Human-in-the-loop Control of a Humanoid Robot for Disaster Response: A Report from the DARPA Robotics Challenge Trials

Mathew DeDonato, Velin Dimitrov, Ruixiang Du, Ryan Giovacchini, Kevin Knoedler, Xianchao Long, Felipe Polido, Michael A. Gennert, and Taşkın Padır
Robotics Engineering Program, Worcester Polytechnic Institute, 100 Institute Road, Worcester, Massachusetts 01609
Siyuan Feng, Hirotaka Moriguchi, Eric Whitman, X. Xinjilefu, and Christopher G. Atkeson
Robotics Institute, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, Pennsylvania 15213

Received 10 March 2014; accepted 7 November 2014

The DARPA Robotics Challenge (DRC) requires teams to integrate mobility, manipulation, and perception to accomplish several disaster-response tasks. We describe our hardware choices and software architecture, which enable human-in-the-loop control of a 28 degree-of-freedom ATLAS humanoid robot over a limited bandwidth link. We discuss our methods, results, and lessons learned for the DRC Trials tasks. The effectiveness of our system architecture was demonstrated as the WPI-CMU DRC Team scored 11 out of a possible 32 points, ranked seventh (out of 16) at the DRC Trials, and was selected as a finalist for the DRC Finals. © 2014 Wiley Periodicals, Inc.

1. INTRODUCTION

Disaster response has been an impactful research focus in advancing the capabilities of intelligent robots (Burke, Murphy, Coovert, & Riddle, 2004; Casper & Murphy, 2003; Nourbakhsh et al., 2005). The Defense Advanced Research Projects Agency (DARPA) Robotics Challenge (DRC) (Pratt & Manzo, 2013), announced in 2012, is the new frontier in efforts to effectively deploy robot systems in natural and man-made disaster situations. The DRC is aimed at advancing robotics research and development in multiple directions, including perception, manipulation, mobility, and supervised autonomy. The challenge tasks are motivated by real disaster sites such as the Great Eastern Japan Earthquake in 2011 (Nagatani et al., 2013) and Hurricane Katrina in 2005. The tasks include a variety of manipulation (turning valves, clearing debris, opening doors, attaching a fire hose, and operating power tools) and mobility (driving a vehicle, climbing a ladder, and traversing rough terrain) tasks. The DRC Trials took place on December 20–21, 2013 and the event was viewed as a formative assessment of the teams participating in the DRC. Details of the tasks, including the evaluation criteria, were well-defined to give the teams an opportunity to test their hardware and software designs so that they might meet the goals of the DRC finals (DARPA, 2013).

Humanoid robots have advantages for completing a wide variety of tasks in human environments, such as turning valves (Alunni et al., 2013), traversing rough terrain (Hirukawa et al., 2007), and driving a vehicle (Rasmussen, Yuvraj, Vallett, Sohn, & Oh, 2013). However, despite receiving great attention to date, humanoid robot motion planning and control remain challenging research topics. The development of computationally efficient algorithms to solve the inverse kinematics (IK) problem for dual arm manipulation tasks has been the focus of Vahrenkamp, Berenson, Asfour, Kuffner, & Dillmann (2009). Probabilistic IK solvers that rely on rapidly exploring random trees are shown to be effective in re-grasping tasks. A full body balance control architecture for humanoid robots with the input being desired contact forces is presented in Stephens & Atkeson (2010). Torso posture control is presented as an example demonstrating the control approach. The high degrees of freedom that are inherently present in humanoid robots have been used to develop control methods such as whole-body torque control with multipoint contacts (Sentis, Park, & Khatib, 2010). Their controller takes into account the relationship between the contacts and the desired center-of-mass maneuvers to achieve balanced motions that rely on stability polygons determined by the contact points. The annual International Conference on Humanoid Robots provides researchers with a platform to share the latest advances in the field.

The paper is organized as follows: We provide a brief description of the ATLAS robot and the operator control station in Section 2. Section 3 details a comparative study of three robot hands to test their suitability for DRC manipulation tasks. Section 5 describes the software architecture. Finally, we present our methods, results, and lessons learned from the DRC Trials tasks in Section 6.

Direct correspondence to: Taşkın Padır, e-mail: tpadir@wpi.edu

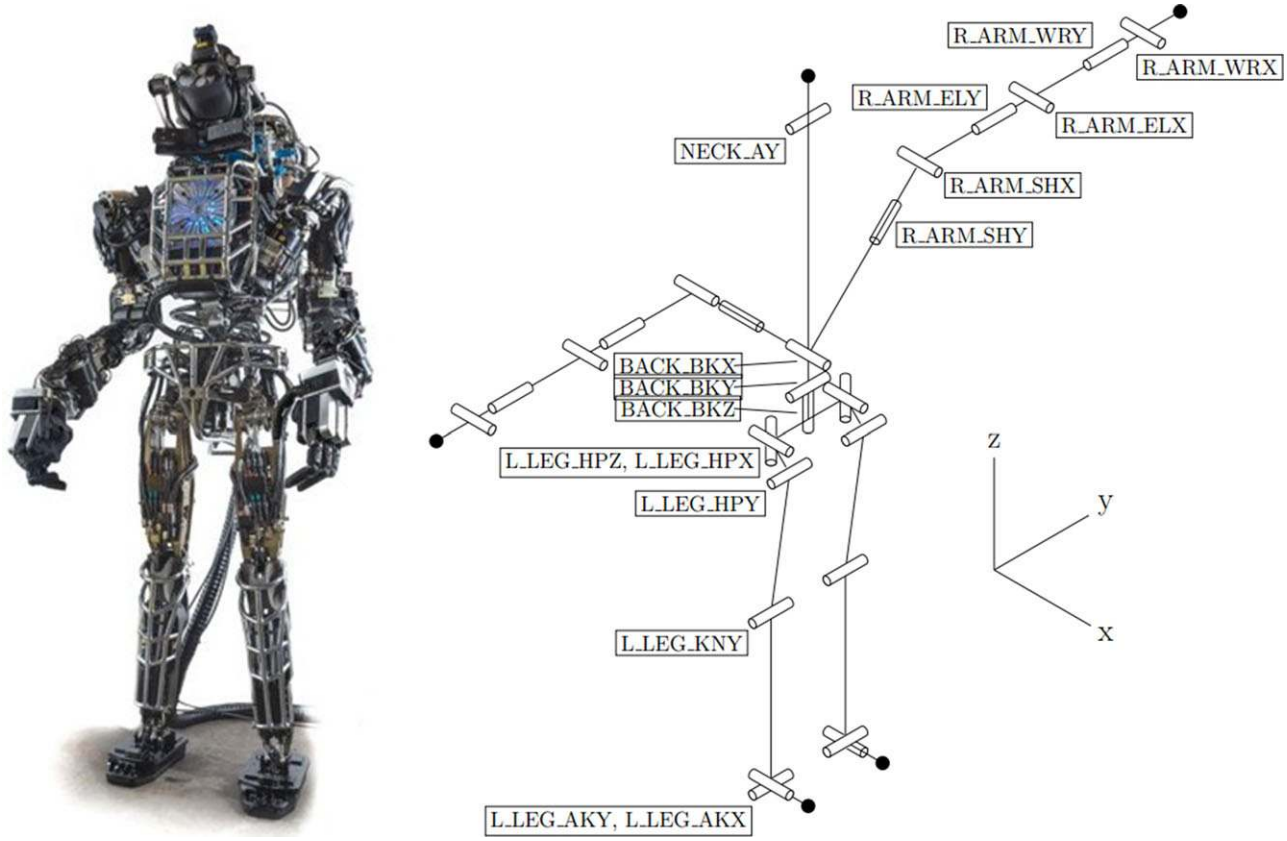

Figure 1. ATLAS robot and a stick figure showing the position and orientation of joints.

2. ROBOT HARDWARE

2.1. Atlas Robot

The Worcester Polytechnic Institute (WPI)–Carnegie Mellon University (CMU) DRC team, originally known as WPI Robotics Engineering C Squad (WRECS), which took 2nd place in the Virtual Robotics Challenge in June 2013, participated in the DRC Trials as the only Track C team. The WPI-CMU DRC team was provided with an ATLAS robot, designed and built by Boston Dynamics specifically for the DRC. ATLAS is a 150 kg humanoid robot with 28 hydraulically actuated degrees of freedom (DOF): 6 in each arm, 6 in each leg, 3 at the torso, and 1 in the neck (Figure 1). Table I presents a description of the arm joints. In addition to load cells for force sensing at the hands and feet and a fiber-optic inertial measurement unit (IMU) at the pelvis for estimating the robot pose, each actuator on the arms has a linear potentiometer for position measurement and two pressure sensors to determine the joint forces based on differential pressure measurements.

The robot's sensor suite also includes three IP (ethernet) cameras positioned around the robot to allow for a near 360° view of the surroundings and a Carnegie Robotics MultiSense SL sensor head that provides visual input to the operator (CRL, 2014). MultiSense SL contains a set of stereo vision cameras and a rotating LIDAR, and it can be used to produce a point-cloud to represent the robot view. Because of the high power and data requirements of the system, ATLAS is tethered to a base station. This tether supplies the robot with 480 V of power, a fiber-optic connection of 10 Gbit/s for network communication, and water cooling. Sensors communicate directly to the Control Station over the fiber-optic network (Figure 2).

