Design • Build • Program • Learn • Compete

SERVO MAGAZINE

FOR THE ROBOT INNOVATOR

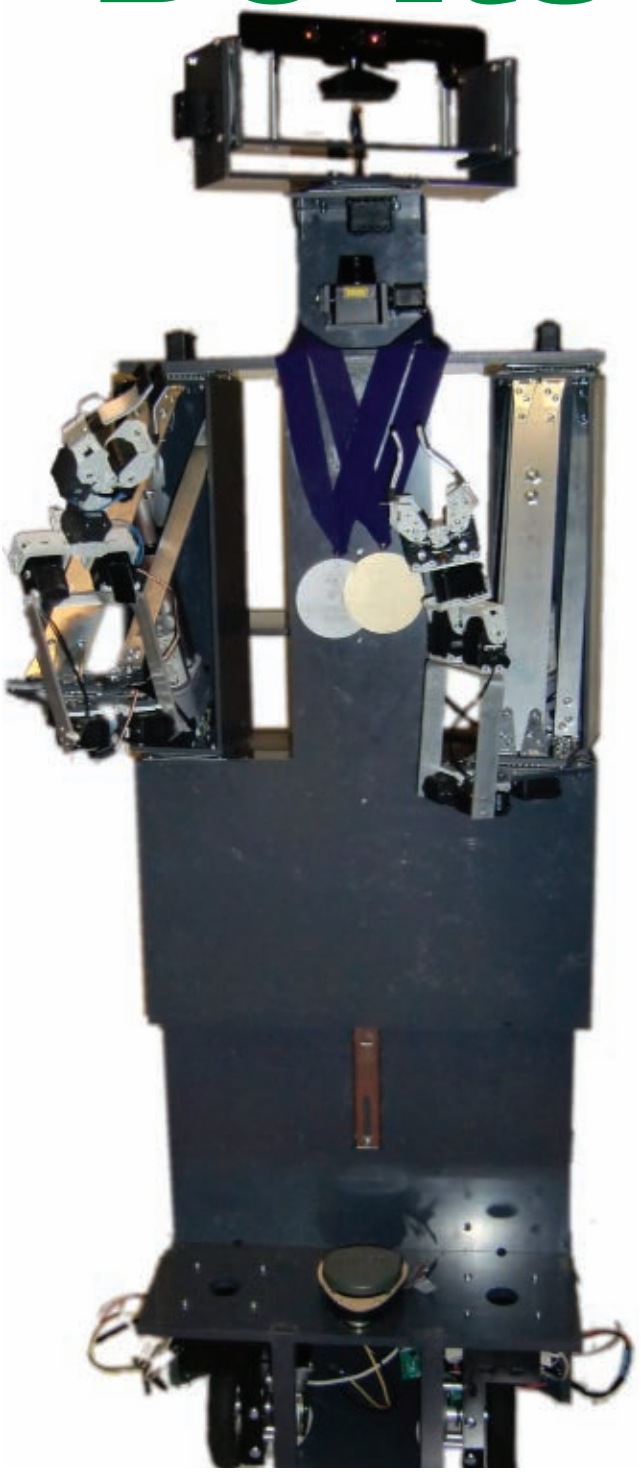
www.servomagazine.com

August 2013

Making Better Arduino Robots With The ARDBOT II

- **PR2Lite Grows Up**
  The latest evolution of this homebrewed PR2.
- **It's All In Your Head**
  How scientists are looking to the anatomy of the human body to make smarter and faster machines.

PR2Lite Grows Up — and So Do Its Makers



by Alan Downing, Matthew Downing, and Frank Ou

You can discuss this topic at **http://forum.servomagazine.com**.

PRLite was first featured on the cover of the December 2011 *SERVO Magazine* (**Figure 1**). In that article, we described how four high school students and their mentor fathers prototyped their own PR2. PRLite had its own tilting LiDAR, a Kinect, a 12" torso lift, two Bioloid arms, a Neato LiDAR on its base, four partially pivoting Parallax motors and wheels, and more. However, PRLite had a short wooden frame with arms positioned at the bottom of the front torso. It didn't look much like Willow Garage's PR2.

