Haptic coupling with the PR2 as a demo of the OROCOS - ROS - Blender integration

Koen Buys, Steven Bellens, Nick Vanthienen, Wilm Decre, Markus Klotzbücher, Tinne De Laet, Ruben Smits, Herman Bruyninckx, and Joris De Schutter

Keywords: *software integration, mobile manipulation, haptic feedback, LWR, PR2, Orocos, ROS, Blender*

I. Introduction

Modern and future robotic applications will have to integrate many software components. Inevitably, the components will be provided by different vendors or research institutes, and will not be designed together, and in many cases also not implemented from the same source code repository. This paper presents a discussion on how we deal with this multi-component, multi-vendor situation in a haptic robot setup. This situation includes the least optimal case when most or all of the components are "legacy", in the sense that they have to be integrated as binary pieces of code, without the possibility to change the components for the sole purpose of facilitating system-level integration. The presented approach is illustrated by an application that integrates (i) two KUKA Light-Weight Robot FRI control components, (ii) a set of ROS components to control a Willow Garage PR2 robot, (iii) a Blender component for on-line visualization, and (iv) a set of Orocos/RTT components that take care of the data flow communication and some simple coordination between all components. This application gathers a large number of aspects of our PR2 Beta Program of the past year.

Many robotics groups (academic as well as industrial) are investigating how to make robotic systems by integrating an increasing number of functional components (hardware as well as software). The state of the art in hardware integration has, since quite some time already, reached a state in which it is not too difficult to make a system out of "common of the shelf" components, not in the least because of the rather high level of standardization of mechanical, power and communication interfaces of such hardware components. However, the state of the art at the software level is much less advanced.

Widely accepted standards for integrate-able component design and interfaces do not yet exist, and there is, even surprisingly very little demand for such standardization from providers of software components. What *did* boom in the software domain during the last decade is the availability of a large number of "software frameworks", mostly with liberal "open source" software (e.g., Orocos [4], Player [6], ROS [9], [11], OpenRTM [8].) but also with significant proprietary parts (e.g., URBI [7], Webots [5]). The current development of these frameworks is almost exclusively focused on extending one's own framework by adding more and more functionality to it, with close to none coordinated efforts for functionality sharing; the Gearbox project [2] is a notable (but not so popular) exception.

In this paper and demonstration we show how we couple a KUKA LWR to the Willow Garage PR2 to allow remote "haptic" control of the PR2 robot. Although, currently, it might be a bit of an overstatement saying that this combination of hardware is "common off the shelf", the authors do believe that more and more research groups, industries, and (professional) users will have access to similar systems in the coming decade, *and* will want to couple some of them in a "haptic" set-up as presented in this paper. The authors also realize that using the term "haptic" in the context of the presented work has a very different meaning (performance-wise) than in the dedicated research on haptic interfaces; nevertheless, the concepts (and the subjective feeling of the human operators involved in the experiments) do have a significant haptic component, in the strict sense of the word, namely to get on-line motion/force feedback from the remote system to which the human operator can react without being disturbed too much by performance issues to make the task unbearable or ineffective.

The presented teleoperation application is furthermore an ideal example of the integration of two component-based robotic frameworks namely OROCOS (Open Robot Control Software) [4] and ROS (Robot Operating System), together with the Blender visualisation environment and thus illustrating 2 of our focus points of the PR2 Beta Program.

II. OROCOS - ROS integration

The demonstration is an excellent showcase for our work on the integration of the Orocos framework and the ROS system. This software is developed as part of the PR2 beta program. More in-depth information can be found in [10]. Installation and usage instructions are available

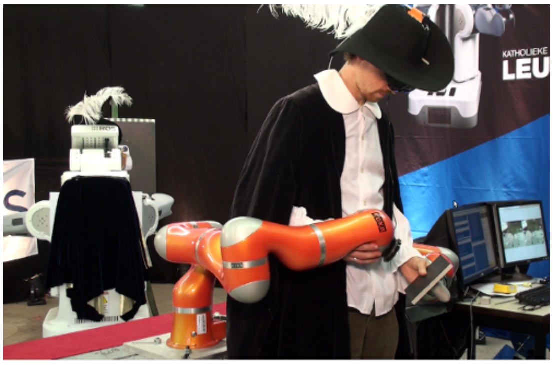

Fig. 1. Illustration of the original setup. In the front the human operator is operating the two KUKA-LWR arms, is wearing VR goggles (displaying the images of the eye cameras of the PR2), and IMU sensor on the head (to couple his head motion to the one of the PR2). In the back the PR2 is used as a slave to perform an "artistic" painting task using the haptic coupling between the KUKA-LWR arms and the PR2.

at `http://www.ros.org/wiki/orocos_toolchain_ros`. We give a live overview of the available features:

- Data flow: Orocos components are equipped with data ports over which they exchange data with other components and processes. In ROS, this is equivalent to nodes exchanging data over topics. By using the integration software, Orocos components can publish data to ROS topics and vice versa;
- Properties: Orocos components are equipped with property variables, whereas ROS nodes have parameters. Orocos components can now read the values of property variables from the ROS parameter server;
- Deployment: Orocos components are deployed in combination with ROS nodes using a single launch script; and
- Generation and Building: scripts and macros are available to generate and build Orocos components similar to ROS packages.

