Robot Web Tools

By Brandon Alexander, Kaijen Hsiao, Chad Jenkins, Bener Suay, and Russell Toris

The robot operating system (ROS) has provided roboticists with a robust, cross-platform middleware, and has also given researchers from around the world a common place to share their ideas and reproduce each other's work. Within the growing ROS community, developers have created software on a wide range of robotic platforms, from Willow Garage's sophisticated PR2 mobile manipulator to LEGO's simple and widely available NXT.

However, while the community's efforts have facilitated significant milestones in robotics research, the core ROS middleware still requires a considerable learning curve, including a general understanding of UNIX systems and languages such as C++, Python, or Java. To generate interest in robotics in a larger, more general population, we must remove this requirement. The World Wide Web provides both a guiding example and a path for broadening the reach and accessibility of robotics. With the goal of building a larger community of robot Web app developers, we describe recent efforts to expose the functionality of ROS via common Web development tools such as JavaScript.

Rosbridge as an Access Point for ROS

Rosbridge provides an application-layer network protocol for robotics, allowing arbitrary, non-ROS client processes (including Web interfaces) to interoperate with ROS processes. Specifically, rosbridge allows clients to publish and subscribe to ROS topic messages, and to invoke ROS services in the server's runtime environment, by transporting JSON-formatted messages over TCP sockets and WebSockets.

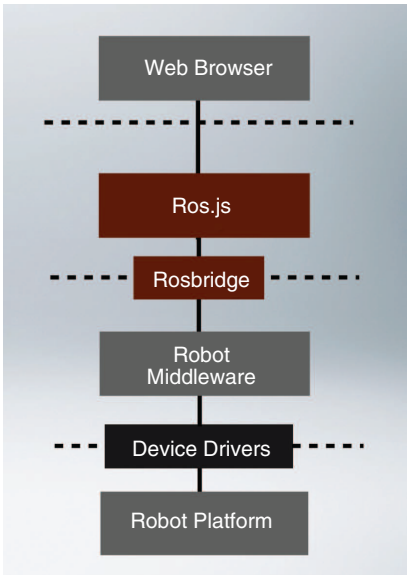

Figure 1. Pipeline for a typical Web application using rosbridge.

The beauty of rosbridge is that its clients are language-independent. That is, rosbridge clients can be written in any language that supports WebSockets. Furthermore, rosbridge itself does not limit clients to ROS. The reference server implementation mentioned above is written for ROS, but the rosbridge protocol is a specification and, as such, is not tied to any programming language or runtime. The JSON-based rosbridge protocol has been designed to enable data publishing, subscribing, and service invocation between any combination of clients and servers, regardless of platform. Implementations have been successfully created using Linux, Windows, Android, iOS, Arduino, common Web browsers, and more.

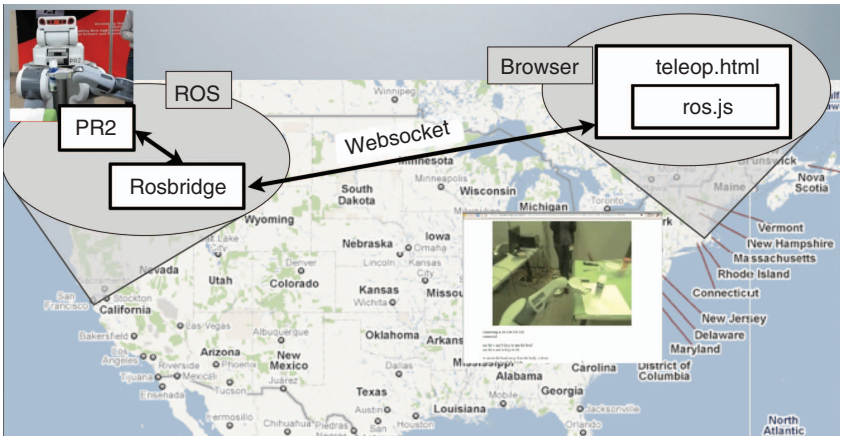

Figure 2. Connection diagram for a PR2 teleop application.

