This paper presents the technical approaches used and experimental results obtained by Team SNU (Seoul National University) at the 2015 DARPA Robotics Challenge (DRC) Finals. Team SNU is one of the newly qualified teams, unlike 12 teams who previously participated in the December 2013 DRC Trials. The hardware platform THORMANG, which we used, has been developed by ROBOTIS. THORMANG is one of the smallest robots at the DRC Finals. Based on this platform, we focused on developing software architecture and controllers in order to perform complex tasks in disaster response situations and modifying hardware modules to maximize manipulability. Ensuring stability and modularization are two main keywords in the technical approaches of the architecture. We designed our interface and controllers to achieve a higher robustness level against disaster situations. Moreover, we concentrated on developing our software architecture by integrating a number of modules to eliminate software system complexity and programming errors. With these efforts on the hardware and software, we successfully finished the competition without falling, and we ranked 12th out of 23 teams. This paper is concluded with a number of lessons learned by analyzing the 2015 DRC Finals. © 2016 Wiley Periodicals, Inc.
Table I. 2015 DRC Finals Team Standings.

<table>
<thead>
<tr>
<th>Team</th>
<th>Score</th>
<th>Time</th>
<th>Trials</th>
<th>Team</th>
<th>Scores</th>
<th>Time</th>
<th>Trials</th>
</tr>
</thead>
<tbody>
<tr>
<td>KAIST</td>
<td>8</td>
<td>44:28</td>
<td>8</td>
<td>THOR</td>
<td>3</td>
<td>27:47</td>
<td>8</td>
</tr>
<tr>
<td>IHMC</td>
<td>8</td>
<td>50:26</td>
<td>20</td>
<td>HRP2-TOYKO</td>
<td>3</td>
<td>30:06</td>
<td>-</td>
</tr>
<tr>
<td>TARTAN RESCUE</td>
<td>8</td>
<td>55:15</td>
<td>18</td>
<td>ROBOTIS</td>
<td>3</td>
<td>30:23</td>
<td>-</td>
</tr>
<tr>
<td>NIMBRO RESCUE</td>
<td>7</td>
<td>34:00</td>
<td>-</td>
<td>VIGIR</td>
<td>3</td>
<td>48:49</td>
<td>8</td>
</tr>
<tr>
<td>ROBOSIMIAN</td>
<td>7</td>
<td>47:59</td>
<td>14</td>
<td>WALK-MAN</td>
<td>2</td>
<td>36:35</td>
<td>-</td>
</tr>
<tr>
<td>MIT</td>
<td>7</td>
<td>50:25</td>
<td>16</td>
<td>TROOPER</td>
<td>2</td>
<td>42:32</td>
<td>9</td>
</tr>
<tr>
<td>WPI-CMU</td>
<td>7</td>
<td>56:06</td>
<td>11</td>
<td>HECTOR</td>
<td>1</td>
<td>02:44</td>
<td>-</td>
</tr>
<tr>
<td>DRC-HUBO AT UNLV</td>
<td>6</td>
<td>57:41</td>
<td>3</td>
<td>VALOR</td>
<td>0</td>
<td>00:00</td>
<td>-</td>
</tr>
<tr>
<td>TRAC LABS</td>
<td>5</td>
<td>49:00</td>
<td>11</td>
<td>AERO</td>
<td>0</td>
<td>00:00</td>
<td>-</td>
</tr>
<tr>
<td>AIST-NEDO</td>
<td>5</td>
<td>52:30</td>
<td>-</td>
<td>GRIT</td>
<td>0</td>
<td>00:00</td>
<td>-</td>
</tr>
<tr>
<td>NEDO-JSK</td>
<td>4</td>
<td>58:39</td>
<td>-</td>
<td>HKU</td>
<td>0</td>
<td>00:00</td>
<td>3</td>
</tr>
<tr>
<td>SNU</td>
<td>4</td>
<td>59:33</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*ahttp://www.teamhector.de/

by the U.S. Defense Advanced Research Projects Agency (DARPA), is a competition involving robot systems developed to provide assistance in natural and man-made disasters. Ever since the first event of the DRC, called the Virtual Robotics Challenge (VRC), which was held in June of 2013, the DRC has encouraged many researchers to develop hardware and software architectures that operate in complex environments.

The DRC was motivated by the Fukushima Daiichi nuclear disaster, which occurred in March, 2011. At the time, robots were unable to stop radioactive material leakage because they could not recognize the target and manipulate power tools in the darkened reactor building. The 2015 DRC Finals comprised various tasks designed to test a robot’s capability to overcome difficulties in disastrous situations. These tasks included the following: Steering a vehicle through an obstacle course, safely egressing a vehicle, opening a door, handling a valve, cutting a hole using a power tool, walking over a pile of rubble, and ascending a short flight of stairs. Unlike the 2013 DRC Trials, the robot needed to complete eight tasks without human intervention within an hour.

The main challenges facing the teams in their efforts to succeed in the missions at the DRC Finals are as follows:¹

- perception: ability to recognize the environment around the robot.
- mounted mobility: mobility to maneuver a vehicle safely.
- dexterity: ability to manipulate the diverse tools designed for humans.
- decision-making: ability to be operated by an autonomous algorithm or by humans who have little robotics training in degraded communication.
- dismounted mobility: mobility to traverse the rough terrain.
- strength: ability to endure the weight of diverse tools and clear away debris.
- degraded communication: software architecture to operate the robot intuitively using limited information.

Table I shows that Team SNU (Seoul National University) is one of 11 newly qualified teams that participated in the 2015 DRC Finals, along with 12 teams that acquired finalist status in the December 2013 DRC Trials. Team SNU ranked 12th at the DRC Finals, despite a lack of experience and short developing time.

We used the hardware THORMANG (ROBOTIS, Co., Ltd., South Korea), which is an upgraded THOR-OP model (Yi et al., 2015). Although Team SNU was one of only a few software-based teams in the DRC Finals, we concentrated both on developing the software architecture and the controller, and modifying the hardware modules to complete the challenge. These balanced developments between hardware and software helped to maximize our robot’s performance in complex tasks. For example, turning on the drill during the wall tasks is one of the most difficult challenges in the DRC Finals, because the robot’s perception system often cannot find the drill switch. We solved this problem by designing a special end-effector with a passive palm to push the button when the hand closes. In another example, we established a strategy in which the robot was able to ascend the stairs by grasping the rail using an end-effector. Not only did this strategy increase the robot’s stability, it also decreased its power consumption.

