Robotran-Yarp interface: a framework for real-time controller development based on multibody dynamics simulation

Timothée Habra*, **Houman Dallali#**, **Alberto Cardellino#**, **Lorenzo Natale#**, **Nikolaos Tsagarakis#**, **Paul Fisette***, **Renaud Ronsse***

* Center for Research in Energy and Mechatronics
Université catholique de Louvain (UCL)
Place du Levant 2, 1348 Louvain-la-Neuve, Belgium
[timothee.habra, paul.fisette, renaud.ronsse]@uclouvain.be

Advanced Robotics & iCub Facility
Instituto Italiano di Tecnologia (IIT)
Via Morego 30, 16163 Genova, Italy
[houman.dallali, alberto.cardellino, lorenzo.natale, nikos.tsagarakis]@iit.it

Introduction

Simulation tools are widely used in testing and prototyping new technologies. By providing a safe and controllable testing environment, they allow fast and cheap code prototyping. Benefits of dynamics simulation are particularly evident for robotics controllers development. Indeed, robotic platforms are not always available for controller testing (long mechanical design, hardware repair, few platforms) and running untested controllers turns out to be dangerous (operator injuries, hardware damage).

Nevertheless, transferring a controller developed in simulation to a real robot is not straightforward. It typically requires to adapt the format of the input and output signals of the controller to the robot actuators and sensors. Not only time consuming and error prone, this coupling of the high-level controller with the robot hardware severely impacts the reusability and lifespan of the code. The code stays with a specific robotic platform and usually gets obsolete as soon as the hardware changes.

To tackle these problems, several middleware tools were developed within different robotics frameworks such as YARP, ROS and Orocos to name a few. By decoupling the controllers from the robot hardware, they encourage code reuse and collaboration across projects. A unique controller can work with different hardware (e.g. new joint encoders) as long as they are interfaced with the middleware. This approach allows to interface the middleware with a simulator that can be seen as just another hardware. Therefore, without changing a single line of code, the very same controller can work with the simulator and with the real robot.

This paper presents the coupling of the Robotran multibody dynamics simulator [1] with the YARP middleware [2]. The proposed framework allows fast controller development and eases collaborations on large scale projects. Robotran was selected as dynamics simulator for its speed and accuracy. To illustrate how this framework can be used with complex mechatronic system controllers, an example with the COMAN humanoid robot is given. The corresponding code is open source[1].

Robotran-Yarp interface

The key idea of YARP is to provide an abstraction layer between controllers and the robot (simulated or real), what we called a middleware above.
Practically, the controller receives inputs (i.e. sensor measurements) and sends outputs (i.e. control commands) through *interfaces*. Interfaces are abstract base classes (in C++) with a set of virtual methods to control actuators or read sensor data. For each device (actuator, sensor) interfaced with YARP, a *driver* should implement the corresponding interface classes (See Figure 1). Note that many device drivers can implement the same interface (e.g. different encoders).

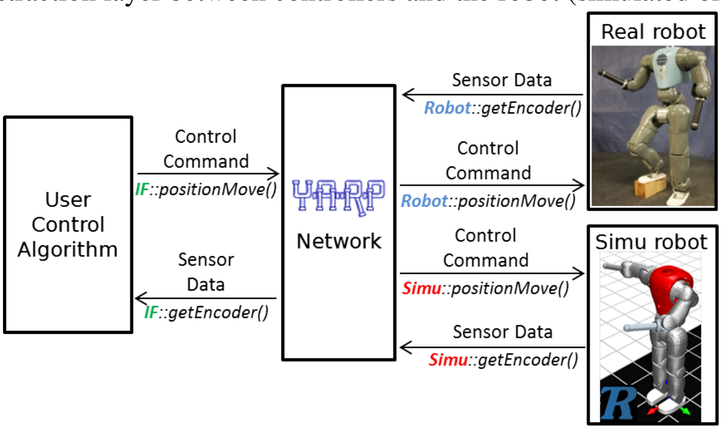

Figure 1: Robotran-Yarp control framework

[1] in public repositories https://gitlab.robotran.be/walkman/coman_robotran

Hence, as the controller is device agnostic, it can work with any platform implementing its interfaces. Robotran is particularly tailored to efficient multibody dynamics simulation. Therefore, the proposed interface aims at testing controllers relying on dynamics (e.g. locomotion, whole-body control). Consequently, the following drivers were implemented in Robotran environment:

- *ControlBoard driver*: translates commands received from the user control algorithm into commands specific for the controlled platform (with position, torque or joint impedance control). It also reads the encoders and joint torques. Typically, at each simulation time step, joint positions, velocities and torques are read and the desired joint positions or torques are sent.
- *Force Torque sensor driver*: gets the wrench measured by a force sensor located on a body. In Robotran, this corresponds to the torque and force measured by a force sensor.
- *IMU driver*: gets the Inertial Measurement Unit (IMU) data (velocity, orientation and gravitational forces) of a body.
- *Clock driver*: gets simulation time in order to synchronize the controllers with the simulation; this driver is specific to simulator drivers. Indeed, simulation time is not always constant and might run faster than real time.

In addition for each interface, YARP implements generic network proxy devices that allow remote execution of the same code; different controller pieces can run on different machines. Thus, controllers on a user laptop can be connected interchangeably to the robot or the simulator with no need to recompile.

Application

This common interface, allowing to run controllers on the robot or on the simulator, offers a wide range of applications.

As stated in the introduction, testing the controller on a simulated environment before the real one allows to do *fast code prototyping*. Furthermore, it is possible to write regression tests that directly interface with the simulator to verify that patches or new development do not modify the expected behavior of robot software.

On top of that, the simulation can also run in parallel to the real robot (See Figure 2). This is particularly useful to perform *state estimation* and *prediction*. Among others, we plan to use this tool to support teleoperation with low bandwidth and noisy communication.

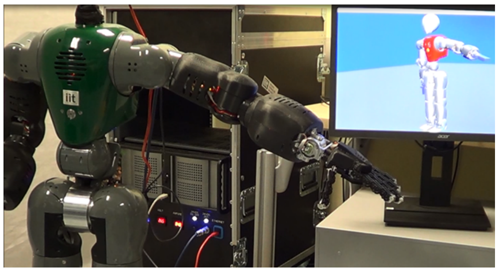

Figure 2: Real and simulated COMAN humanoid controlled by a unique controller

Finally, the simulation can run after the real test as a post-process for further *analysis* and *failure diagnostic* (thanks to YARP data recording features).

Future work will consist in bringing the Robotran simulator and the interface plugin in a generic library. It will further be extended to other platforms, like e.g. the iCub and WALKMAN humanoid robots.

Acknowledgments

This work was supported by the European Community's 7th Framework Programme (FP7-ICT-2013-10) under Grant 611832 (WALKMAN collaborative project). We are grateful to Alexandra Zobova and Nicolas Van der Noot for their support.

References

[1] N. Docquier, A. Poncelet and P. Fisette. ROBOTRAN: a powerful symbolic generator of multibody models. Mechanical Sciences, Vol. 4, pp. 199–219, 2013.

[2] G. Metta, P. Fitzpatrick, L. Natale. Yarp: Yet another robot platform. International Journal of Advanced Robotics Systems, special issue on Software Development and Integration in Robotics, Vol. 3, No. 1, pp. 43–48, 2006.