Digital Object Identifier 10.1109/MRA.2012.2221235

Date of publication: 6 December 2012

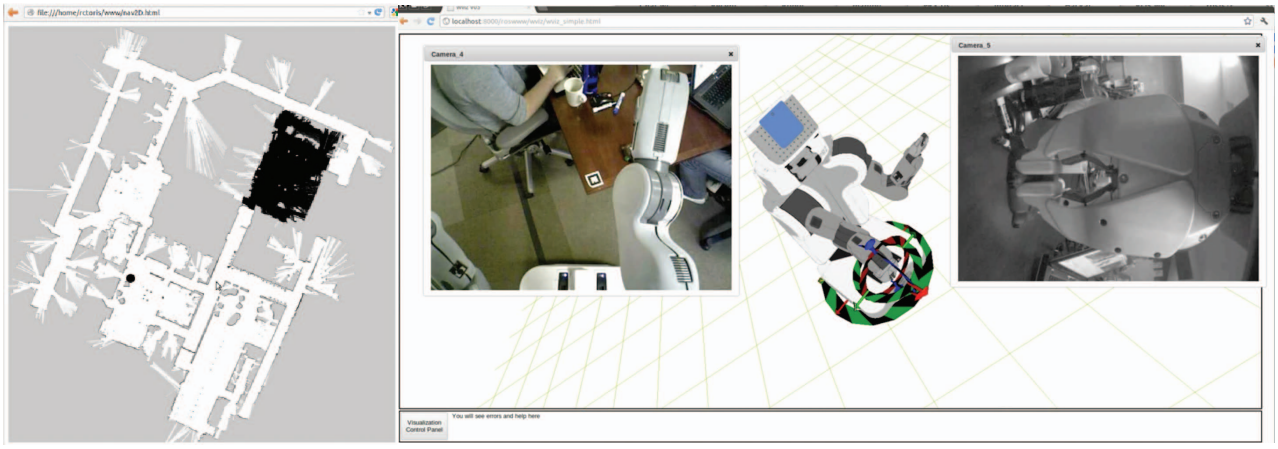

Figure 3. Scavenger hunt Interface 1: wviz-based Web interface that provides autonomous map navigation, camera streaming, and interactive marker tools for moving the arms, base, torso, and head.

A stable and robust implementation of a ROS rosbridge server (as well as a number of useful Web tools) is available in the rosbridge_suite stack on ros.org (http://www.ros.org/wiki/rosbridge_suite), and a specification of the rosbridge protocol is available at http://rosbridge.org.

Ros.js

For integration with modern Web tools, a JavaScript library named ros.js was built to facilitate communication between the browser and rosbridge. A JavaScript Web application running in the browser can communicate with a ROS application running on a remote robot or server using ros.js and rosbridge.

Ros.js is designed to be lightweight and event-based. Its lightweight code base allows easier integration within existing, large-scale JavaScript applications, in addition to allowing other robot Web tools to build off of it. The event-based library will emit an event on the basis of feedback from the robot, server, or user. An event-based ros.js allows for a more responsive UI and decouples the ros.js module from other JavaScript modules (Figures 1–2).

Wviz

Wviz (short for Web visualization) is a robot and sensor data visualization tool that runs in a Web browser. Similar to RViz, it renders 3-D models of the robot and sensor data, lets the user add or remove different sensor data displays, and modifies the properties of each display (e.g., image size, color, topic, TF, etc.).

Although wviz is designed to be generic, creating a customized version of wviz is fairly straightforward: necessary widgets and displays (such as image, robot model, and interactive markers) can be called from the main HTML file and loaded with the application, as opposed to being added dynamically by the user. Wviz and related packages can be found in the bosch_web_visualization stack.

Robot Web Hackathon: PR2 Scavenger Hunt

During the week of 13 August 2012, a Robot Web Hackathon was held at Willow Garage to gather people from the ROS community interested in creating and using robot Web tools. Using Web teleoperation of a Scavenger Hunt task as a motivating example, the event's focus was to create and test a set of community tools for creating robot Web applications. Attendees included faculty, students, and researchers from Brown University, WPI, Georgia Tech, Bosch LLC, and Willow Garage, primarily drawn from the current rosbridge user community.