On the other hand, our software system had two main characteristics. First, ensuring the robot’s stability was considered to be the first priority in our developments. Surviving falling incidents in a disaster area is very difficult. In fact,
none of the biped robots performed the “getting up” task all by themselves when they fell down in the DRC Finals. Some teams, including Team AIST-NEDO, Team THOR, and Team ROBOTIS, withdrew from the competition because of hardware malfunction caused by impact from the field (Guizzo & Ackerman, 2015; Kaneko et al., 2015; McGill, Yi, & Lee, 2015). Team SNU, however, successfully finished the DRC Finals because we concentrated on developing our interface and controller considering the robot’s stability as the top priority. For instance, to increase stability during biped walking, the Double Support Phase (DSP) is changed automatically by regarding the status of the robot using inertial measurement unit (IMU) and force-torque (FT) sensors.

Second, modularization was necessary to develop the system efficiently. Our hardware platform, THORMANG, is a modular humanoid robot that can be easily modified to complete tasks. Similar to the hardware platform, we developed a software architecture comprising various modules. Therefore, our software engineers developed modules regardless of the developing environments and programming languages. Also, modularization was really helpful in eliminating programming errors because our developers did not have to consider the whole architecture system anymore. With this approach, our team placed not only in the upper ranks of 11 newly qualified teams, but also in the top rank of 4 teams who used the THORMANG platform despite the lack of developing time, as shown in Table I.

A brief overview of our system and results was presented by Kim et al. (2015). In addition, the detailed strategy only for the wall mission and its analysis were described by Park et al. (2015). Therefore, in this paper, we focus on our unique developments, our strategies, and what we learned from the DRC.

This paper is organized as follows. Section 2 introduces an overview of Team SNU’s architecture, including hardware and software. Sections 3 and 4 present unique approaches in the perception system and the walking controller for stability, respectively. Next, we discuss manipulation and communication systems in Sections 5 and 6. Section 7 discusses the results of the 2015 DRC Finals as well as those of our lab tests. Finally, Section 8 summarizes a few points about what we learned from the 2015 DRC Finals, and the paper is concluded in Section 9.

2. SYSTEM ARCHITECTURE OVERVIEW

This section presents the specifications of THORMANG and the details of our technical approaches for the software architecture. As shown in Figure 1(a), THORMANG’s design is based on THOR-OP, which was used in the 2013 DRC Trials as a hardware platform of Team THOR. The representative characteristic of the THORMANG platform is modularity, i.e., strategically replacing any component on purpose. In fact, although four teams (Team SNU, Team THOR, Team ROBOTIS, and Team Hector) used the THORMANG platform in the DRC Finals, the shapes and sizes of the robots are different, as shown in Figure 1. In particular, we attached the following two special modules: the iris camera module to expand THORMANG’s view, and the end-effector module with a passive palm to enhance the grasping robustness.

Also, we designed the software architecture comprising independent modules. In particular, the field computers

Figure 1. THORMANG platform at the DRC Finals: (a) THORMANG of Team SNU, (b) THORMANG2 of Team ROBOTIS, (c) Johnny05 of Team Hector, and (d) THOR-RD of Team THOR.
between the computers in the robot and the computers for the operators perform complex calculations to overcome the low communication line bandwidth by handling the raw data of the robot’s computers. The operators are able to select data obtained by the field computers depending on THORMANG’s tasks.

2.1. Hardware Architecture

THORMANG consists of 32 actuators (except the actuators for LiDARs): eight in each arm (including the gripper), six in each leg, two in the torso, and two in the head (Figure 2(a)). Every actuator is a Dynamixel PRO developed by ROBOTIS for commercial products. The height, weight, and wingspan of THORMANG are 1.47 m, 60 kg, and 1.95 m, respectively.

THORMANG has two computers called the Robot Control Unit (RCU) and the Robot Perception Unit (RPU) to individually manage robot control and robot perception. These computers have an Intel i5 2.7 GHz quad core and 8 GB DDR3 RAM. Figure 2(b) shows the communication between the actuators and the computer comprising four RS-485 channels. Each RS-485 channel manages the actuators in each limb at 100 Hz. We use the following three LiPo batteries to operate THORMANG during the competition: a 5-cell LiPo battery (11,000 mAh, 18.5 V) for the onboard computers, and two 6-cell LiPo batteries (22,000 mAh, 22.2 V) for operating the actuators.

2.1.1. Sensory Components

The original version of THORMANG used webcams, LiDARs, an IMU sensor, and FT sensors as sensory components. In addition to these sensors, we used another module with an iris camera for the driving mission. The iris camera has a wide field-of-view (FOV) and an adjustable aperture. Hence, the robot can cover the front view of THORMANG for the driving task without limited visibility caused by sunlight intensity. The specifications of each sensor are as follows.

- **three webcams (Logitech C905)**: these webcams are located in front of the head in a row to monitor the wide front of THORMANG. Each webcam has 480×640 resolution and 45° FOV. We also obtained sounds around THORMANG using microphones on the webcams. We used these sounds to operate the robot when the operator could not secure a clear view in a degraded communication.
- **two LiDARs (Hokuyo UTM-30LX-EW LiDAR)**: one LiDAR in the chest is used to recognize the target (e.g., valves, door knobs, and drills) position and orientation. This LiDAR is attached on a panning servo motor with a range of ±45°. The other LiDAR, with a rolling servo motor, is located in the front of the head. The main goal of the head LiDAR is to draw an approximated 2D map surrounding for THORMANG in degraded communication. In this case, the rolling motor is not used. The head LiDAR is operated to gather only one layer of raw data, which is parallel to the horizontal ground surface.
- **an IMU sensor (Microstrain 3DM-GX4-45)**: the accelerations and angular velocities of the pelvis can be measured because of the IMU sensor with a three-axis accelerometer and a gyro sensor. A complementary filter is implemented to estimate the pelvis orientation.
- **two FT sensors (ATI Mini 58)**: the FT sensor on foot measures the reaction force between the environment and the robot’s foot. The FT sensor is connected to an analog-to-digital converter (ADC), which has been developed by ROBOTIS, instead of a standard product, because of the lack of space for installation. However, the FT sensor value is noisy because of the small measurement range of the ADC provided by ROBOTIS. Therefore, we used a low-pass filter for noise reduction.
- **an iris camera (Imagingsource DFK 23G618.I)**: the iris camera has an adjustable aperture for controlling the amount of light. The resolution of an iris camera with 120° FOV is 640×480.