Much has changed over the last two years. First, three of the builders are now in universities and make only occasional appearances at our weekend meetings. Similarly, our robot has grown up to look and act more like a PR2. Now made of machine-cut PVC, PR2Lite is between 40" and 52" high at the shoulders, and has LiDARs and a Kinect placed similarly to their PR2 counterparts. Plus, PR2Lite sports new upper arms powered by linear actuators, a new CHR-UM6 IMU, and a lot of new ROS-compatible software.

Our PR2Lite team now mostly consists of Alan (software), Frank (hardware, networking, firmware), and Matthew (builder, designer, high school robot club president-elect). **Figure 2** contrasts the old wooden prototype with the revised PR2Lite.

Design of the Body

Matthew designed and built PR2Lite's body. He used Autodesk Inventor to model the pieces which were then machine-cut from quarter and half inch PVC by Mr. Plastics in Oakland, CA. Like PR2, our robot is the appropriate height to manipulate objects on tables and do other human interactions. The Autodesk Inventor screenshots show the design of the arms and upper torso when lowered and raised **(Figure 3)**.

The upper arms are lifted by 4" linear actuators and rotated at the shoulders by two AX-12 servos. The linear actuators are

kept close to the body and ROS-compatible software converts the linear length to angle, and vice versa for position feedback. The upper arms use a parallelogram design to allow the lower arm to remain on a horizontal plane **(Figure 3)**.

Four inch wide lazy susans on top and below the compartment for the linear actuator make it easier for the servos to pan the arms. The lower arms are currently similar to AX-12 CrustCrawler arms with dual grippers.

The base is 9" high — the same as PR2's base — and can hold up to five large 12-volt sealed lead-acid batteries. The wheel design and software was derived directly from our original PRLite prototype. The computer and electronics are largely contained throughout the torso.

FIGURE 1. The December 2011 *SERVO Magazine* with a picture of the first PRLite prototype beside Willow Garage's PR2.

FIGURE 2. The first prototype (left) beside the new and improved PVC PR2Lite (right).

The Autodesk design was then manually converted into a URDF model for ROS to do the different transforms, display, and planning for PR2Lite. The passive cyclic joints are a problem for ROS' URDF which requires an acyclic tree-structure.

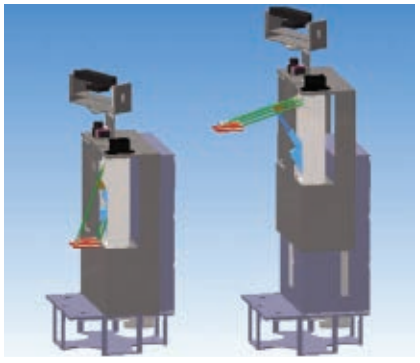


FIGURE 3. The robot has a telescopic body, lifting up by one foot. Also notice the parallelogram shape of the upper arms (green) which keeps the elbows (red) horizontal — even when the upper arms lift up.

For example, PR2Lite's panning shoulder, the arm, and the linear actuator form a triangle with two passive joints.

The linear actuator is along the hypotenuse and raises or lowers the arm when the actuator is extended/retracted. In our URDF, we artificially break the cycle by fooling ROS into thinking that the complex shoulder joints are just another Dynamixel joint.

Basic trigonometry translates the linear length into an angle of rotation for the upper arm that is returned in a Dynamixel joint state message.

The URDF is used by ROS to do transformations between different frames and joints, allowing general control of robots with many degrees of freedom. An output of *rosrun tf view_frames* gives an idea of the scale of PR2Lite **(Figure 4)**.