Table I. A description of ATLAS arm joints.

DOF	Joint	Min	Max	Range	Description
1	SHY	−90°	45°	135°	Shoulder axial rotation
2	SHX	−90°	90°	180°	Shoulder perpendicular rotation
3	ELY	0°	180°	180°	Elbow axial rotation
4	ELX	0°	135°	135°	Elbow perpendicular rotation
5	WRY	0°	180°	180°	Wrist axial rotation
6	WRX	−67.5°	67.5°	135°	Wrist perpendicular rotation

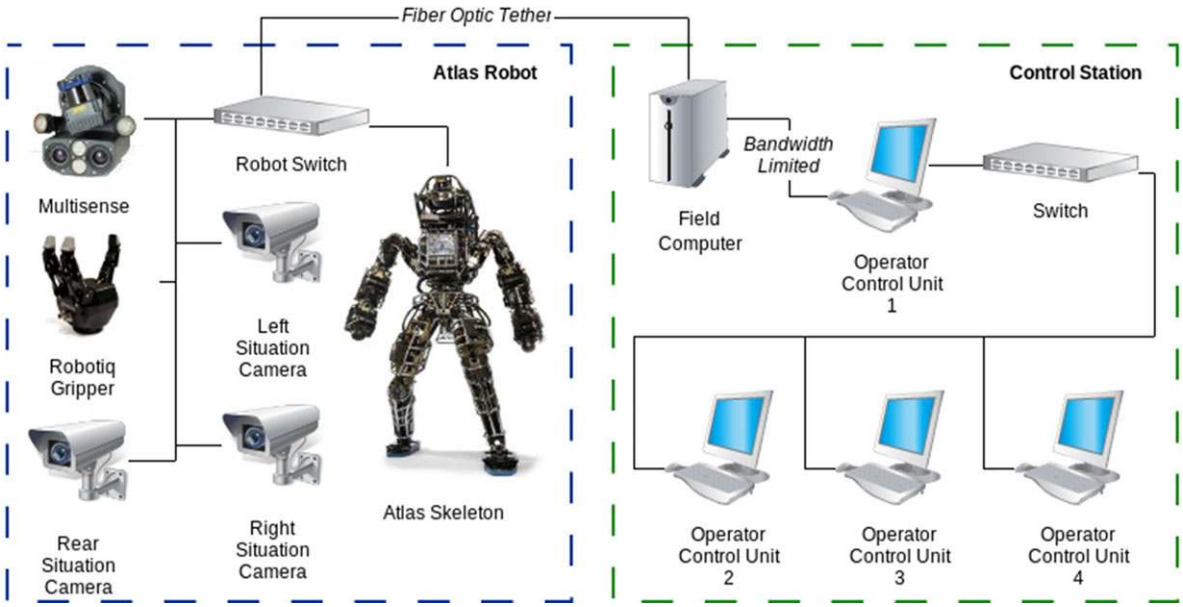

Figure 2. A visualization of the network layout between the robot sensors and the operator control station.

2.2. Control Station

The control station enables the operators to control the robot remotely without line of sight, and it is comprised of five computers: the field computer, the primary operator control unit (OCU), and three auxiliary OCUs. The field computer manages all communications with the robot, limits and compresses the high-resolution data from the robot to be sent to the OCUs, and runs the autonomous robot software. The primary OCU (OCU 1 in Figure 3) is tasked with decompressing and relaying information sent over the communications pipeline. The auxiliary OCUs act as terminals to accommodate information to and from the users. The field computer is connected directly to the ATLAS network through the fiber-optic line, and it is also connected to OCU 1 through a limited bandwidth connection specified by DARPA. At the DRC Trials, the connection bandwidth and latency alternated every minute between a bandwidth of 1 Mbit/s and a latency of 0.05 s each way (0.1 s round-trip) and a bandwidth of 0.1 Mbit/s and a latency of 0.5 s each way (1 s round-trip). The primary OCU and auxiliary OCUs are interconnected via a standard gigabit network (Figure 2). All OCUs are able to run the user interface. To provide the operator with an unrestricted view of the interface, each computer has dual monitors, with OCU 1 and OCU 2 containing a second row of displays above the first. These secondary displays mirror the screens of the opposite OCU, giving each user awareness of what his co-operator is doing (Figure 3).

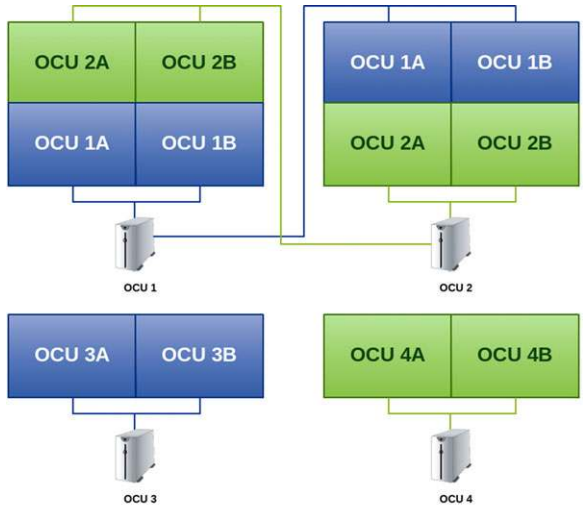

Figure 3. The layout and functions of each display in the control station and which Operator Control Unit (OCU) they connect to.

3. A COMPARATIVE STUDY OF ROBOT HANDS

Since manipulation is an essential requirement in the DRC tasks, we performed an experimental evaluation of the iRobot, Sandia, and Robotiq hands, shown in Figure 4, that could be interfaced with ATLAS. Table II provides a

Figure 4. Left to right: Robotiq, Sandia, and iRobot hands.

Table II. Robot hand specifications.

	iRobot	Sandia	Robotiq
No. of fingers	3	4	3
DoF	5	12	4
Weight (kg)	1.53	2.95	2.3
Base size $L \times W \times H$ (cm)	$13 \times 11.5 \times 8.25$	$11.5 \times 11.5 \times 18$	$12.75 \times 12.75 \times 10.25$
Drive type	Worm gear	Gears	Worm gear
Trans. type	Spectra braid line	Steel cable	Mechanical linkage
Max current 24 V	5 A	2.5 A	1.5 A
Mechanical safety	Magnetic finger coupling	Mechanical fuse	Mechanical fuse

summary of the design specifications of each robotic hand being compared in this study.

Each of the three fingers on the iRobot's hand is controlled through a tendon that runs the length of the finger and is powered by an electric motor and wormgear combination built into the base. Because the finger joint between the distal and proximal phalanges is flexible, each finger can conform to the object being grasped as the tendon is reeled in. The thumb has a second powered tendon connected to the back side of the proximal phalange for improved control. The last degree of freedom comes from an independent motor that controls the spread of the two fingers opposite to the thumb, giving the hand extra compliance for small or round objects. In contrast, each finger of the Sandia hand is an independent module with three degrees of freedom: lateral movement on the base, base joint flex, and distal phalange joint flex. Each joint is controlled through a system of steel cable pulleys. The electric motor actuators are built into the base of each finger and rely on a backdrivable gear reduction to drive the steel cables. Lastly, the Robotiq hand is similar to the iRobot in design; it has three underactuated fingers driven by worm gears and the ability to spread the two fingers opposite the thumb. But unlike the iRobot hand, compliance is achieved through mechanical linkages, making it more robust to external forces during manipulation tasks.

For feedback to the operator, the Sandia hand is equipped with two cameras on the palm for visual approach while reaching for objects and the possibility of stereo vision. It also has an array of tactile sensors on its palm as well as current feedback from the drive motors. However, the Sandia hand has no tactile feedback on the fingers. The iRobot hand comes with optional "smart" fingers that include accelerometers, encoders, flex sensors, and tactile sensors. The Robotiq hand provides only the motor encoder and current measurements.

We performed two sets of experiments in order to evaluate the three robot hands. The tests involved maximum tensile and shear force measurements of each hand while grasping a variety of objects listed as debris pieces in the DRC rules. The objects used consist of $2 \times 4 \times 36$ in. ($5.08 \times 10.16 \times 91.44$ cm) and $4 \times 4 \times 36$ in. ($10.16 \times 10.16 \times 91.44$ cm) wood pieces, and a 1.5 in. (3.81 cm) diameter by 36 in. (91.44 cm) long metal pipe. For all the tests, each hand was positioned around the object being tested, and the grasping force was set to the maximum allowed.

The tensile tests involve resistance against objects being pulled perpendicular to the palm, while the shear tests involve resistance against objects being pulled parallel to the palm. Each test was done with multiple object orientations in order to access the best gripping technique. The test descriptions are related to the face of the object that is in contact with the palm, which means, for example, that the "2×4 short test" means the 2 inside of the object is parallel to the palm.

Figure 5 shows the setup used for running the tests. On one side of the 2×4 that runs along the table and forms the

Figure 5. Stress test setup.