2.1.2. End-effectors

Two types of end-effectors from ROBOTIS were provided for grasping: an end-effector with two sticks and an end-effector with underactuated fingers. Figure 3(a) shows the end-effector with two sticks and one actuator. This gripper has a traditional parallel mechanism to firmly grasp an object. The other end-effector has passive joints with a spring-loaded linkage, as shown in Figure 3(b). This end-effector can wrap the object in its fingers, although there is one actuator in the end-effector (Rouleau, 2015; Rouleau & Hong, 2014). However, these end-effectors do not allow...
Figure 3. Original grippers for THORMANG: (a) gripper with two sticks, (b) gripper with underactuated fingers.

Figure 4. Overview of our software architecture: (a) diagram of our software architecture, (b) finite State Machine for the docking mission in the wall task.

Figure 3. Original grippers for THORMANG: (a) gripper with two sticks, (b) gripper with underactuated fingers.

Figure 4. Overview of our software architecture: (a) diagram of our software architecture, (b) finite State Machine for the docking mission in the wall task.

precise grasping to turn on the drill. Thus, we designed a new end-effector that would provide a grasping skill for switching on a drill by modifying the gripper with underactuated fingers. Section 5.2 describes the design concept and performance of this end-effector.

2.2. Software Architecture

Figure 4 illustrates the software system design, which uses seven computers containing various independent modules: Two computers in the robot (as mentioned in Section 2.1), two computers as field servers, and three computers for operation. The subsections that follow briefly describe the background of the communications system by the DARPA and the configuration details for software architecture.

2.2.1. Overview of DRC Finals Communications System

The DARPA recommends the following three types of computer units to be used in constructing the software architecture: a Robot Computer Unit, a Field Computer Unit (FCU), and an Operator Computer Unit (OCU). First, the robot computer unit (e.g., RCU and RPU) in our robot directly manages the hardware. Second, the FCU handles communication between the robot and the operator. Third, the OCU in the operator control station has an interface for robot control.

Compared with the DRC Trials, one of the main difficulties during the DRC Finals is the wireless communications link, which is degraded and periodically interrupted. Three wireless lines are used for communications among the robot computer units, FCUs, and OCUs. First, the robot computer unit communicates with the FCU over a wireless LAN called Link1. Link1 is an always-on bidirectional communication line with a 300 Mbit/s bandwidth. Second, the OCU communicates with the FCU through Link2 and Link3 using a Degraded Communications Emulator (DCE). Link2 is a unidirectional UDP line, which supports 300 Mbit/s. However, this link provides 1-s bursts interspersed with blackouts while the robot is performing indoor tasks (i.e., valve, wall, and surprise tasks). The blackouts can last from 1 to 30 s. Link3 is an always-on bidirectional TCP or UDP line between the OCU and the FCU. This link supports a small constant data rate of 9,600 bit/s.

2.2.2. Robot Computer Units

There are two robot computer units, as shown in Figure 4(a). First, the RCU handles the actuators responsible for moving the robot, FT sensors on foot, and an IMU sensor in the torso. The RCU comprises various modules, including the device manager for actuators and sensors, robot controller, and simulators. Moreover, we used IntervalZero RTX library to implement a real-time system. Using the RTX, one of the four threads in the CPU of the RCU is used to ensure real-time robot control. Second, the RPU is in charge of THORMANG’s perception system, including vision sensors and LiDARs. The RPU can send the raw data obtained from these sensors to the field server by communicating through Link1.

We use two simulators in the RCU, namely V-Rep and RoboticsLab, to cross-validate our motion planners and controllers. V-Rep is based on the Vortex physics engine producing high-fidelity physics simulations. V-Rep offers real-world parameters, including resultant force and a non-linear friction model, which make the simulator realistic and precise (Rohmer, Singh, & Freese, 2013). Meanwhile, the RoboticsLab has high calculation speed in the virtual world (Yi, 2008). It also provides the SDK to implement rigid body kinematics and dynamics. Finally, every module located on RCU and RPU could be operated to control the robot in a
virtual environment, because both V-Rep and RoboticsLab support remote API to customize the simulator.

Meanwhile, every module in the RCU and the RPU is accessed by an event-based Finite State Machine (FSM) for each task to control the robot. Each FSM has various subtasks, including perception, control for end-effectors, and mobility. To construct an efficient semiautonomy system, we divided the subtasks into two groups, depending on the subject of the action: behavior by the robot alone and behavior by human-robot interaction. For example, Figure 4(b) shows the FSM for the docking task in order for the drill to be grasped by the robot. Ready is the subtask in which the robot performs ready poses for reaching the drill, which is recognized by an object recognizer. Thus, the robot performs the predefined posture without the judgment of the operators. In contrast, Grasping is the subtask that is performed by human-robot interaction. During the wall mission, the robot cannot recognize whether the drill is turned on or not. Thus, the operators can supervise the robot’s behavior if our robot fails to turn on the drill by checking the data of the sound and vision sensors.

2.2.3. Field Computer Units

In the field server there are two FCUs: one for control and the other for perception. Our FCUs are in the middle of the communications between the robot computer units and the OCUs. Our FCUs share the raw data of the robot’s sensors through Link1, and they send these data to the operators by communicating through Link2 and Link3. The FCUs handle complex calculation, such as object recognition and compressing perception data to send to the operators. Hence, this system reduces the power consumption of THORMANG and decreases data size from the robot to the operators.