III. Blender interface

A growing number of robotics labs cooperate in the "Blender for Robotics" project [3], to integrate the *open-source* computer animation program Blender into their research, in the first place as a visualisation tool, but gradually more and more as a full-fledged component in a robot simulation and programming environment. The robotic research group in Leuven focuses on improving Blender support for the task specification and control of complex, sensor-based robot tasks. We also illustrate the visualisation of complex robot systems (like the PR2, figure 2) connected over different types of middleware (Yarp, ROS, Corba, ...). Our current efforts include integrating our results in the MORSE Open Robot Simulator project [1].

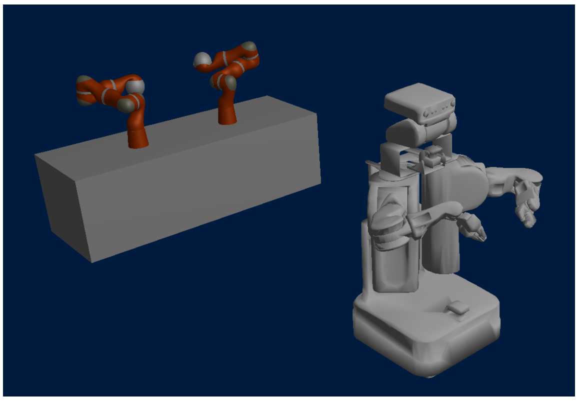

Fig. 2. Illustration of the simulated setup in Blender.

IV. Demonstration and interactive aspects

In the original paper we presented a teleoperation setup including haptic coupling between two two-arm robot systems (see Figure 1) to realize an "artistic" painting task; the goal is to control remotely the arms of a PR2 (the slave side) using two KUKA Light-Weight Robots (LWR) (the master side). The haptic artistic painting task uses the Orocos-ROS integration (Section II) and Blender as a visualizator and simulator (Section III), hereby serving as a showcase for our developments concerning the Orocos-ROS-Blender software integration.

In the demonstration, accompanying this paper, people get a hands-on experience demonstrating the three main features of the application.

Firstly, the haptic coupling between the PR2 robot arms and the KUKA youBot is shown. People can manipulate the PR2 arms and observe the youBot reaction or vice versa. The aim here is not to illustrate the best performant haptic coupling setup, but rather illustrating modern tools combining mechanically very different platforms.

Secondly, by means of the wearable display, we provide them visual feedback of the PR2 eye cameras.

Thirdly, they can wear a hat, with integrated IMU sensor, which couples the PR2 head movement to that of the human. This head movement coupling, together with the visual feedback, improves the human identifying himself as the PR2.

On the software level, we give the audience an impression of how the software architecture looks like by explaining what different components exist, how they communicate with each other, etc . . .. They can see how tightly both the OROCOS and ROS framework are integrated and how easy it is to use both frameworks together.

Furthermore, we will compare the existing implementation using the KUKA lightweight arms with the implementation using the youBot, thereby demonstrating how one can substitute any robot in the system with minimal effort.

References

[1] MORSE, the OpenRobot simulator. http://www.openrobots.org/wiki/morse.

[2] G. Biggs and A. Makarenko. The GearBox project. http://gearbox.sourceforge.net/.

[3] H. Bruyninckx. Blender for robotics. http://wiki.blender.org/index.php/Robotics:Index. Accessed online 20 June 2010.

[4] H. Bruyninckx. Open RObot COntrol Software. http://www.orocos.org/, 2001. Last visited June 2011.

[5] Cyberbotics. Webots. http://www.cyberbotics.com/.

[6] B. Gerkey, R. Vaughan, A. Howard, and N. Koenig. The Player/Stage project. http://playerstage.sourceforge.net/, 2001.

[7] Gostai. Urbi. http://gostai.com/, 2007.

[8] National Institute of Advanced Industrial Science and Technology, Intelligent Systems Research Institute. OpenRTM-AIst. http://www.openrtm.org. Last visited November 2010.

[9] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. B. Foote, J. Leibs, R. Wheeler, and A. Y. Ng. ROS: an open-source Robot Operating System. In *ICRA Workshop on Open Source Software*, 2009.

[10] R. Smits and H. Bruyninckx. Composition of complex robot applications via data flow integration. In *Int. Conf. Robotics and Automation*, pages 5576–5580, Shangai, China, 2011.

[11] Willow Garage. Robot Operating System (ROS). http://www.ros.org, 2008. Last visited 2011.