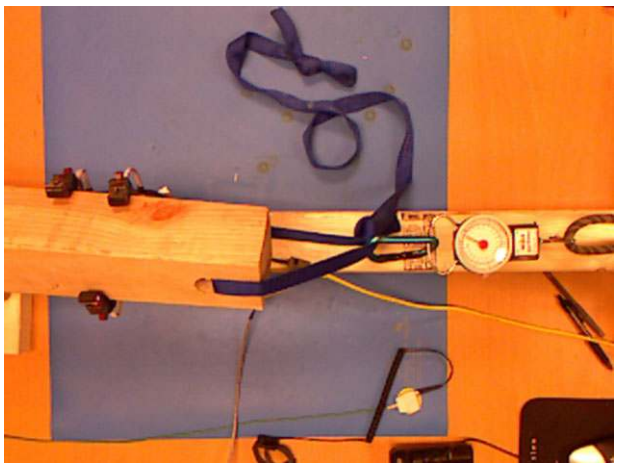

Figure 6. iRobot 4 × 4 diamond shear test.

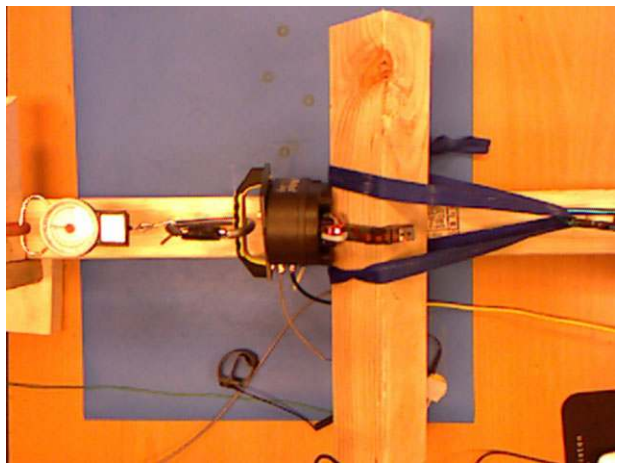

Figure 7. iRobot 4 × 4 diamond tensile test.

base of the setup, a bolted metal vice can generate large controllable forces. The hand to be tested is placed on the middle and attached to a fixed point on the other end through a spring and hook scale. For the shear tests, the hand was firmly secured into the setup itself; for the tensile tests, the hand was held floating in midair by the scale on one side and the grasped stock material on the other (Figure 8). Furthermore, Figures 6, 7, 8, and 9 illustrate the standardized tests performed within the scope of this study.

Tables III and IV summarize the results of tensile and shear tests. As an extra set of experiments for the Robotiq hand, we tried grasping normally against the palm as well as only using the fingertips. As a result of this comparative study, we chose to use the Robotiq hand as our active gripper. Nevertheless, none of the hands was strong enough to perform some of the tasks required for the DRC.

3.1 ROBOTIQ THREE-FINGER HAND

For the DRC Trials, the WPI-CMU team opted to use one Robotiq end-effector. The next couple of sections will outline the user interface to control the hand and hardware modifications specific to the DRC tasks.

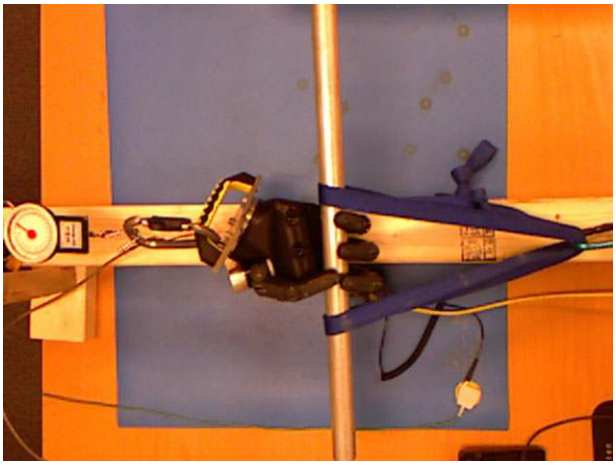

Figure 8. Sandia pipe tensile test.

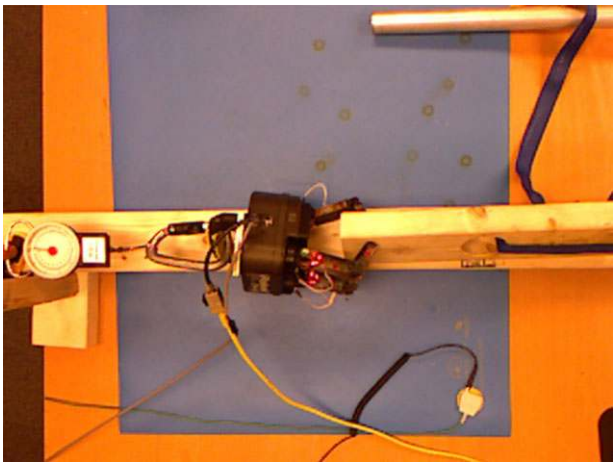

Figure 9. iRobot 2 × 4 end tensile test.

Table III. Experimental results of the tensile stress tests.

	Tensile (N)			
	iRobot	Sandia	Robotiq Palm	Robotiq Tip
2 × 4 Pine				
Short	9	27	220+	98
Long	142	220+	220+	53
End	13	4	89	89
4 × 4 Pine				
Diamond	58	75	220+	220+
Square	49	106	220+	89
End	22	4	98	89
Steel Pipe				
Across	220+	220+	220+	220+

Table IV. Experimental results of the shear tests.

	Shear (N)			
	iRobot	Sandia	Robotiq Palm	Robotiq Tip
2 × 4 Pine				
Short	120	40	142	89
Long	93	58	220	76
4 × 4 Pine				
Diamond	142	49	111	76
Square	62	40	200	111
Steel Pipe				
Across	220+	89	220+	220+

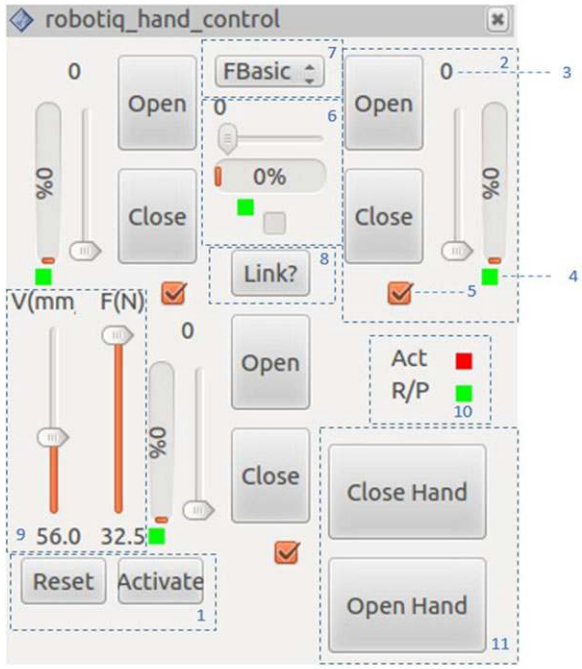

Figure 10. Graphical user interface designed for the Robotiq hand.

The end-effector control was kept separate from most other systems for the competition. The Robotiq hand provides several modes of operation, such as basic mode, wide mode, pinch mode, and scissor mode (Robotiq, 2014). We can also control each finger independently. At the same time, we can implement position, speed, or force control on the hand. We designed a ros-visualization (rviz) panel as a graphic user interface (GUI) to control the hand (Figure 10). It includes buttons used to reset (or deactivate) and activate the hand; a control & display set, which is used to control specific finger and show its status; "Open" and "Close" buttons, which are used to control individual fingers; an

Figure 11. Robotiq finger modifications.

Figure 12. Ladder task pipe hands.

indicator providing status information for each finger; a checkbox used to enable and disable the control of the finger; a control & display set used to control the angle between the two contiguous fingers; a drop menu used to change the mode of operation; and a button used to link the fingers. In the default case, the user can control the fingers separately. If the fingers are linked, they would rotate together. Sliders are used to set the velocity and force parameters for controlling the hand. An "Act" indicator shows that the hand is activated (green) or not (red). The "R/P" indicator shows that the motor of the finger is running (red) or pending (green). Finally, there are buttons designed for fully opening and closing the hand.

The Robotiq hand allows for fingertip modifications, and distal and proximal phalange finger pad alterations. Based on this, a few task-specific finger modifications were designed to improve performance. Micro spikes were added for finer grip strength on the debris task. A proximal distal extension was used for drill operation, and a hose was attached to the fingertip. Figure 11 shows the finger modifications for the drill task on the proximal phalange, and the hose attachment modification on the fingertip. In addition, a number of interchangeable passive end-effectors were designed specifically for each task. For example, we used hook-pipe hands on the ladder task (Figure 12), a 6 in. (15.25 cm) pipe hand in the vehicle task for steering and the

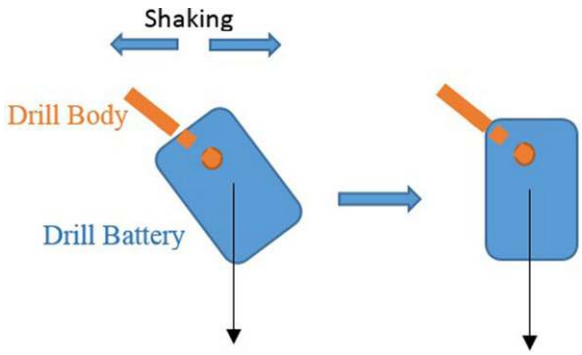

Figure 13. Shaking strategy.

valve task, and an 18 in. (45.75 cm) pipe hand in the door task to turn the door handle.