2.2.4. Operation Computer Units

Prior studies of the DRC Trials have reported that the complex User Interface (UI) not only increases unknown programming errors, but it also decreases operation performance (Fallon et al., 2015; Johnson et al., 2015; Stentz et al., 2015). Accordingly, we consider some issues for UI implementation in human-robot interaction. First, there are a number of controllers and information obtained from the robot’s sensors. Second, degraded communications interrupt the direct interaction between the operators and the robot. Third, although the multioperator interface can enhance the human-robot interaction (Burke & Murphy, 2004), this system often disturbs the communication between the operators. We address these issues by developing the main operation tool (OCU1) and two operation tools (OCU2 and OCU3) that would support OCU1, as shown in Figure 5.

First, OCU1 is designed such that the primary operator alone can control the robot and understand the current status under poor communication conditions. OCU1 handles necessary data for operating the robot, including current joint status, FT and IMU sensor values, resized images and
S. Kim et al.: Team SNU’s Control Strategies to Enhancing Robot’s Capability

Figure 6. Viewer in OCU3 at driving mission practice: (a) raw image from iris camera, (b) bird’s-eye view, (c) real view.

sounds from the multimedia converter, and 2D map data from the head LiDAR, by communicating through Link3. These data are treated in degraded communication by letting the operator choose which data he will receive among the 2D map image viewer, sound listener, and low-quality image viewer depending on the situation. For instance, the operator chooses the sound data to recognize whether the drill is turned on or not, and the low-quality image to judge the distance between the robot and the wall during the wall task.

Second, OCU2 based on ROS\(^8\) is specialized to present data from the webcams and LiDARs. The robot viewer based on Rviz\(^9\) displays point cloud data (PCD) from the chest LiDAR and the current posture of the robot from OCU1 in the virtual environment. Therefore, the operator for OCU2 can select a target point on the PCD using an interactive marker when the object recognizer fails to find a target.

Third, OCU3 is specialized to provide the relationship between the robot and the vehicle for the driving task. OCU3 has a device manager that handles a steering wheel and a pedal for racing games to intuitively control the robot, and a bird’s-eye viewer that provides the predicted path and an elevated view of the ground from above. Figure 6 shows a bird’s-eye view transferred from a raw image and a real top view during the parking task practice.

3. PERCEPTION

The perception system in the FCU is divided into two functional modules: the object recognizer and the multimedia converter for the multimedia data size reduction. First, the object recognizer is in charge of an automatic perception to find the location of objects, such as valves, drills, and door knobs. Second, the multimedia converter reduces the sizes of the raw streaming data obtained from the sensory compo-

\(^8\)http://www.ros.org
\(^9\)http://wiki.ros.org/rviz

ments. To overcome degraded communication as mentioned in Section 2.2.1, the multimedia converter compresses the image and the sound obtained from the robot and sends resized data to the operators through Link3.

3.1. Object Recognizer

Automatic perception is the process of finding objects by extracting features from the sensory data and performing classifications based on the extracted features. Therefore, the performance of automatic perception is affected not only by algorithms for object detection but also by the target’s shapes. We experimentally used different algorithms depending on the targets. We implemented RANdom SAmple Consensus (RANSAC) (Fischler & Bolles, 1981), Histograms of Oriented Gradients (HOG) (Dalal & Triggs, 2005), and Scale Invariant Feature Transform (SIFT) (Lowe, 2004) for the door, valve, and wall tasks, respectively. In particular, we established a unique strategy for the valve task based on the HOG features obtained from a 2D depth image, which is acquired by projecting PCD onto a 2D plane, instead of an RGB color image. This strategy was inspired by the fact that the gradient information of the depth map can be more effective than that of the color image in discriminating textureless objects, such as the valve from the background. Therefore, the proposed method has an advantage in terms of computation time and precision, because there is no process for aligning the RGB image and depth map.

We evaluated the proposed strategy in a laboratory environment before applying it to our robot. At first, valve images were captured by varying the camera angle and distance from the value to obtain 1,279 pairs of color images with 960×540 pixel size and depth images with 512×424 pixel size. We randomly selected 640 images to train our detector. The other images were used for testing. Training and test images were randomly separated five times to avoid bias from the previous random selection. Given training and test images, the training examples were collected to train a classifier from 80% of 640 training images. Initial
Algorithm 1: Recognition for the valve task.

Data: Cloud data $X_L$

Result: Object center $C$ and Object distance from robot $D$

begin
    filtered data $\hat{X}_L = \text{bilateral filter}(X_L)$;
    subtract planar structure;
    for $S_i \rightarrow S_1, S_2, S_3, \ldots$ do
        find planar structure $S_i$ using RANSAC;
        $S_{i+1} = \hat{X}_L - S_i$;
        if Number of Point Cloud $S_i \leq \theta$ then
            break;
        end
    end
    find valve region using SVM trained with HOG features;
    compute center $C$ and distance $D$;
end

training samples comprising 512 positive samples, which correspond to 50×50 image patches surrounding the valve, were cropped from each depth image. About 1,000 negative samples were also randomly cropped with the same size. A linear SVM (Smola & Schölkopf, 1998) was then trained using the HOG feature vectors computed from the training samples. We conducted the bootstrapping twice after training the SVM by scanning the remaining 20% of the training images, where the training samples were not collected. With the trained SVM, we collected false positive examples and retrained the SVM in using a new training set containing the collected false positive examples as negative. It is well known that collecting the false positive examples and retraining the SVM using the newly collected examples significantly improves the SVM performance (Dalal & Triggs, 2005). Meanwhile, we considered two other detectors based on the scanning window for comparison. One of these detectors is the RGB+HOG corresponding to Dalal & Triggs (2005), and the other is the SIFT+BoF, whose SIFT points and their descriptors are computed from each local patch, and a feature vector of a fixed size is produced based on the Bag-Of-Features (Csurka, Dance, Fan, Willamowski, & Bray, 2004) with respect to each local patch. The two detectors were trained by almost identically performing the previous procedures for the proposed Depth+HOG. The only difference is the base resolution, which comes out as 70×70 pixels rather than 50×50 pixel in the two methods because the color image is larger than the depth image with respect to the same scene. The detection process using the three detectors is identical. The HOG or SIFT+BoF

Figure 7. Results of detected valve at various perspectives: (a) average recall of the proposed algorithm, (b) average precision of the proposed algorithm, (c) illustration of valve recognition.
feature vectors were computed from all regions in the depth or RGB images, and each feature vector was classified by the trained SVM whether each region contains the target valve or not. Nonmaximum suppression (Dalal & Triggs, 2005) was then applied to combine the multiple detected windows into a single window around the target valve. Figures 7(a) and 7(b) show the average recalls and precisions of the three detectors, respectively. The recalls and precisions of four cases were measured where the cases were divided by the angles between the robot’s front and the wall. The notations, set01 to set04, mean 0–20, 20–40, 40–60, and 60–90 deg, respectively. The proposed Depth + HOG algorithm provided higher recall and precision by over 10% than the other detectors. This result indicates that the valve can be more effectively detected by computing the HOG features based on the depth image rather than the RGB image.