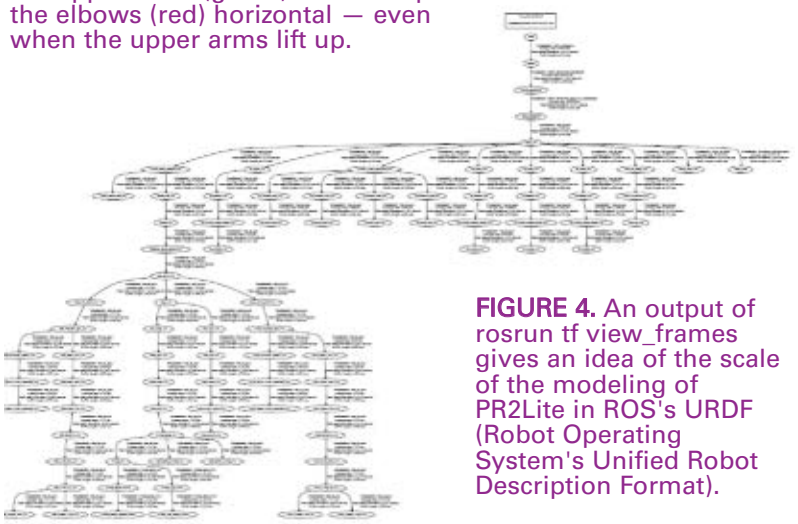


FIGURE 4. An output of rosrun tf view_frames gives an idea of the scale of the modeling of PR2Lite in ROS's URDF (Robot Operating System's Unified Robot Description Format).

New Linear Actuator and Wheel Controllers

Before leaving for Berkeley, Robert redesigned PR2Lite's controllers for the linear actuators and wheels. His firmware implemented a protocol stack over RS-485 that broadcasted datagrams or stream-based connections with packet collision avoidance and retry capability. The RS-485 provides much improved electrical noise immunity compared to the I^2C bus in the old PRLite.

The new design also uses MOSFETs to replace the old relays. The wheel controllers utilize photogates to act as encoders and provide PID control. Unavailable at the time, Parallax has now changed the design to their wheel controllers to also support this functionality.

Although the microcontrollers on the RS-485 bus can try to avoid packet collisions, it is hard for the PC with the USB serial port to do so. Frank has changed the software to give priority to the

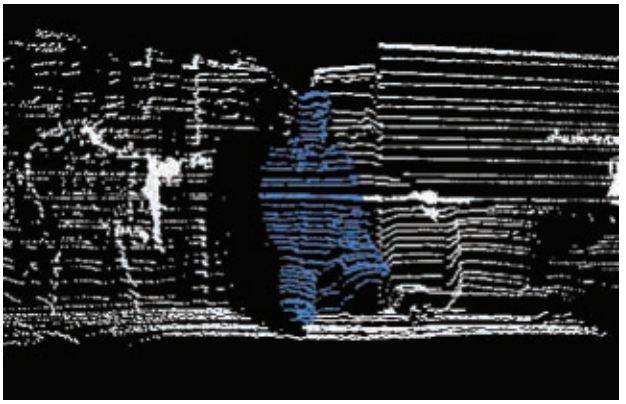


FIGURE 5. A kneeling person seen in a PointCloud from PR2Lite's tilting LiDAR. The person has been highlighted in blue in post-editing to be more obvious in the picture, but is actually very easy to see in the 3D model created by the PointCloud.

PC to transmit packages over the RS-485 bus. All the microcontrollers will withhold from transmitting packages when the PC is talking.

This method reduces packet collisions between microcontrollers and the PC, and still allows the microcontrollers to send out frequent position updates from the linear actuators and wheels.

The four wheels of the PR2Lite are driven by four motors with four wheel controllers, and must start and stop at the same time. A cable connecting the four wheel controllers was added to function as a synchronization signaling bus to ensure that all four wheels will move at the same time.

For example, if for any reason one wheel controller fails to receive a move command from the PC, the other three wheel controllers will not start. By monitoring the status of all the wheel controllers, the PC can decide to retransmit the move command or to cancel the move command. This mechanism ensures that PR2Lite will not crash due to communication errors with the wheels.

Tilting LiDAR

PR2Lite has a Hokuyo URG laser distance center that can be tilted by a Dynamixel AX-12+ servo. Like PR2's tilting LiDAR, ours is located on the neck below the 3D camera. We modified the University of Arizona (UofA) Wubble2 code to set tilting speeds separately and to publish transforms. The UA demo program also assembles scans and publishes a *point_cloud*. The current assembler doesn't self-filter or shadow-filter.

Eventually, we'll convert the PointCloud to PointCloud2, and then use the PCL library. In addition, the LiDAR will also be statically tilted for obstacle avoidance in SLAM. **Figure 5** shows the LiDAR's PointCloud of a chair and kneeling person.