To increase grasp reliably for the debris task, our team experimented with a variety of fingertip spikes and extensions. After extensive testing, we opted for a two-spike combination per finger without substantially increasing the finger size. These spikes bite into the different debris pieces, securing them inside the grasp. Only two spikes from opposite fingers were need for the piece to be reliably moved around.

For activating the rotary tool during the drill task, an extension to the proximal phalange of one of the fingers was added (Figure 11). This extension was designed in a such way that when the rotary tool was placed in a specific way on the grasp, the closing of the finger motion would press the on/off button and turn the tool on. Grasping the tool from the table in this precise position is very difficult. Thus a dynamic alignment technique was developed consisting of a prescripted shaking motion. The shaking strategy is based on the fact that the battery of the drill, the heaviest part, is assembled at the bottom. Since the mass center of the battery is not on the axis of the drill body, the drill can be rotated using the torque generated by gravity (Figure 13). The drill can automatically be positioned in the desired orientation after shaking for a period of 30 s at approximately 1 Hz.

The hose attachment tool consists of a 1/8 in. L-shaped aluminum piece 2.5 in. long (Figure 11). The concept is that the specific finger would be placed over the area of the fire hose that can spin freely. The finger could be opened and the arm rotated counterclockwise around the axis of the fire hose until the limit of the full body controller. Then the finger could be closed again so as to engage with the freely moving screw portion, and the sequence can be repeated.

4. SOFTWARE ARCHITECTURE

The overall goal in designing the WRECS software architecture is to enable human-in-the-loop control of a complex robotic system over a limited bandwidth link. To meet this goal, tasks that require high bandwidth and/or low latency

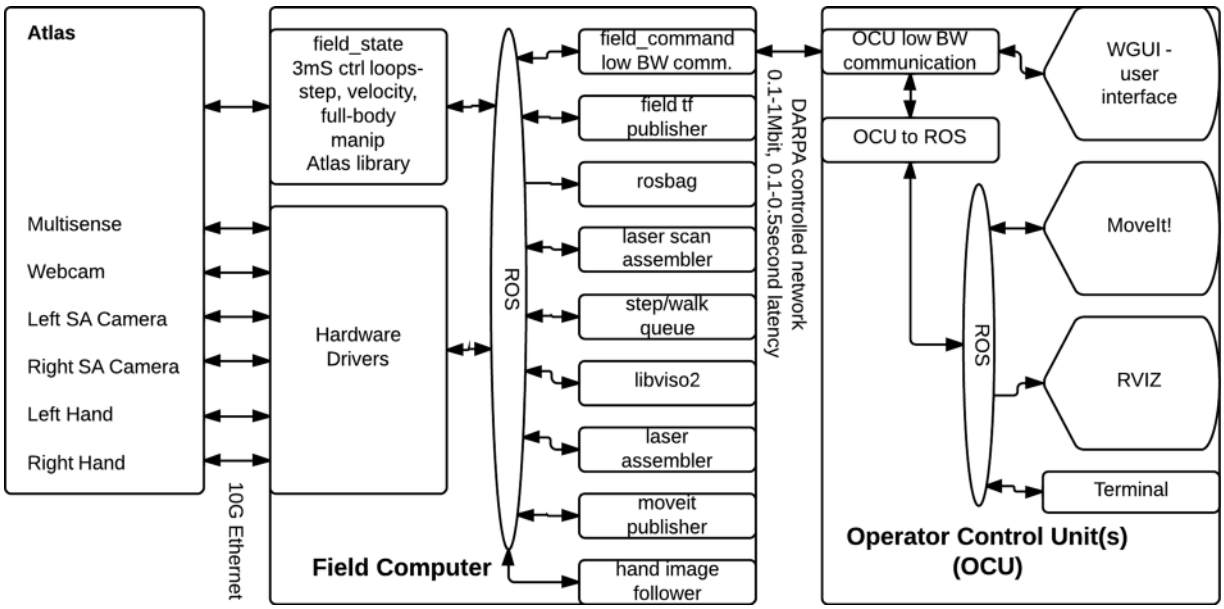

Figure 14. A flow chart showing the software architecture and communication between different subsystems.

communications are run on the field computer. The field computer is connected to the ATLAS robot via 10 G Ethernet. User interfaces and tasks that work with low bandwidth and high latency are run on the OCUs.

The software on the field computer is divided into several subsystems (Figure 14). The various parts communicate with one another using ROS (Quigley et al., 2009). `field_state` is the primary interface to the robot hardware. All of the critical control loops that run synchronously with the robot 3 ms controller cycle time are run within this process. Those control loops include full-body manipulation, stepping, and velocity control. The `field_command` process is responsible for compressing and managing data sent to the operator control unit over the DARPA controlled network (0.1–1 Mbit/s, 0.1–1 s round-trip, or 50 mS to 0.5 s one-way latency). `field_command` compresses and limits outgoing data.

The OCUs provide interfaces to the human operators controlling the robot. These include our custom user interface (WGUI), and modified versions of MoveIt! and rviz. One process is responsible for managing the low bandwidth communication link with the field computer. That task decompresses incoming data and compresses outgoing data. A separate process (OCU to ROS) acts as a bridge between our custom protocol and standard ROS messages as used by MoveIt! and rviz.

One of the keys to our success was providing good situational awareness to the human operator and allowing the human operator to control the robot at several different levels of abstraction. Providing good situational awareness comes down to compressing data well and making effective use of the limited bandwidth pipe between the field computer and the OCU. To make good use of the pipe, we sent smaller and more compressed data during low bandwidth (0.1 Mbit/s) times. This was important, as sending a large data set (1 Mbyte) over the 0.1 Mbit/s pipe would take over a minute.

The various levels of human operator control were important to allow flexibility to solve problems. The lowest level was direct control of individual joint positions and forces. A level above that was inverse kinematic control of the feet and hands. Above that was a full-body manipulation controller. Each of these also had various levels of automation, GUIs, and scripting. These allowed the operator to choose an appropriate level of abstraction/automation when solving a problem.

5. RESULTS AND DISCUSSION

Next, we discuss our approach, results, and lessons learned in the DRC Trials tasks to demonstrate our overall strategy for human-in-the-loop control of the ATLAS robot.

5.1. Vehicle

The vehicle tasks consist of a small utility vehicle the robot must drive through a slalom course of highway safety barriers. Our approach focused on allowing the robot to control its speed, while implementing operator-assisted steering. The stereo cameras and LIDAR are used to generate two independent estimates of the velocity of the vehicle (Figure 15). The operator can choose which velocity estimate is used by the robot depending on the environment.

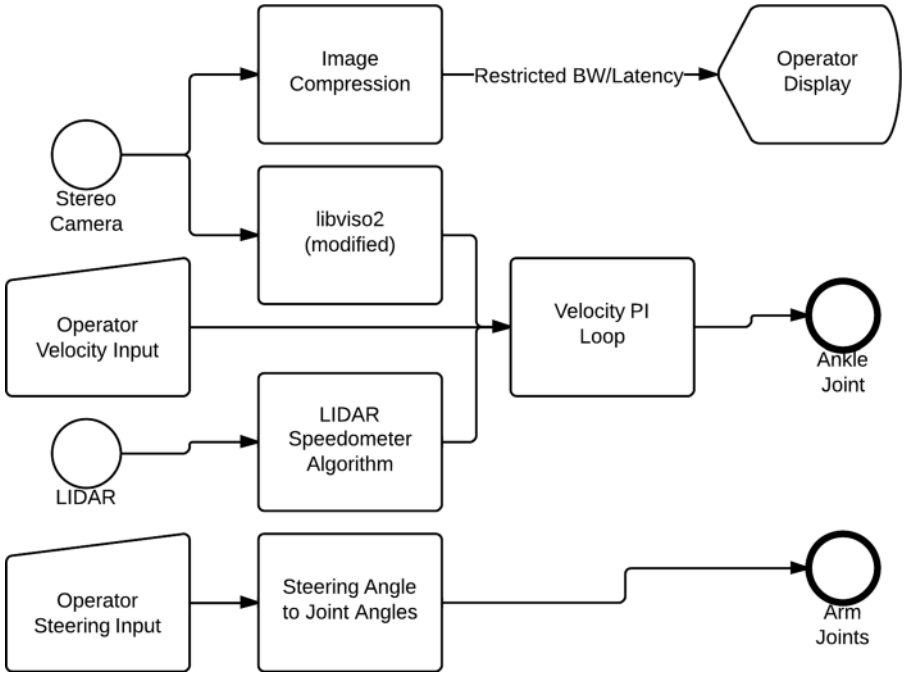

Figure 15. The robot uses its stereo cameras and LIDAR to estimate the velocity of the vehicle.

In visually dense environments, the stereo cameras with a modified version of the `libviso2` package provide very good data. In relatively flat/poor lighting or environments with little visual diversity, the LIDAR-based estimate provides better data. The desired velocity is provided by the operator and is passed to a PI controller that actuates the throttle using the ankle joint of the robot. Once set, the robot drives the vehicle at a steady and slow pace autonomously.

Most of the complexity in navigating obstacles is associated with steering, which human operators are very good at handling, even in latency-degraded conditions. Therefore, steering is handled by the operator in real-time, allowing the robot to adapt to difficult or changing conditions easily. Visual feedback is provided to the operator from the stereo cameras. The images are compressed to preserve bandwidth and displayed to the operator. The operator then commands a steering angle, and the robot generates the appropriate arm joint angles to execute the desired steering angle.