The application of the proposed strategy to our robot is described in the discussion that follows. Assuming that the operator places our robot in front of the target valve, it can be observed in a depth image of 314 × 600 pixels, as shown in Figure 7(c). After smoothing using a bilateral filter to remove jitter noise in the raw depth data, the RANSAC-based plane detection is applied to the filtered depth image to extract the plane, which corresponds to the wall behind our target valve. Subsequently, a valve region is detected as previously explained. We could fix the size of the scanning region as 40 × 40 pixels because the distance variations between the robot and the target valve were not large when capturing the depth data to detect the valve. The detected region, which surrounds the green pixels in the depth image shown in Figure 7(c), is verified by comparing the real and estimated valves sizes. We finally considered the target valve to be detected if the difference was not larger than a threshold. The valve’s center location and the distance between the valve and the robot are computed as the mean location and the mean depth of the pixels in the detected region, respectively. Algorithm 1 summarizes these procedures.

Training the SVM is important in achieving a reliable detection with a very high successful rate. The SVM should be trained to detect the target valve under the real environment with the angle and scale variations between the robot and the valve. Accordingly, we gathered positive training examples by varying the viewpoints by 5 deg in the range of 150 deg from the valve center. The positive examples were collected with slight scale variations to make our SVM robust to scale variations. Our initial training set comprised 1,500 positive examples and 6,000 negative examples, which were randomly collected. The SVM was trained using the training examples. We also conducted bootstrapping twice after SVM training.

3.2 Multimedia Processing

Sharing views and sound around the robot helps the operator intuitively perceive the robot status. However, raw streaming data from the sensors of the robot could not be directly sent to the operator, because of the small constant data rate of Link3. Therefore, the multimedia converter in the FCU for perception is to compress the image and sound obtained from the webcams. For image data, the raw JPEG image with 480 × 640 pixels is converted to 30 × 40 pixels, as shown in Figures 8(a) and 8(b). The converted image size is about 1 kB, and it can be sent at 1 Hz. Moreover, the sound from the webcams is refined as a 16 kbps MP3 file. The multimedia converter then deletes the upper register of the refined sound because the operators can successfully understand whether the drill was turned on or not by hearing only the lower register of the sound.

4. WALKING

This section presents our locomotion controllers. The locomotion control scheme basically consists of three parts: high-level planning with footstep planning, low-level planning including preview control for the center of mass (COM) trajectory generation, and the robot control with inverse kinematics, as shown in Figure 9.

However, there are several issues that have a negative effect on bipedal walking stability. First, joint elasticity causes unmeasurable deformation. Second, performing continuous dynamic walking is hard because of the limited actuator performance and low response time of sensors by a frequency of 100 Hz. Third, the torque limits of the actuators and the short leg length mean that the robot is in no condition to ascend the stairs. The following subsections describe how each issue affects the walking stability and how we have overcome these limitations.

4.1 Joint Elasticity

The internal structure of an electric actuator module comprises two parts: a motor with an encoder and a
Figure 9. Schematic structure for locomotion control.

A speed reducer with several parts that can amplify the output torque. Since the encoder is directly attached to the motor, it only measures the input of the speed reducer, not the output. However, there is elasticity between the input and the output shafts in the speed reducer because the reducer (e.g., Harmonic Drive or cycloid) is not completely rigid (Legnani & Faglia, 1992; Sensinger & Lipsey, 2012). The actual joint angle cannot be accurately measured because of this elasticity on the reducer. Therefore, this joint elasticity can create problems with the robot’s balancing and the performance of the foot trajectory tracking in the specific case of humanoid walking.

Dynamixel, which is the actuator module in THORMANG, has joint elasticity. To figure out the joint elasticity of Dynamixel, we measured the joint deflection by changing the load weight. Figure 10(a) shows the error between the actual joint and measured angles from the encoder. With the joint elasticity, the pelvis of THORMANG is tilted toward the direction of gravity from the horizontal line, whereas that of the simulated robot perfectly maintains its pelvis orientation, as shown in Figure 10(b). This result is obtained because the simulation of the robot does not include the joint compliance. If this difference between the actual and desired postures of the real robot is large, the swing foot cannot be exactly controlled. Furthermore, the robot’s base frame cannot be accurately estimated.

The specific problem of compensating for a tilted pelvis angle has been investigated in previous works. Laser range finders (LRFs) and vision cameras are attached to a hip joint to measure its deflection angle (Oda & Nakane, 2008; Oda & Yoneda, 2013). In other research, torque sensors are used to compensate for the deflections of each leg joint (Johnson et al., 2015). However, these approaches require additional sensors for the deflection measurement.

Therefore, we designed the compensator for joint elasticity to maintain pelvis orientation accurately by using only the attached IMU sensor in the pelvis, as follows:

\[ q_{\text{hip, re}} = q_{\text{hip, d}} + q_{\text{comp}} \]  

with

\[ q_{\text{comp}} = \begin{cases} 
q_{\text{off}} + k_p \theta & \text{during the single support phase (SSP)}, \\
0 & \text{otherwise}, 
\end{cases} \]

where \( q_{\text{hip, re}}, q_{\text{hip, d}}, q_{\text{comp}}, q_{\text{off}}, \) and \( k_p \) are the final desired angles, the initial desired angle of the hip joints, the compensation angle for the tilted angle, the constant offset angle during SSP, and compensator’s gain, respectively. \( \theta \) means

Figure 10. Joint elasticity: (a) relationship between load torque and deformation angle, (b) comparison of posture between the commanded posture and the actual posture, when the robot raised the right leg.
S. Kim et al.: Team SNU’s Control Strategies to Enhancing Robot’s Capability

Figure 11. Compensator for joint elasticity: (a) tilted angle by link elasticity and the compensated result, (b) actual foot position by joint elasticity and the compensated result.