The goals for the week included improving and testing a new version of rosbridge, creating a commonly-agreed-upon version of ros.js, incorporating both into wviz and other existing robot Web applications being developed by the participants, and, finally, using the results to create Web interfaces for teleoperating PR2 robots in a scavenger hunt. Two interfaces were created to allow participants to locate, pick up, and photograph a variety of scavenger hunt objects. These interfaces were used successfully by several on-site participants, as well as a remote user from across the country.

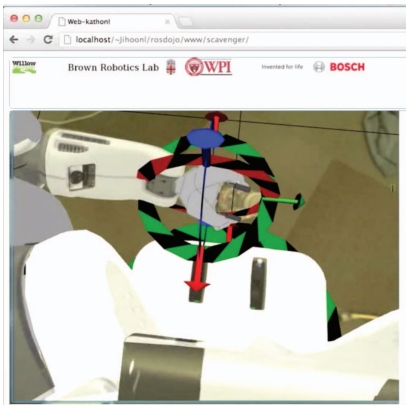

Figure 4. Scavenger hunt interface 2: interactive marker tools for moving the robot overlaid over the robot's camera view.

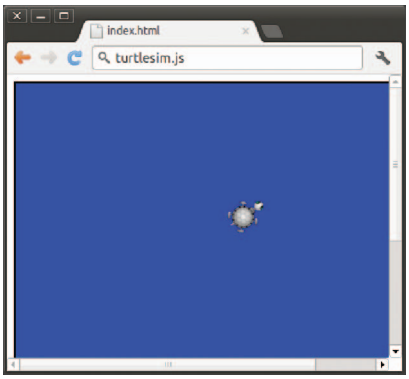

Figure 5. Tutorial interface for teleoperating a simulated turtle.