Our robot successfully drove the course at the 2013 DRC Trials reaching the finish zone in 6 min, 21 s and earning one point. This result was the fastest posted by any team at the trials and earned Team WRECS a "Best In Task" award for the driving task. In addition, ours was the only ATLAS robot to successfully drive.

Figure 16 shows the visual feedback provided back to the operator to help set the desired steering angle at the DRC Trials. The image is fuzzy and degraded due to the significant compression applied to keep bandwidth use low, but it is still very usable for avoiding obstacles. The large highway barriers are easily visible, and some sense of distance can be extrapolated for the context in the image.

While the current implementation proved successful at the 2013 DRC Trials, significant improvements can be made to enable new features and improve robustness. For this approach to be effective in a real disaster scenario, the robot needs to be able to enter the vehicle by itself, drive to the affected location, and exit the vehicle. Significant development needs to be done to enable ingress and egress since the vehicle fit is so tight.

5.2. Terrain

The rough terrain task consisted of walking over inclines and then piles of cinder blocks, including tilted cinder blocks. In our initial tests, the walking and step controllers from Boston Dynamics were able to walk over much of the terrain, but not all of it. We therefore decided to develop our own walking controller, based on our previous work (Feng, Xinjilefu, Huang, & Atkeson, 2013; Stephens, 2011; Whitman, 2013).

Our analysis of the DRC Trials terrain task was that it was a "stepping stone" task, in that it required accurate foot placement. The robot's feet needed to be placed within the boundary of individual cinder blocks. We therefore developed a walking controller that focused on achieving known footstep targets, while footstep plans were manually generated.

Figure 16. The view provided back to the operator during the 2013 DRC Trails.

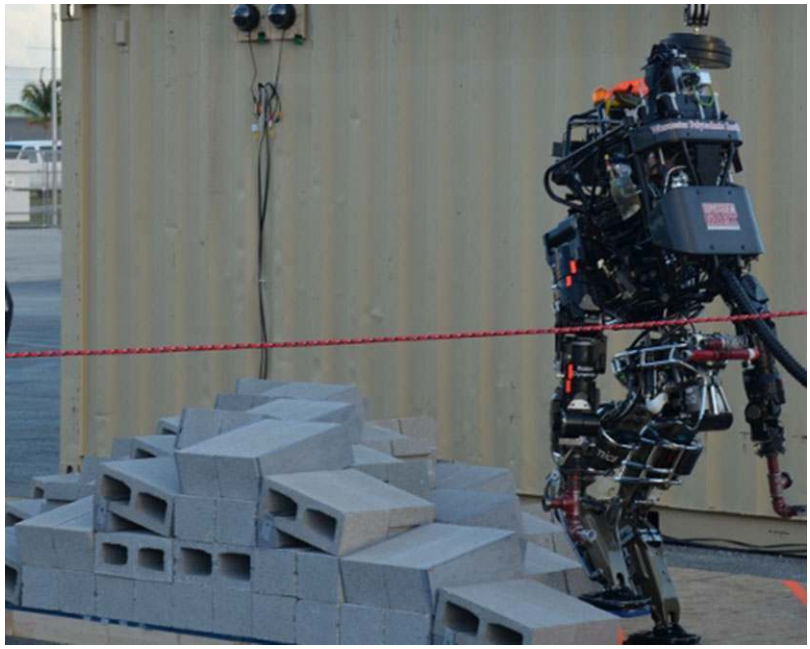

Figure 17. Terrain task.

A second decision we made was to walk slowly to minimize the risk of falling over. Our walking in the DRC Trials was essentially statically stable. This decision was driven by the limited development time and the substantial modeling and torque measurement errors we were seeing on the ATLAS robot. The high level desired motions, such as center of mass (CoM) and swing foot trajectories, were generated with quintic splines. The given foot step locations were used as knot points for the splines. The desired center of pressure (CoP) trajectory was generated using a linear inverted pendulum model (LIPM).

On each controller tick (every 3 ms), an inverse dynamics calculation was performed to generate appropriate joint torques (similar to computed torque control), and an inverse kinematics calculation was performed to provide desired joint positions and velocities. The inverse dynamics calculation was made more complicated since the robot has a "floating base" and is not rigidly attached to the ground. A quadratic programming problem was solved to trade off achieving desired joint accelerations, joint torques, and contact forces. Similarly, an inverse kinematics component also solved using quadratic programming was added to

Figure 18. Ladder task.

the controller to help deal with modeling errors associated with the ATLAS robot, as well as provide desired joint positions and velocities for the control system provided by Boston Dynamics.

We also developed a state estimator to estimate pelvis translational velocity and Cartesian position. We used the IMU orientation estimate directly. Based on which foot was on the ground, we used leg kinematics to provide us with a "measurement" of pelvis velocity and position. We used a simple Kalman filter to process this information.

In terms of operator control for walking, we provided our human operator with a live camera stream augmented with the current swing foot pose computed from forward kinematics, and we let the operator "nudge" the swing foot around in the six-dimensional Cartesian space by commanding offsets in foot position and orientation. Once the operator was satisfied with the foot pose, a "continue" command was given, allowing the robot to lower the swing foot straight down until ground contact was detected. We chose this approach because operators understood the robot images more easily than laser scan data.

Our approach, however, requires substantial input from the operator, and it extends the single support phase unnecessarily since the operator commands were given during single support rather than double support. We plan to refine this approach in future work.

To fully control CoP to achieve better balancing and be more robust to perturbation during walking, we controlled the stance ankle joints in pure torque mode. An integrator on swing foot placement was used to compensate for stance ankle position errors.

For static walking, the CoM needs to be completely shifted to the next stance foot during double support. When taking longer strides or stepping to a greater height, extending the rear leg knee alone is often insufficient to move the CoM all the way. Toe-off is one solution to this problem. During double support in our controller, toe-off is triggered when the rear knee approaches the joint angle limit (straight knee).

Once triggered, special modifications are used in both the inverse dynamics and inverse kinematics controllers. We first move the rear foot reference point, where the Jacobian is computed to the toe. In the inverse dynamics controller, the contact cost term for the rear foot is transformed to its local frame, and the row that corresponds to pitch angular acceleration cost is removed. We also constrain the allowed pitch torque to be zero. This effectively turns the rear foot contact into an unactuated pin joint around the pitch axis. In the inverse kinematics controller, we transformed the rear foot's pitch tracking error into the foot frame and dropped the pitch term. A slightly bent rear knee angle is used to bias the inverse kinematics toward using ankle angle for a toe-off solution.

During static robot experiments, the measured CoM location, which is measured with foot force sensors, deviates from the model's prediction. We also believe this modeling error depends on the robot configuration. During the second half of double support and the full single support phase,

we integrated this error and used it to offset the desired CoM location so that the true CoM matched the desired CoM. Assuming the robot is moving slowly enough, we can approximate the true location of CoM with the measured CoP.

Due to the tight schedule for the DRC Trials, we have not conducted systematic system identification procedures on the robot. We hope to improve the quality of both kinematic and dynamic models in the near future. All the leg joint level sensing on the ATLAS robot, such as positioning, velocity (numerically differentiated from position), and torque, are pretransmission. This hardware design choice reduces jitter in the low-level joint control, but it introduces problems in the forward kinematics and torque control. Better state estimation is necessary to achieve more accurate position tracking and force control.

We learned that our walking approach largely worked on the physical robot, although we did not adequately handle knee torque limits. Compared to our previous work, the inverse kinematics component of the controller was added to help deal with modeling errors associated with the ATLAS robot, as well as to provide desired joint positions and velocities for the control system provided by Boston Dynamics. We found that inverse dynamics plays a more important role for heavily loaded joints such as the stance leg. In particular, center of pressure control is very important for balance. However, on lightly loaded limbs (e.g., swing leg or arms), friction and unmodeled dynamics tends to dominate the rigid-body model torques that inverse dynamics produces, so inverse kinematics is more helpful. We considered explicitly switching between control modes as the contact state changed, but the switching transients were hard to manage. Performing both inverse dynamics and inverse kinematics concurrently for all joints eliminated the switching issue, and we found that the appropriate control mode dominated at the appropriate time.

In our own tests we were able to walk across the DRC Trials terrain course. At the DRC Trials, we began to have problems where the controller would ask for more torque than a knee was capable of, and that knee would collapse. In our actual terrain DRC Trial (Figure 17), we successfully walked across the first two sections of terrain earning the first two points allocated to the task, and then a knee collapsed in the middle of the third section. A subsequent attempt on the third terrain section was started too far to the side and the belay weights pulled the robot over. Therefore, the robot did not climb over the roughest third section, where no surface was level to the ground, and it did not earn the respective third and fourth points.

5.3. Ladder

The difficulty of the ladder task (climbing a steep staircase) depended on the kinematic capability of the robot. The ATLAS robot did not have enough ankle joint range to walk up the ladder with knees facing backward while keeping its feet flat. It would have had to have at least one foot tilted and on a tread edge for a statically stable climb. The ATLAS robot also did not have sufficient backward-facing sensing to guide the robot up in reverse. We decided climbing the ladder backward was too risky given the limited development time.