Figure 12. Walking pattern generation with modified preview controller and adjustable DSP timer.

the error between the desired pelvis angle and the measured angle determined by the IMU sensor. $k_p$ was tuned by walking experiments ($k_p = 3.5$: standard walking, $k_p = 5.5$: walking with the drill).

We validated the performance of our compensator by measuring the tilted pelvis angle with and without the compensator using a motion capture studio.\(^\text{10}\) The compensator result is denoted by the red line in Figure 11(a). The blue line in the same figure denotes the measured roll-angle of the pelvis by elasticity without any compensator while the robot raises its leg. The error of 2.4 deg cannot be ignored because this results in an error of 0.03 m on the foot location, as shown in Figure 11(b). In contrast, with our compensator, our robot could reduce the error by the joint elasticity.

4.2. Walking Pattern Generation

As shown in Figure 12, the main purpose of the low-level planning is to generate a desired COM from a reference zero-moment point (ZMP) plan. The following two factors have to be considered in generating a reference ZMP: the walking period and a portion of the DSP. A reference ZMP for dynamic walking generally has a short walking period and a small portion of the DSP.

However, we designed the DSP duration to constitute a substantial portion of the reference ZMP trajectory to overcome hardware-related issues, such as the limited performance of the actuators and the response time of the FT and IMU sensors. We also designed an adjustable DSP timer to recover robot stability. The adjustable DSP timer increases DSP duration in terms of the robot’s balancing state because our robot often could not recover its stability during walking. We therefore established a strategy wherein the DSP duration was adjustable in the balancing state. For example, if the IMU sensor’s value is larger than the threshold, the robot maintains DSP until the value is found below the threshold. The threshold is obtained by experiments.

To calculate the desired COM trajectory, we used a preview control that is the online pattern generation method with a Linear Inverted Pendulum Model (LIPM), reference ZMP plan, and current robot status (Kajita et al., 2003). However, the basic preview control algorithm with a ZMP-based LIPM has a limitation when a disturbance occurs or a reference ZMP trajectory suddenly changes by the adjustable DSP timer. To solve this problem, we used a modified preview controller (Nishiwaki & Kagami, 2011). The algorithm modifies the reference ZMP trajectory by considering the current COM status and the permissible ZMP region. Figure 12 shows the desired ZMP and COM trajectory in the lateral direction by generating the modified preview controller and adjustable DSP timer.

4.3. Walking on Unknown Ground

An uneven ground surface can cause bipedal walking instability. Therefore, we developed two types of sensor feedback controllers with FT and IMU sensors to increase stability on uneven grounds. First, the impedance controller (Kim, Park, & Oh, 2007) is used to modify the vertical motion, roll, and pitch components of the swing foot trajectory as follows:

$$X(s) = \frac{F(s)}{m.s^2 + d.s + k_r},$$

where $X(s)$, $m_r$, $d_r$, and $k_r$ are the displacement between the desired and the actual trajectory, the equivalent mass, the damping, and the stiffness between the foot and the pelvis of the robot, respectively. Additionally, the estimated

\(^{10}\)Nexus 1.8.1 and T160 cameras.

contact force, $F_e(s)$, is calculated by (Komati, Clévy, & Lutz, 2014)

$$F_e(s) = \frac{d_e s + k_e}{m_e s^2 + d_e s + k_e} F_m(s),$$

where $F_e(s)$, $F_m(s)$, $m_e$, $d_e$, and $k_e$ are the estimated contact force on the ground, the measured force of a FT sensor, the equivalent mass of the foot, the damping, and the stiffness, respectively.

For the impedance control, the parameters are determined as values that do not disturb standard walking. In our experiments, $m_e$, $d_e$, $k_e$, $m_e$, $d_e$, and $k_e$ for absorbing the impact force are set to 50 kg, 3,000 Ns/m, 0.2 N/m, 1.5 kg, 150 Ns/m, and 1 N/m, respectively. Also, in order to reduce the unexpected moments, $m_e$, $d_e$, $k_e$, $m_e$, $d_e$, and $k_e$ are set to 50 kgm$^2$, 2,500 Nms/rad, 0.15 Nm/rad, 1.5 kgm$^2$, 100 Nms/rad, and 0.5 Nm/rad, respectively. Figure 13 shows the sequence snapshots when the robot steps on an obstacle in both simulation and real-world environments. Although there was an unexpected contact between the swing foot and the obstacle, walking was not influenced by the obstacle because the impedance controller reduced the landing impact.

Second, the upper-body posture controller with the IMU sensor is used to modify the pelvis position and the orientation so the robot could walk upright on the inclined ground. However, slope estimation becomes a difficult problem because the IMU sensor value is affected by many factors. These factors include global inclination of the ground and deflections of joints and links because of elasticity. We therefore decided to update the ground slope only during the DSP.

4.4. Stair Climbing

For the stairs task, the robot must ascend a stairway that has a rail on the left side. However, our robot was unable to ascend the stairs by walking because of the torque limit and its short leg. Accordingly, we designed the walking procedure with a whole-body behavior as follows. First, THORMANG finds the rail using the chest LiDAR, then it grasps the rail with one hand, as shown in Figure 14(a). The purpose of this step is not only to ensure stability during
SSP, but also to avoid the torque limits of the leg’s actuators. Lastly, THORMANG raises a leg by rolling its foot to avoid self-collision between its thigh and its calf, as shown in Figure 14(b).

Figure 14(c) shows the input torque of the knee joint when the robot raises its right leg while grasping the rail. In comparison with the torque when the robot raised its right leg without the rail, the torque level dropped significantly. Table II shows the results of five tests performed in a week leading up to the DRC Finals. The average time of each phase for going up a step on the stairs is 101 s. As shown in Table II, column 1, our operator spent a great deal of time on Rail Detection, because the operator manually determined which point to grasp on the rail. We relied on a manual decision rather than an automatic decision because improper grasping caused by a small orientation error could easily lead THORMANG to fail on the stair task.