Neato LiDAR

We disassembled a used Neato and extracted its motherboard and LiDAR. The Neato LiDAR is the main

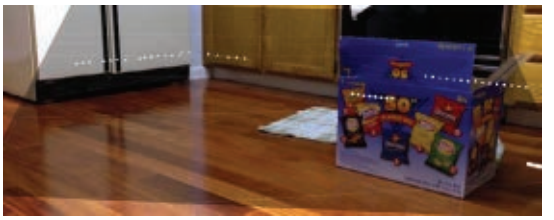


FIGURE 6. Notice the points that the Neato sees superimposed over the kitchen it was looking at.

sensor for SLAM. Like with PR2, the LiDAR was positioned horizontally on the base of PR2Lite, and the motherboard was placed on a shelf within the torso. We replaced the original motherboard's instantaneous "ON" switch with an LED so that the LiDAR will turn on whenever the ROS node is started.

We kept the whole motherboard for ease of development. At some time in the future, we may also use the Neato's sonar and edge detection sensors. The batteries must be attached for the motherboard to boot, even though we have the AC recharger permanently attached to the motherboard. The AC recharger can be powered by a battery through a DC to AC converter.

The Neato is also used by SLAM. PR2Lite's SLAM has been configured to use a new CH Robotics IMU and the latest revised PID. The CH Robotics IMU was chosen because it already has an ROS-compatible driver and firmware.

Figure 6 is picture of our kitchen with the LiDAR output superimposed. Soon, similar low-end LiDARs are expected to be for sale — without requiring purchase of a used vacuum.

Arm Navigation

Our work on the PR2Lite arm navigation is built upon the non-PR2 ROS pioneers, specifically Pi Robot (Patrick Goebel), Maxwell (Michael Ferguson), Turtlebot (Willow Garage), and Wubble (Anton Rebgun) — all of whom configured ROS to handle Dynamixel-based arms. Unfortunately, there was no perfect precedence for us:

- The Turtlebot and Maxwell use the simple arm driver with Arbotix.
- Wubble uses OpenRave for the trajectory planning and wrote a custom *wubble_follow_joint_trajectory*.
- Pi Robot uses OpenRave for the trajectory planning with USB2Dynamixel.

The issues that are different for PR2Lite include:

- A passive joint formed by a parallelogram for the upper arm that kept the elbow pan joint level with the ground as it is raised or lowered. ROS does not have built-in support for URDFs with joints that are not acyclic.
- Having two complex arms.
- Having a linear actuator to lift the upper arm. The linear actuator is part of the closed-loop kinematic joint chain not supported by ROS Electric arm navigation.
- Using the ROS Electric arm trajectory planning code, along with a USB2Dynamixel.

We updated PR2's Arm Kinematics plug-in to hard-code our passive elbow joints based on the shoulder angle of the upper arm. Simple changes were made so that the IK seed state input satisfies our parallelogram constraint.

Once we got the parallelogram arms supported, the Arm Navigation Warehouse tutorial worked great in simulation. Getting the arm navigation to work on the real robot was much harder. Our configuration to execute arm navigation on a real robot was derived from the following:

- The Dynamixel controller by Anton Rebgun (UofA) which evolved from the simple arm controller.
- The Dynamixel joint state publisher from Pi Robot which maps the Dynamixel controller to the joint names, and then puts them into an array of *joint_states* as required by the *joint_states* message. To make coding easier, we implemented a naming convention for the joints, controllers, and left/right sides.
- The Arbotix *follow_joint_trajectory*. This is the key missing piece that maps the ROS electric arm navigation to the Dynamixel state/commands. The baud rate and read/write characteristics for the USB2Dynamixel have to be set correctly. To satisfy the precise arm movements required for chess, for example, we extensively rewrote this node for PR2Lite's configuration to check for joint status to ensure precise following of trajectories while compensating for motor stalls.
- The *fake_pos.py* from Turtlebot. The warehouse viewer will hang if the non-fixed joints in the URDF are not sending out fake positions. To get the list of missing joints, use *rxconsole* and *rxloggerlevel* to obtain the DEBUG messages for the environment server.
- A transform publisher to send out the quaternion for the robot.
- The launch file for the ROS planning scene warehouse visualizer required not using *sim_time* or *fake_time*; setting *use_monitor* to true; setting *execute_left_trajectory*; and setting *use_robot_data* to true.

Each of the above requires their own configuration that needs to be consistent with all of the others. If one thing is not perfect, the ROS warehouse viewer will hang and require debugging using *rxgraph*, *rxconsole*, *rxloggerlevel*, *roswtf*, the *tf* tools, etc. It's not easy and there is a learning curve.