ATLAS had other challenges in climbing the ladder facing forward using a statically stable climb. At one point, one foot is on a tread, and one foot is on the next tread up. The tread above that obstructs the calf of the upper leg, forcing the knee to be moved backward and the CoM of the robot to be moved behind the polygon of support. The arms must compensate for the tipping moment. We initially explored grabbing the railings to provide the stabilizing force (as humans did in our initial testing). We ran into two problems. The hands provided by DARPA (Sandia and iRobot hands) did not have sufficient grip strength or reliability. The grasps seemed vulnerable to misalignment and large twisting moments on the hands. The second problem was that ATLAS's arms are quite weak, and the arms grasped the railings in configurations that accentuated that weakness.

We considered grasping the treads in front of the robot instead of grasping the railings. This allowed the arms to be almost fully extended, which is a configuration in which with limited torques quite large hand forces can be generated in the direction of the length of the arm. At about that time we were allowed to consider alternative hand designs, and we rapidly prototyped hook hands based on plumbing pipe segments. The hook hands hooking the treads ahead of the robot were able to generate quite large grasp forces. The approach of hooking onto the treads worked well until the top of the ladder was reached, and there were no more treads to grasp. We would then have to switch over to grasping the railing. With the time available, we were able to climb the ladder to the point where there were no more treads to grasp.

The controller for ladder climbing was similar to our controllers for rough terrain walking and full body manipulation. The high-level control was provided by a manually generated script that implemented a quadruped climbing gait. The robot climbed one tread at a time. First the arms grasped a tread. Then one foot moved up one tread, followed by the other foot moving to the same tread. One arm moved up one tread, followed by the other arm. We inserted opportunities for a human operator to affect limb placement. The robot would look at where the limb was supposed to go. The limb would move near to its target. The operator could use the keyboard to precisely place the limb with 1 cm increments horizontally, using visual feedback from the robot's cameras. The correct vertical height was found automatically, using force sensors to detect contact for the feet and position sensing when contact is known to have already occurred for the hands.

Low-level control was provided by our inverse dynamics and inverse kinematics controllers. Once on the steps, only the toes of the feet are supported, so we adjusted the center of pressure constraint accordingly. Having all of the weight on the toes makes the robot vulnerable to rotational slipping, causing unexpected yaw rotations. To correctly place the hands on the next step to recover from such rotations, we needed to rotate the inverse kinematics solution to match the measured orientation. We therefore periodically rotated the inverse kinematics solution such that the feet were aligned with the measured feet orientations, allowing the robot to reorient its upper body toward the ladder and correctly reach targets in the real world.

The robot's shoulders are nearly as wide as the railings, so the necessity of shifting weight from side to side results in a significant danger of bumping the arms on the railings. We avoided such collisions by estimating the railing location based on the hand location (based on the assumption that the hand is pushed up against the side of the step) and adding inequality constraints to the inverse kinematics quadratic program. The inequality constraints limit how far outward each elbow can move in the lateral direction. Additionally, when we wanted to intentionally lean on the railing, we provided a desired elbow location (only in the lateral direction) with a low weight. To prevent the problem from becoming overly constrained by elbow management, we used low weights for commanding hand orientation.

Our robot model had inaccurate forward kinematics. One result is that if the hands are resting on one step and the robot steps up one rung on the ladder, even though the true position of the hands will not have moved, the measured position will have moved several centimeters. We therefore introduced an integrator that gradually adjusts the desired position of both hands in the horizontal plane based on the deviation between the measured and desired CoM position. Essentially, we used the arms to pull or push the CoM into the desired position. To avoid unintentionally rotating the robot, this integrator was only active while both hands were in contact with the step (not during hand repositioning).

At the Trials, the robot performed similarly to testing, climbing until it ran out of treads to grab in front. We passed the first and second markings on the ladder, earning the first and second points in the task. The robot was unable to crest the landing at the top of ladder due to a lack of convenient handholds, and therefore it did not achieve the third and fourth points. This was the highest any ATLAS robot climbed the ladder (Figure 18).

5.4. Debris

While our team had confidence in scoring at least one point in all of the other tasks, we knew that the debris task was one of our weakest. The original plan for the debris task was scrapped within the last month of development due to its complex nature. We settled on a user-assisted, preprogrammed approach to this task instead. Our approach consisted of preprogramming grasp positions for each piece of debris. Once the robot arm was in a relatively close position, the user would then use the keyboard and the cameras to nudge the arm into place around the debris. The hand was then closed, and the user made visual confirmation of the positive capture through the single image. Once captured, a preprogrammed script was run to lift and remove the piece of debris.

During the DRC Trials, we managed to remove 4 out of 15 pieces of debris successfully. To receive a point, however, we needed to successfully remove a fifth piece of debris. During our attempt to remove the fifth piece, the robot contacted the cinder block wall upon which the debris was resting, and it lost balance and fell over. The subsequent reset of the debris task left us too little time to continue, so we failed to score any points on the task.

5.5. Door

The door task simulates the situation in which the robot needs to walk through a series of doors and reach an area to perform further tasks. The task involved opening and passing through three doors. The first and second doors are push and pull doors, respectively, while the third one is a pull door with weighted closure. Passing through a 33.5 in. (85.09 cm) wide door requires the operator to align the robot with the doorway accurately.

To complete this task, ATLAS first walks toward the left side of the door. The right arm is used to turn the door handle. When the door is unlatched, the robot just nudges the door with its right hand or uses its left hand to open the door fully, depending on the door orientation. Finally, the robot adjusts its position and turns 90° to walk through the door sideways. During this process, the operator selects points in the LIDAR point cloud to mark the location of the door handle and the door opening, and to plan the robot motions. The robot is controlled by the operator to make sure that it does not fail due to false detection. After lab testing, it was evident that opening the door with the Robotiq hand required too much time and precise control from the operator. The final choice for an end-effector was a pipe with an attached spring, which turned out to be a simple, flexible, and efficient solution. The left hand was replaced by a longer straight pipe due to the consideration of expanding the work space of the left arm. To accelerate both the walking and the manipulation process, a door task control panel was designed (Figure 19). Subtasks such as reaching toward the door handle and stepping through doorway were scripted.

Even though our strategy to perform the door task proved to be effective in our lab tests and during the practice runs at the DRC Trials, the strong winds on the day of the trials prevented us from finishing this task. For the first push door, winds caused the door to close twice after we managed to open it and prepared to pass through. As

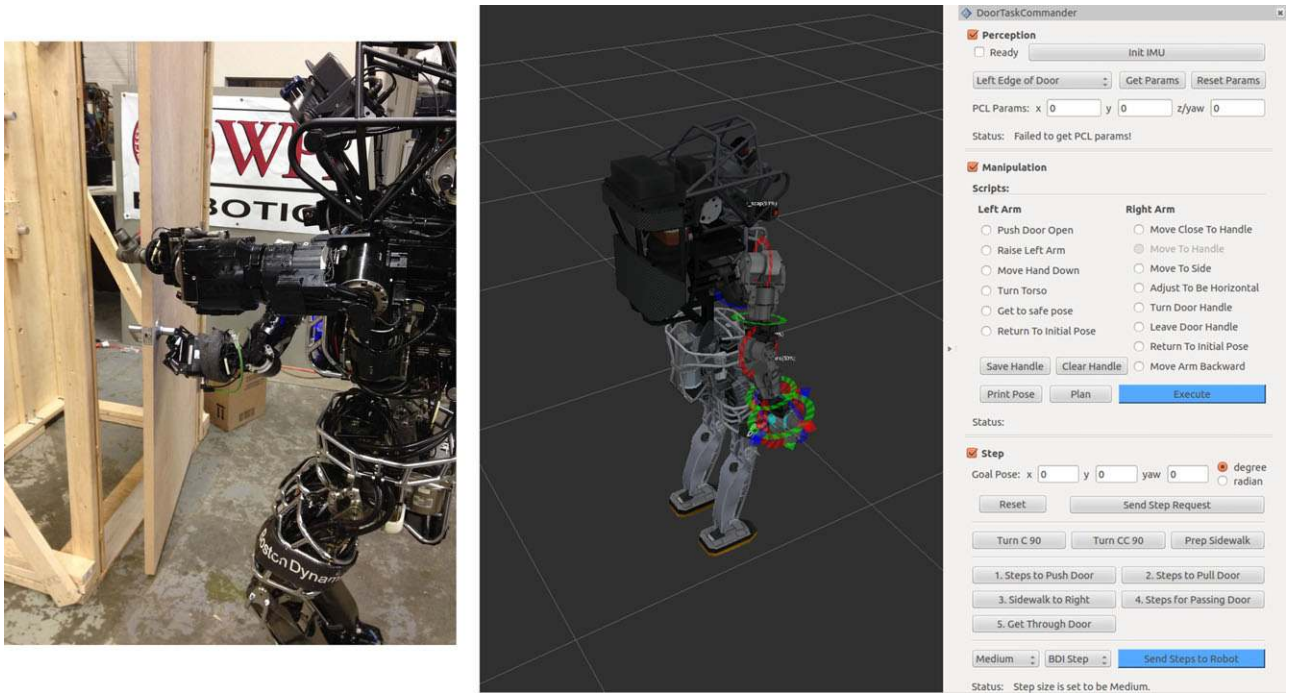

Figure 19. ATLAS robot opening a pull door (left) and the door task control panel (right).