5. MANIPULATION

To control the upper body, we used a joint level controller and a task level controller. The task controller we used is based on Constrained Closed Loop Inverse Kinematics (Constrained CLIK) (Dariush, Hammam, & Orin, 2010; Dariush, Zhu, Arumbakkam, & Fujimura, 2010), as follows:

\[
\dot{q} = J^*([x_d + K(x_d - x)]).
\]  

(5)

where \( q \in \mathbb{R}^7 \), \( x_d \in \mathbb{R}^6 \), \( x \in \mathbb{R}^6 \), and \( K \in \mathbb{R}^{6 \times 6} \) are the joint value, the desired pose, the current pose of the arm, and the square matrix for CLIK gain, respectively. The weighted pseudoinverse of the Jacobian, \( J^* \in \mathbb{R}^{7 \times 6} \), is calculated by

\[
J^* = W^{-1}J^T(JW^{-1}J^T + \lambda^2I)^{-1}.
\]

(6)

where \( W \in \mathbb{R}^{7 \times 7} \) is the weighting matrix for avoidance of the joint limit and the torque limit, \( \lambda \) is a scalar component for singularity avoidance (Buss, 2004), and \( I \in \mathbb{R}^{6 \times 6} \) is an identity matrix.

With this controller, the robot can perform manipulation tasks such as turning the valve and rotating the doorknob. The following two subsections present our strategy to overcome some of the hardware-related limitations during manipulation.
Table II. Analysis of stairs task during practice runs in a week leading up to the DRC Finals.

<table>
<thead>
<tr>
<th></th>
<th>Rail Detection</th>
<th>Grasping</th>
<th>Raising Right Leg</th>
<th>Raising Left Leg</th>
</tr>
</thead>
<tbody>
<tr>
<td>Success Rate(^{12})</td>
<td>18/20</td>
<td>16/20</td>
<td>17/20</td>
<td>18/20</td>
</tr>
<tr>
<td>Average Time</td>
<td>51 s</td>
<td>12 s</td>
<td>18 s</td>
<td>20 s</td>
</tr>
<tr>
<td>Operation Type(^{13})</td>
<td>Manual</td>
<td>Semiautomatic</td>
<td>Automatic</td>
<td>Automatic</td>
</tr>
</tbody>
</table>

5.1. Torque Limit Issues

Humanoids used for rescue must have not only manipulability but also the strength to endure the weight of an object, because the robot must perform complex tasks in a disaster area. In the DRC case, the robot must pick up the drill and remove debris to succeed in the mission. However, the maximum torques on some of the actuators on the arm were not enough to perform these tasks. The payload for dexterous manipulation was less than 1 Kg, whereas the debris and drill were much heavier than that. To solve this issue, we designed the predefined trajectories by considering the kinematic configuration of the arm. For example, we designed a circular trajectory by only using one shoulder joint to cut a hole, as shown in Figure 15.

5.2. End-effector for the Drill

There are two types of the end-effectors provided by ROBOTIS, as mentioned in Section 2.1.2. However, these end-effectors are barely suitable for turning on the drill, because it is difficult to predict the kinematic states of the object after grasping the drill.

Therefore, we developed a new type of end-effector by taking the concept of docking inspired from self-reconfiguring modular robotics (Ostergaard, Kassow, Beck, & Lund, 2006; Yim et al., 2007). With this concept, we developed a new end-effector that can dock with a drill. The goal of the design is as follows. First, the new end-effector should be developed by modifying the original end-effector to reduce development time and cost. Second, the drill should be held firmly and become a constant kinematic condition after docking. Finally, the drill must be turned on regardless of the drill status when the end-effector closes.

According to these design goals, we developed the improved end-effector with underactuated fingers and spring hinges, as shown in Figure 16(a). The device was designed to have a simple shape without an additional actuator for the sake of minimizing changes in the form and weight of the existing gripper. The new end-effector consisted of two sliding panels, two tips, and spring hinges. Two aluminum sliding panels rotated and pulled the drill. The spring hinges provide elastic force for the sliding panels to return to its original state. Finally, tips were used for pushing the drill power button.

Figure 16(b) describes how the developed device helps the docking between the gripper and the drill. First, as both sides of the palm get closed, the sliding panel begins to rotate the drill. Next, when the drill is perpendicular to the gripper, the sliding panels wrap the drill entirely and pull the drill inside the gripper. Finally, the tip on the palm begins to push the button switch of the drill when the gripper closes.

To verify the performance of the developed gripper, we experimented to test whether the end-effector could dock with the drill, which was randomly placed. Figure 16(c) shows the result of the available range in order to dock with the drill through the experiments. As shown in Figure 16(c), the success boundary of docking is about 80 deg at the front and back sides of the drill, respectively.

6. COMMUNICATIONS

As mentioned in Section 2.2, our software system is a multilayer system comprising many computer units. In particular, various modules in each computer units are implemented using different programming languages to not only overcome the lack of development time but also decrease programming errors. Therefore, an efficient communication...
model between the modules is necessary to manage the software system.

We built the publish/subscribe communication model with three decoupling abilities as follows (Eugster, Felber, Guerraoui, & Kermarrec, 2003; Hartanto & Eich, 2014):

- **space decoupling**: each module does not need to know each other.
- **time decoupling**: each module does not need to actively participate in the interaction at the same time.
- **synchronization decoupling**: each module can asynchronously use information anytime and anywhere.

To build the publish/subscribe model, a message server in OCU1 is constructed to manage information among the computer units. The main roles of the server are not only sharing information from various modules in operation tools, but also storing log messages. Therefore, each operation tool can selectively subscribe to the broadcasting data from OCU1. Our server also records the log file for debugging. We can analyze the errors caused by the operators and the robot system using the logging system.

On the other hand, there are three limited links for communication between computer units, as mentioned in Section 2.2.1: Link1 (wireless link, 300 Mbit/s, always-on), Link2 (UDP line, 300 Mbit/s, randomly blackout), and Link3 (TCP line, 9,600 bit/s, always-on). In particular, to fully utilize Link3, the protocol of the system packet for Link3 is divided into two areas, namely the header area and the data area. The header has 8 bytes for task ID, packet length, and module flags, as shown in Figure 17(a). The task ID with 2 bytes contains main task and sub-task IDs. The packet length with 2 bytes is used to check the error packet. The module flags contain target module information. The data area contains sensory information, including encoders, low-quality image, and sound of THORMANG. The maximum size of the data area is 992 bytes.