Once the arm planning worked in the visualizer, for the real robot, the arm planning was still not effective enough for PR2Lite to play chess. The generic arm planner in *arm_navigation* generally failed to find a valid plan due to a lack of a good analytic solver.

For chess, we had to make many incremental plans using the numeric solver based on current or projected position. The wrist points the grippers straight down to make the solutions simpler but, if necessary, it can rotate 360 degrees while still pointing downwards.

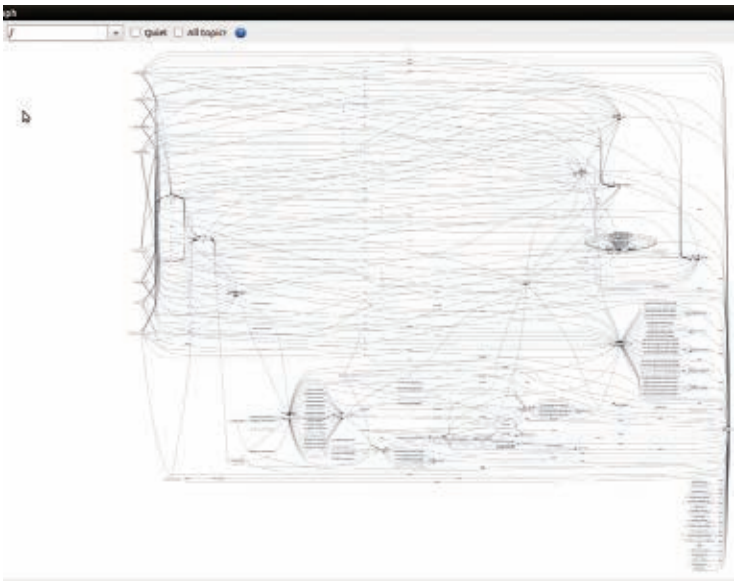


FIGURE 7. You won't be able to read the graph, but it gives you an idea of the complexity of the configuration of the arm navigation.

MoveIt! is the next generation arm navigation software for ROS. It has been recently released and solves many of these final issues, but this early release of MoveIt! introduces new ones. It may be time for us to upgrade. Up to now, we have deferred the upgrade from Electric to Groovy to avoid the very different build infrastructure and incompatible APIs.

The *rxgraph* for PR2Lite for both arm navigation and the base is shown in the graph in **Figure 7** where each of the nodes is a different package (written in C++ or Python) with messages being passed between them. You won't be able to read the graph, but it gives you an idea of the complexity of the configuration.

GUI

We modified a GUI by Patrick Goebel for Pi Robot that can be used both by tablets and laptops. **Figure 8** shows a sample screenshot. Further enhancements are planned.

References

http://mattdowning.wordpress.com/pr2-lite/

www.servomagazine.com/uploads/issue_downloads/PR-Lite.pdf

https://github.com/rqou/prlite-pc

www.willowgarage.com/velo2g?

www.pirobot.org/

www.ros.org/wiki/maxwell?

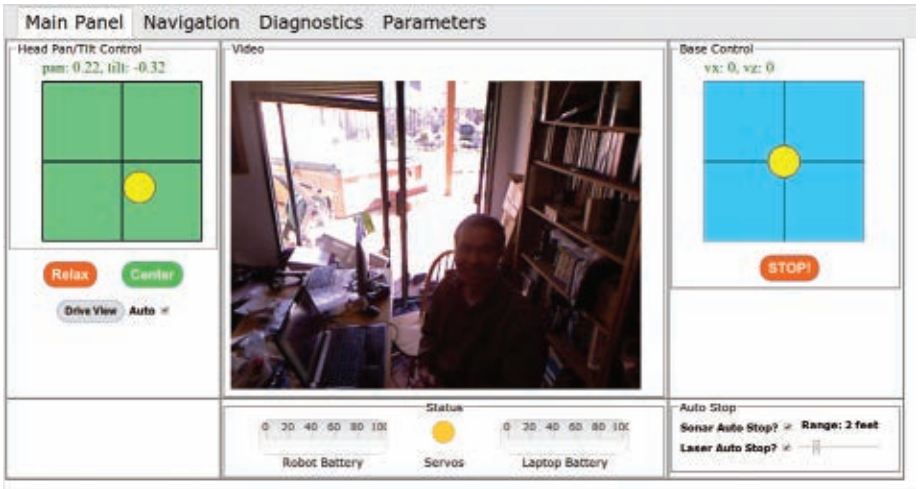


FIGURE 8. A screenshot of the GUI, meant to be compatible with tablets and laptops. Thanks Patrick Goebel (Pi Robot)!