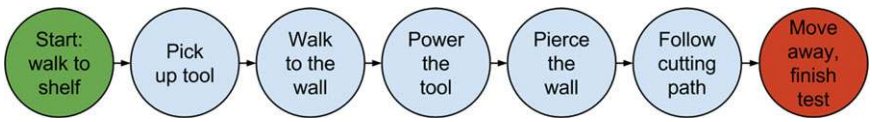

Figure 20. Flowchart of the wall task.

a result, there is a need to develop robust two-arm manipulation algorithms while keeping the robot balanced. Since we were unsuccessful at opening the first door, we did not score any points.

5.6. Wall

For the wall task, the robot must use a single-handed cordless rotary tool with an on/off switch to cut and remove a prescribed triangular-shaped piece from a wall. The wall material is 0.5 in. (1.27 cm) thick drywall. The three vertices of the right triangle are 6 in. (15.24 cm) diameter circles. The edges connecting the vertices are 6 in. (15.24 cm) wide. The vertical edge is 12 in. (30.48 cm) long, and the horizontal edge is 24 in. (60.96 cm) long. The bottom of the triangle is 36 in. (91.44 cm) above the ground.

The task is scored by the number of cuts completed within the allotted 30 min. One point is awarded for each side of the triangle completed, with the third point only being scored when the cut out is completely removed from the wall. The cuts must be made completely within the green and blue areas, but there can be any number of cuts within that area.

At the start, ATLAS was standing with the Robotiq hand, configured with the correct fingertip modifications and properly initiated. At this point, ATLAS proceeded through the following steps (Figure 20):

The first step is to use the LIDAR point cloud to detect where the rotary tool is and superimpose a virtual model over it. This model serves as a marker that includes the correct foot placement for grasping the rotary tool. After reaching the correct location, we transition into manipulate mode and move the right arm to predetermined locations, leading all the way to the tool. We finalize the approach manually using end-effector nudges, checking it visually as well as with the force sensors on the wrist. Once the rotary tool is confirmed to be grasped, the arm is moved to a prescripted shake position. The left to right shaking motion as described earlier places the rotary tool in the right configuration for pressing the trigger. After each shake, we check if the trigger was pressed, and if that is the case we move on to the next phase.

Figure 21. User view from the sensor head during the cutting process.

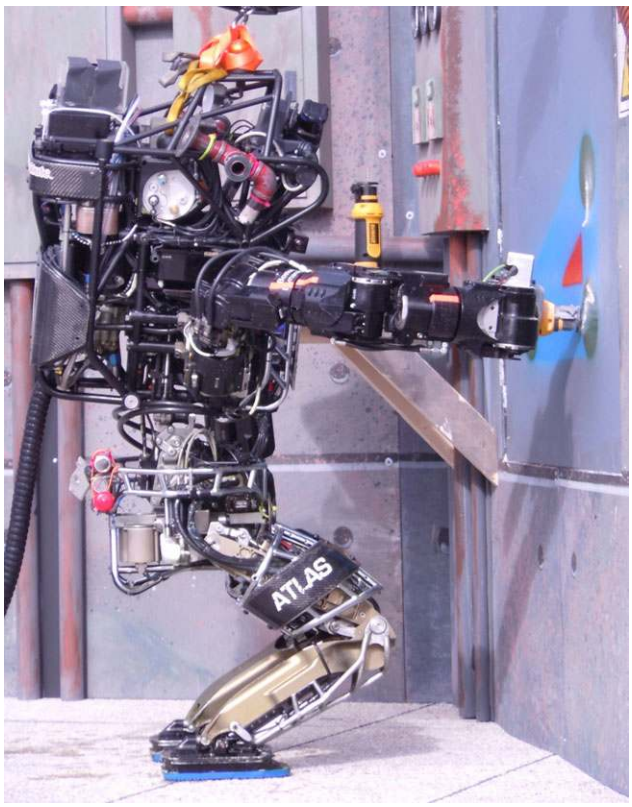

Figure 22. ATLAS attempting the wall task.

The next step is generating the steps to the wall. After these are taken, most times there is another small stepping procedure that has to occur in order to align ATLAS properly with the wall. The alignment between the wall and ATLAS is crucial in order for the robot to have the correct range of motion to cut the entire triangle. At this point the procedure is to move the end-effector close to the initial cutting position, followed by switching to the full-body controller and then starting cutting (Figure 21). The cutting is done manually using a simple nudging control, and the controller maintains the end-effector for the most part in the correct plane. Some force is applied against the drywall.

During test sessions and dress rehearsals before the DRC Trials, the wall task would consistently score points. Poor performance was only a problem during the period where the Boston Dynamics stepping behavior was failing midwalk due to a hardware problem. When all systems performed adequately, cutting the triangle out within the bounds and time limit did not prove to be difficult.

During the trials, we were not able to complete any subtasks for the wall task (Figure 22). The robot successfully picked up the rotary tool and moved to the proper location to cut the wall. After plunging into the wall and making a short cut, a previously unencountered user error caused the right arm end-effector to try and maintain a global position, which is not possible if the controller is also trying to move the left arm end-effector to the right (for cutting).

After a few unsuccessful attempts to continue cutting, the robot removed the tool from the wall with the intent to try to cut from a different starting location. Unfortunately, the robot at that point lost balance and fell backward. The three points were distributed by cutting each side of the triangle. The cuts needed to start from the circle bounding each vertex and stay within the boundaries of the lines. We failed to score any points because we did not successfully cut within the time limit.

5.7. Valve

For this task, the robot must close three industrial valves during a period of 30 min. Those consist of a 13 in. ball valve, and two gate valves with turning wheels of 9 and 18 in. The ball valve has to be closed 90° clockwise (until the lever is horizontal). The gate valves have to be rotated clockwise a full turn to be considered closed.

The valve turning task was specified at 5 lbf (22.24 N) maximum force limitation on the valves, which allowed us to focus on single arm manipulation approaches. In earlier attempts, one of the three robot hands was used for grasping the edge of the valve and turning, much like a human would do. This approach turned out to be difficult and ineffective due to the constrained arm motions of the ATLAS while holding the valve. It should be noted that each of the ATLAS' arms has 6 DOF and the control method provided by the robot separates the manipulation and balancing. As a result, we utilized our full-body controller. The full-body controller has the advantage of shifting the CoM and lower-body joints to give an extended range of motion. Furthermore, we used a 6 in. long pipe hand as the end-effector. The hand can be placed inside the gate valve, and because they are symmetrical around the centerline, the full-body controller roll constraints can be removed. The practical result of this approach is an added virtual degree of freedom to the arm.

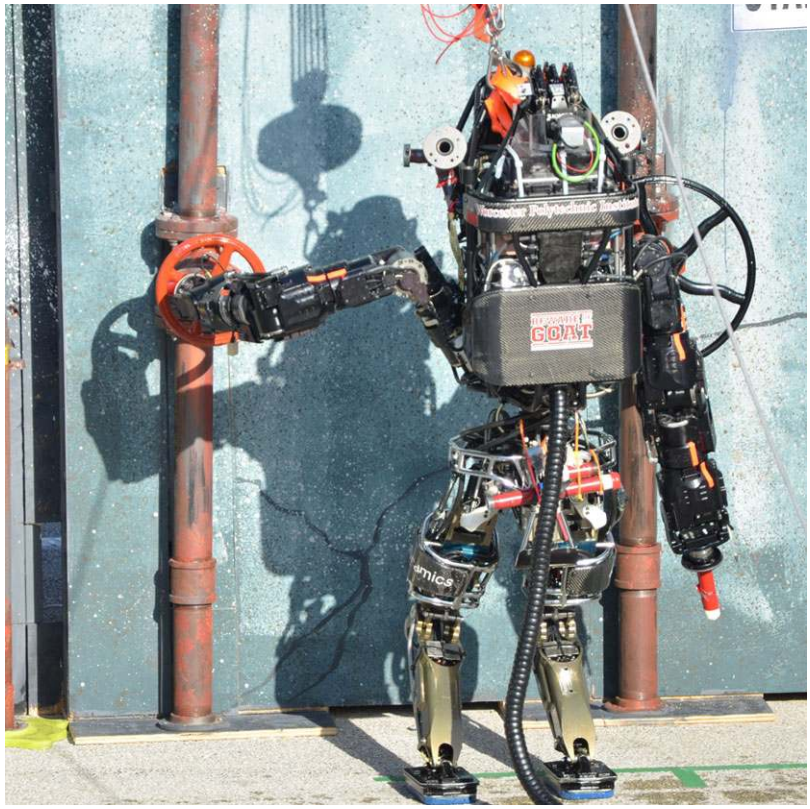

Figure 23. Valve Turning using full body controller.

The operator can position the robot arm using small end-effector motions based on user keyboard inputs, called nudges. Nudges are 1 in. translations that can be commanded by the operator, moving the end-effector in the specified direction to close the valve. An interactive visual marker, similar to the one used for the drill, is overlaid onto the valve we want to turn in three-dimensional (3D) space. The marker has an attached foot placement that positions ATLAS in an ideal position for valve manipulation. The arm is manipulated to a prescripted position with the pipe hand semialigned with the valve. We then switch to the full-body controller, and using visual feedback and nudges, we engage the valve and turn it (Figure 23).

