). Collision worlds that contain multiple collision objects can also be created and maintained. This enables collision and distance checking between single objects, as well as entire collision worlds. The `robot_collision_checking` package includes an example node that demonstrates how to create a collision world of ROS objects, use FCL functionality to perform collision-checking on these objects, and visualize the world in RViz (Kam et al., 2015).

The `robot_collision_checking` library is currently being used by the `constrained_manipulability`, developed by the same authors. Within the `constrained_manipulability` package there are more examples of using the `robot_collision_checking` library with URDF files and collision meshes in order to calculate collisions/distances between a robot and environmental objects, such as primitives and octomaps.

Statement of Need

Collision-checking is an increasingly important tool as robots are deployed into unstructured and dynamic environments, while ROS 1 and ROS 2 provide the most popular means of controlling robots for research applications. In the ROS ecosystem, one popular means of enabling collision-checking is via MoveIt (Coleman et al., 2014), a path planning and trajectory execution open-source software. The MoveIt collision-checking API can expose two different collision checkers: bullet and FCL. However, to leverage this functionality users have to install the entire MoveIt suite and either integrate their robot into MoveIt or ensure that their platform is already available to the software suite. Moreover, while MoveIt is an extremely sophisticated motion planning library, accessing the lower-level functionalities for purposes like collision and distance checking requires in-depth knowledge of the library's structure and hierarchy. Pinocchio (Carpentier et al., 2021) is another powerful robot modeling software that is also built upon FCL but suffers from the same overhead as MoveIt. The `robot_collision_checking` library aims to address the need for a lightweight alternative by providing a simple and transparent ROS interface to the FCL library.

Our package is similar to Python-fcl, which provides a Python binding of FCL that could also be used in a ROS architecture. The key difference is that our implementation is written in C++. The ros_collision_checking package also offers a collision-checking system for 2D vehicles in a ROS environment. Our collision-checking system instead extends the general capabilities of FCL for proximity querying any geometric model and can thus be applied in numerous robotics contexts.

Acknowledgements

Mark Zolotas is currently at Toyota Research Institute (TRI), Cambridge, MA, USA. This paper describes work performed at Northeastern University and is not associated with TRI.

Taskin Padir holds concurrent appointments as a Professor of Electrical and Computer Engineering at Northeastern University and as an Amazon Scholar. This paper describes work performed at Northeastern University and is not associated with Amazon.

References

Carpentier, J., Budhiraja, R., & Mansard, N. (2021, July). Proximal and Sparse Resolution of Constrained Dynamic Equations. *Robotics: Science and Systems 2021*. https://hal.inria.fr/hal-03271811

Coleman, D., Sucan, I., Chitta, S., & Correll, N. (2014). Reducing the barrier to entry of complex robotic software: A moveit! Case study. *Journal of Software Engineering for Robotics*, *5*(1), 3–16.

Hornung, A., Wurm, K. M., Bennewitz, M., Stachniss, C., & Burgard, W. (2013). OctoMap: An efficient probabilistic 3D mapping framework based on octrees. *Autonomous Robots*. https://doi.org/10.1007/s10514-012-9321-0

Kam, H. R., Lee, S.-H., Park, T., & Kim, C.-H. (2015). Rviz: A toolkit for real domain data visualization. *Telecommunication Systems*, *60*, 337–345.

Macenski, S., Foote, T., Gerkey, B., Lalancette, C., & Woodall, W. (2022). Robot operating system 2: Design, architecture, and uses in the wild. *Science Robotics*, *7*(66), eabm6074. https://doi.org/10.1126/scirobotics.abm6074

Pan, J., Chitta, S., & Manocha, D. (2012). FCL: A general purpose library for collision and proximity queries. *IEEE International Conference on Robotics and Automation*, 3859–3866. https://doi.org/10.1109/ICRA.2012.6225337

Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R., & Ng, A. (2009). ROS: An open-source robot operating system. *ICRA Workshop on Open Source Software*, *3*.