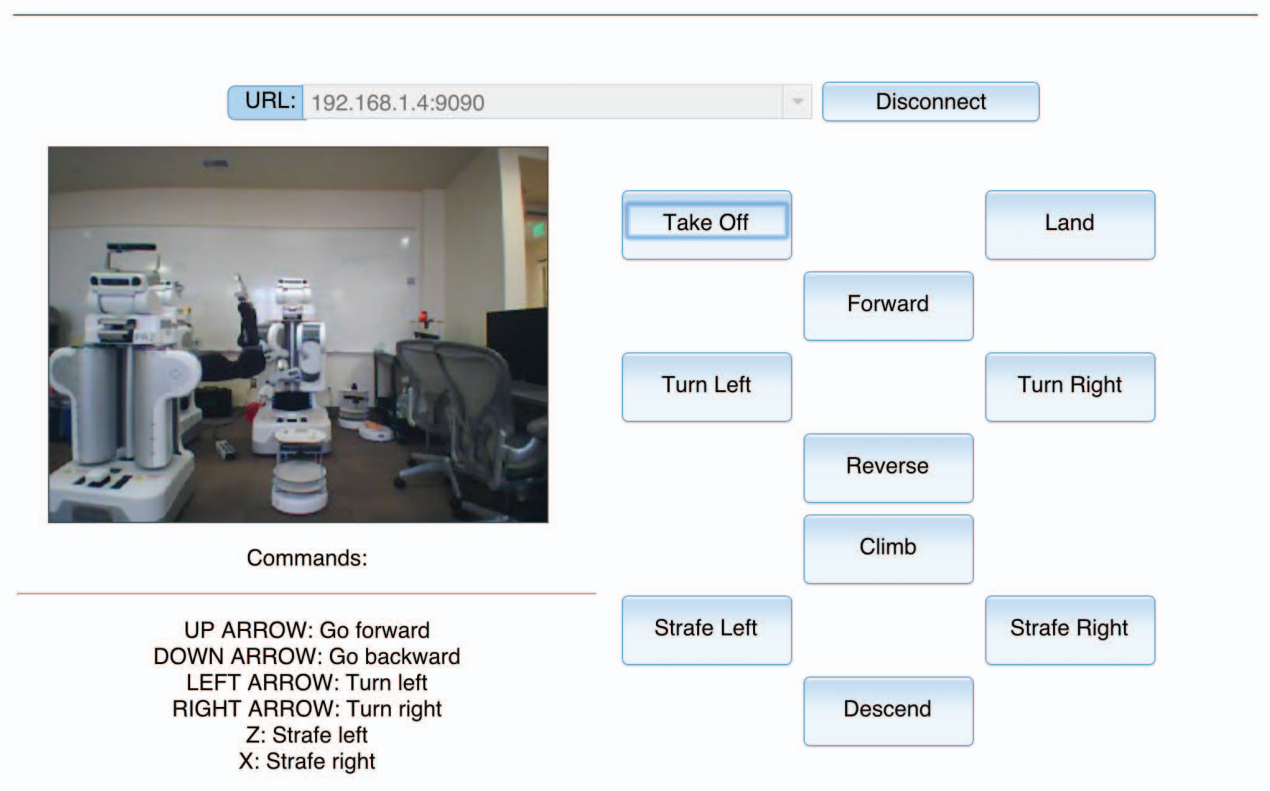

Figure 6. Tutorial interface for teleoperating a simulated turtle. Left: Tutorial interface for teleoperating an AR Drone.

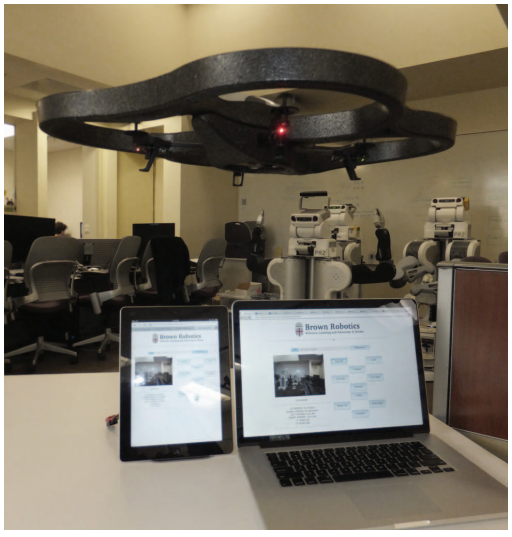

Figure 7. An AR Drone being teleoperated.

Both interfaces used the PR2 Interactive Manipulation pipeline (described in a previous ROS Topics article) as the robot backend. A Web-based implementation of interactive marker clients allowed for similar functionality in the browser as the interactive marker tools used in RViz. One of the two interfaces was made using wviz, and was customized to provide streaming camera images from the head and forearm as well as a rendered robot model with interactive markers upon launch. Additional widgets allowed the user to ask the robot to autonomously navigate using a map, and to take a camera snapshot of found objects (Figures 3 and 4).

RobotWebTools.org

To organize the available open-source, BSD-licensed tools for robot Web applications, including those described above, and to provide support and community, a new portal wiki—Robot Web Tools—has been created at robotwebtools.org. Robot Web Tools is designed to enable Web developers, roboticists, and even students to start building a robot Web application quickly. Thus, the walkthroughs cater to all abilities, from the novice to the advanced user. Currently available tutorials include interfaces for teleoperating a simulated turtle (Figure 5) as well as a physical AR Drone (quadrotor) (Figures 6 and 7).

In addition to basic tutorials, the portal provides information on a variety of tools, libraries, and sample applications, from low-level JavaScript modules such as ros.js or 2dmap.js to full-featured robot Web applications like the Robot Management System (a Web application test platform for Human-Robot Interaction experiments). Many of the tools and projects are open sourced under the GitHub organization github.com/robotwebtools.

For those interested in joining the community of robot Web developers, information on how to join the Google Group and ongoing weekly calls is available at robotwebtools.org. And, if you create a robot Web application or tool that you would like to share, please post information about it on the Robot Web Tools wiki!