The nudging approach has limitations compared to a more autonomous approach. Nudging is heavily dependent on the human operator, making it inconsistent between runs. It is also relatively slow. A few automated attempts showed that when successful, ATLAS was able to close a valve in 30 s or less, while nudging would take between 3 and 5 min. The three points on this task corresponded to closing the three different valves. Because we managed to close all three valves within the time limit and did not reset the robot during the run, we earned the maximum four points on this task.

5.8. Hose

This task involves locating and grasping the nozzle of a fire hose, unreeling the hose, and attaching the hose nozzle to the wye connector. The hose and wye fitting are set up in a warehouse scenario. The hose reel and wye fitting are mounted to a wall at a height approximately 1 m above the ground. The hose was positioned with the nozzle hanging down from the reel.

Our final strategy for the hose task is depicted in Figure 24 and can be described as follows: On the start line, the robot scans the whole course to allow the operator to identify the location of the hose reel and hose nozzle. To grasp the hose nozzle, the robot needed to stand in front of the reel; this allowed for extra clearance between the hose nozzle and the reel. Similar to the wall and valve tasks, an interactive marker allowed the operator to position the robot directly inside of the point cloud to manipulate the hose reel, after which the operator would select the desired step size and distance from the marker to position the final foot location. A smooth trajectory would be generated based on the stepping starting point and destination. After tweaking the desired path manually, the operator would command the robot to execute walking to the reel. Once the

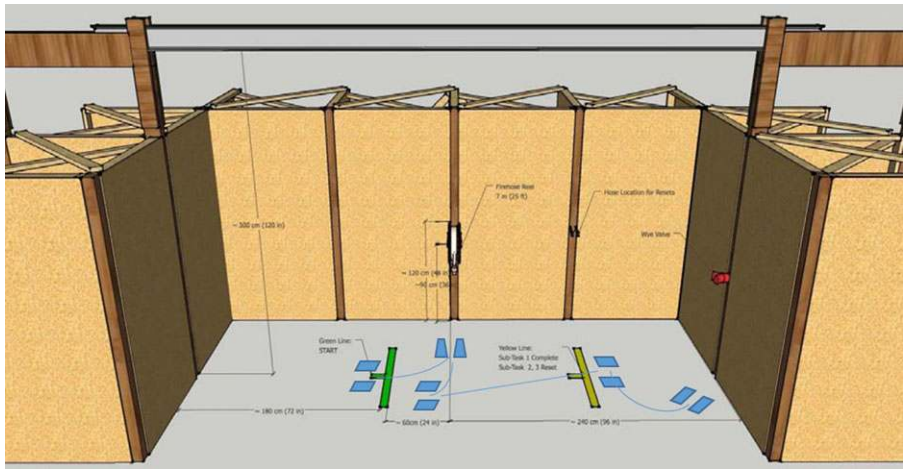

Figure 24. The hose task scenario and the robot stepping trajectory.

robot completed walking, final tweaks were done manually to align the robot.

The operator would then choose a grasping script from a predefined library generated during testing in the lab. Using visual feedback, the operator would manually move the robot's hand the last 0.1–0.2 m to grasp the hose nozzle. After the hose nozzle was within the grasp of the robot's hand, another script was activated that brought the robot's arm quickly to its side in order to break the hose reel free and have the hose unreel itself. After grasping the hose, the robot would step back and realign to help clear any potential hosing around its feet. The procedure for alignment with the wye fitting was similar to reaching the reel. The only notable difference was that the robot would approach the wye fitting with two stepping phases, a coarse and fine alignment. This was done to minimize any error that would accumulate over longer distances. To aid in attaching the hose nozzle to the wye fitting, the robot's hand was equipped with a finger extension (Figure 11). The finger extension would come into contact with the rotary collar, and while the hand rotated the hose nozzle, the finger extender ensured that the collar would also rotate.

At the DRC Trials, the operator successfully controlled the robot to pick up the hose nozzle and touch it to the wye fitting within the first 12 min of the 30 provided, earning the first two points of the task, respectively. However, in the remaining 18 min, the robot was unable to attach the hose nozzle to the wye fitting.

5.9. Points

Team WPI-CMU scored a total of 11 out of a possible 32 points. We acquired all four points in the valve task, two points each on the terrain, ladder, and hose tasks, and one point in the vehicle task. These 11 points gave us a ranking of seventh out of 16 teams. As a result, Team WPI-CMU advanced to the DRC Finals, which will take place in June 2015. The breakdown of points by task is presented in Table V.

Table V. WPI-CMU DRC Team Points Breakdown at the DRC Trials.

Task	Points	Interventions
Vehicle	1	0
Terrain	2	2
Ladder	2	1
Debris	0	2
Door	0	2
Wall	0	1
Valve	4	0
Hose	2	0
Total	11	8

6. CONCLUSION

We discussed the details of our system architecture for human-in-the-loop control of humanoid robots for disaster response tasks. We utilized varying levels of teleoperation and autonomy to complete DRC Trials tasks. The strengths and deficiencies of our approach have been identified by examining select DRC Trials tasks. Based on the strengths of, and despite the deficiencies in, our approach, the WPI-CMU team became a DRC finalist by ranking seventh out of 16 teams. The team also received the Best-in-Task: Vehicle Award by completing the first subtask in 6 min. The WPI-CMU DRC Team continues to improve the performance of the framework by incorporating new supervised autonomy and user interface tools with a goal toward meeting the requirements set forth by the DRC Finals.

ACKNOWLEDGMENTS

This work is sponsored by the Defense Advanced Research Project Agency, DARPA Robotics Challenge Program under Contract No. HR0011-14-C-0011. We also acknowledge our corporate sponsors NVIDIA and Axis Communications for providing equipment support.

REFERENCES

DARPA (2013). Drc trials rules. http://www.theroboticschallenge.org/files/DRCTrialsRulesRelease7DISTAR22157.pdf.

Alunni, N., Phillips-Grafftin, C., Suay, H. B., Lofaro, D., Berenson, D., Chernova, S., Lindeman, R. W., & Oh, P. (2013). Toward a user-guided manipulation framework for high-dof robots with limited communication. In Technologies for Practical Robot Applications (TePRA), 2013 IEEE International Conference (pp. 1–6). IEEE.

Burke, J. L., Murphy, R. R., Coovert, M. D., & Riddle, D. L. (2004). Moonlight in Miami: Field study of human-robot interaction in the context of an urban search and rescue disaster response training exercise. Human-Computer Interaction, 19(1-2), 85–116.

Casper, J., & Murphy, R. (2003). Human-robot interactions during the robot-assisted urban search and rescue response at the World Trade Center. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, 33(3), 367–385.

CRL, C. R. L. (2014). Carnegie robotics multisense sl. http://carnegierobotics.com/multisense-sl.

Feng, S., Xinjilefu, X., Huang, W., & Atkeson, C. (2013). 3d walking based on online optimization. In IEEE-RAS International Conference on Humanoid Robots.

Hirukawa, H., Hattori, S., Kajita, S., Harada, K., Kaneko, K., Kanehiro, F., Morisawa, M., & Nakaoka, S. (2007). A pattern generator of humanoid robots walking on a rough terrain. In Robotics and Automation, 2007 IEEE International Conference (pp. 2181–2187), Rome, Italy. IEEE.

Nagatani, K., Kiribayashi, S., Okada, Y., Otake, K., Yoshida, K., Tadokoro, S., Nishimura, T., Yoshida, T., Koyanagi, E., Fukushima, M., et al. (2013). Emergency response to the nuclear accident at the Fukushima Daiichi nuclear power plants using mobile rescue robots. Journal of Field Robotics, 30(1), 44–63.

Nourbakhsh, I., Sycara, K., Koes, M., Yong, M., Lewis, M., & Burion, S. (2005). Human-robot teaming for search and rescue. Pervasive Computing, 4(1), 72–79.

Pratt, G., & Manzo, J. (2013). The darpa robotics challenge [competitions]. IEEE Robotics Automation Magazine, 20(2), 10–12.

Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R., & Ng, A. Y. (2009). Ros: An open-source robot operating system. In ICRA workshop on open source software (vol. 3, p. 5), Kobe, Japan. IEEE.

Rasmussen, C., Yuvraj, K., Vallett, R., Sohn, K., & Oh, P. (2013). Towards functional labeling of utility vehicle point clouds for humanoid driving. In Technologies for Practical Robot Applications (TePRA), 2013 IEEE International Conference (pp. 1–6).

Robotiq (2014). Robotiq adaptive gripper specification sheet http://robotiq.com/products/industrial-robot-hand/.

Sentis, L., Park, J., & Khatib, O. (2010). Compliant control of multicontact and center-of-mass behaviors in humanoid robots. IEEE Transactions on Robotics, 26(3), 483–501.

Stephens, B. (2011). Push recovery control for force-controlled humanoid robots. Ph.D. thesis, The Robotics Institute, Carnegie Mellon University, Pittsburgh.

Stephens, B., & Atkeson, C. (2010). Dynamic balance force control for compliant humanoid robots. In Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference (pp. 1248–1255).

Vahrenkamp, N., Berenson, D., Asfour, T., Kuffner, J., & Dillmann, R. (2009). Humanoid motion planning for dual-arm manipulation and re-grasping tasks. In Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference (pp. 2464–2470).

Whitman, E. (2013). Coordination of multiple dynamic programming policies for control of bipedal walking. Ph.D. thesis, The Robotics Institute, Carnegie Mellon University, Pittsburgh.