by dropping the king off the table. Another time, PR2Lite did its best to swipe off all its pieces due to the servos beginning to overheat. This showed that although the AX-12s used for the arms were sufficient for simple teleoperation, they were underpowered for the lengthy arm planning and the precision movements required for chess.

For Maker Faire, we also constructed a new transportation base. PR2Lite is large and heavy, and requires strong supports to prevent it from tipping or jostling in transport. The new transportation base can be strapped to the loops that secure the removable seats of a minivan.

PR2Lite's head needs to be detached to fit through the side door. A hydraulic motor cycle lift is used to safely raise it to the right height for the car door. In contrast, Michael's Maxwell can be easily disassembled and packed into a small suitcase.

Maker Faire 2013 Demonstration

At the 2013 Bay Area Maker Faire, PR2Lite took on Maxwell and Michael Ferguson in a robot vs. robot battle of chess **(Figure 9)**. Michael and Maxwell are the 2011 champions of the AAAI small manipulator chess challenge.

Michael also wrote a series on Maxwell in the April, May, and June 2012 issues of *SERVO Magazine*. Both robots were running different variations of the same code.

The Kinect was used to identify where black or white pieces were located and whether the pieces were moved. GNUChess was used to implement the rules of chess, validate legal moves, and track which pieces were where. Finally, PR2Lite used ROS Electric-based arm navigation while Maxwell used the recently released MoveIt! for ROS Groovy.

Maxwell was a formidable opponent, and played an excellent game of chess. PR2Lite tried hard, and provided entertainment when it didn't work. One time, it accidentally picked up its king on its first move, and mistakenly resigned

Future Plans

PR2Lite continues to evolve. Powerful MX64 and MX106 servos and a Velo gripper courtesy of Willow Garage will soon replace the AX-12s in one of its arms **(Figure 10)**. In addition, the arm planning will eventually use MoveIt!. To extend mobility time, we plan to replace its current battery-draining computer and its DC to AC converter with powerful laptops.

PR2Lite has now reached the level of maturity that applications developed for PR2 can be adapted for its own use. Prior PR2 hackathons have demonstrated keyboard playing and drawing capabilities that look like good potential initial candidates. Also, the new HBRC floorbot challenge is intriguing. Information and the latest updates are available at **http://mattdowning.wordpress.com/pr2-lite**.

Only five years ago, it was inconceivable that a hobbyist robot this advanced could be built mainly by high school students for price tag significantly lower than a Nao or Darwin humanoid robots. Now, you can build your own PR2 too! SV

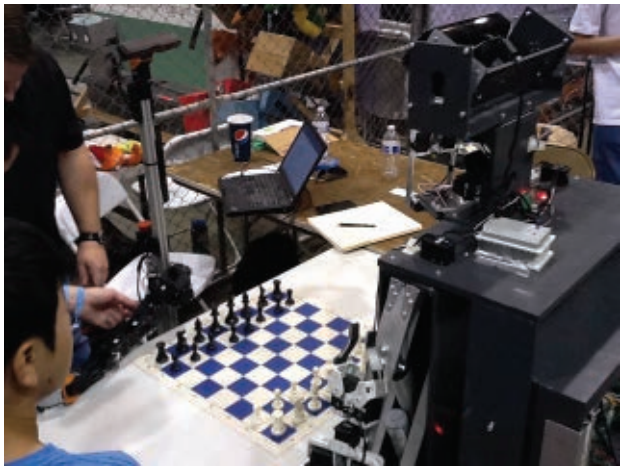


FIGURE 9. PR2Lite (right) plays Michael Ferguson's Maxwell (left) in chess at the 2013 Bay Area Maker Faire in San Mateo, CA.

FIGURE 10. A Velo gripper (left), courtesy of Willow Garage next to our current gripper (right). The Velo is already larger than the current gripper, and it will be even bigger and heavier with servos.