![Figure 16.](image)

Figure 16. Developed end-effector with underactuated fingers and spring hinge: (a) design of developed gripper, (b) docking sequence with developed gripper, and (c) available range of docking from the experiments.

However, the data area size is limited to sending all sensory information. Therefore, this area should contain selective sensory data to show the circumstances of THORMANG depending on the situations. Accordingly, sound data are only sent to recognize whether the drill is turned on or not, and the current torques of the actuators are collected to judge the contact state between the drill and the cutting wall. This effort for Link3 helps the operator precisely determine what is happening despite the limited communication line bandwidth.

We validated the performance of our communication model by operating the wall task using only Link3. Figure 17(b) shows the size of sending and receiving data during the wall task. Although the TCP line supports a small data rate of 9,600 bit/s, the operator could successfully control the robot by sending data that he wanted to receive.

![Figure 17.](image)

Figure 17. Configuration of Link3: (a) system packet for Link3, (b) result of packet data in Link3 during experiment.
7. LAB TESTS AND DRC FINALS

Although we were fully prepared for all the missions in our lab, we only succeeded in four tasks during the DRC Finals, as shown in Figure 18. This section presents the strategies for several tasks, the difficulties in implementing those strategies, and our performance in the DRC Finals. Each team performed two runs in the DRC Finals. Therefore, we concentrated on presenting only the better run. Please see the video on our homepage\textsuperscript{13} for the overall performance from the lab test and the DRC Finals.

7.1. Driving

The robot must drive a vehicle (Polaris Ranger XP 900) through barriers in the course during the driving task. Therefore, the prerequisites for driving are what posture the robot should take behind the wheel and how the robot can drive the vehicle to a point adjacent to the wooden platform, which is a kind of single stair-step for the egress task.

We solved these problems by establishing the following strategies. First, we decided that the robot held the vehicle frame with the left hand and the steering wheel with the right hand, as shown in Figure 19(a). With the iris camera and webcams, we can secure not only the front view of THORMANG for the driving but also the side view of the vehicle for the next task, egressing. Second, we attached a pedal-assistant tool to the vehicle so that the pedal can be pressed, as shown in Figure 19(b). Our robot can drive at a constant velocity using the pedal-assistant tool. Third, the operator performed stop-and-go driving to accurately park the vehicle.

Team SNU performed the driving mission with a 100% success rate during the five practice runs in a week leading up to the competition, because the operator could successfully obtain a sense of distance using a bird’s-eyes view from

\textsuperscript{13}http://dyros.snu.ac.kr/drc-comp/
Figure 19. Strategy for the driving task: (a) view of the operators at the driving mission, (b) pedal-assistant tool.

Figure 20. Snapshots of the experiment for the egress task.

the iris camera, as mentioned in Section 2.2.4. The average time for the driving task was 104 s. We also succeeded at the driving task during the DRC Finals without any difficulty during the two runs. Consequently, Team SNU scored one point in this task with a finish time of 00:01:26. We were then the fifth fastest team in the DRC Finals.

7.2. Egress

The robot needs to get out of the vehicle and reach the door front on foot to complete the egress task. This mission is very difficult because any unexpected contact between the robot and the vehicle may cause robot instability. Figure 20 shows the process of the egress task in our lab. In preparing the egress mission, we opted for jumping because our robot was short. THORMANG used two arms to push its body forward against the vehicle and jump out of it.

The success ratio of the egress mission during five practice runs in a week leading up to the DRC Finals was 60%, with an average time of 212 s. The robot fell down in two out of five trials when it jumped out of the vehicle because of the large impact force between the foot and the ground. Therefore, we did not attempt the egress task at the DRC Finals. There was at least a 3 deg inclination, and we were sure that the robot would fall down at this inclined surface based on our experience during the practice runs.

7.3. Door

The robot must pass the cross line at the back of the door during the door task. Because many teams at the DRC Trials experienced difficulties opening the door against the wind, we established the FSM for the door task as a four-step procedure. First, the robot finds the position and orientation of the doorknob using the perception sensors. Second, the robot rotates the doorknob and opens the door slightly with the left hand. Third, the robot pushes the door with the right hand if it was not completely open. Fourth, the robot passes through the door after rotating its upper body by 90 deg to avoid collision between its shoulders and the door frame.

Table III shows the results of the five practice tests in our lab. During the practice runs, the operator directly located the goal position and orientation on the PCD if our perception algorithm failed to recognize the doorknob. At the competition, our perception algorithm succeeded in finding the doorknob position, and THORMANG opened the door.
Table III. Analysis of door task during practice runs in a week leading up to the DRC Finals.

<table>
<thead>
<tr>
<th></th>
<th>Doorknob Detection</th>
<th>Grasping</th>
<th>Door Opening</th>
<th>Walking</th>
</tr>
</thead>
<tbody>
<tr>
<td>Success Rate</td>
<td>4/5</td>
<td>5/5</td>
<td>5/5</td>
<td>4/5</td>
</tr>
<tr>
<td>Average Time</td>
<td>24 s</td>
<td>17 s</td>
<td>33 s</td>
<td>90 s</td>
</tr>
<tr>
<td>Operation Type</td>
<td>Automatic</td>
<td>Semiautomatic</td>
<td>Automatic</td>
<td>Semiautomatic</td>
</tr>
</tbody>
</table>

Table IV. Analysis of the valve task during practice runs in a week leading up to the DRC Finals.

<table>
<thead>
<tr>
<th></th>
<th>Valve Detection</th>
<th>Grasping</th>
<th>Valve Opening</th>
</tr>
</thead>
<tbody>
<tr>
<td>Success Rate</td>
<td>5/5</td>
<td>5/5</td>
<td>5/5</td>
</tr>
<tr>
<td>Average Time</td>
<td>24 s</td>
<td>15 s</td>
<td>10 s</td>
</tr>
<tr>
<td>Operation Type</td>
<td>Automatic</td>
<td>Semiautomatic</td>
<td>Automatic</td>
</tr>
</tbody>
</table>

widely. However, we had the RTX error in the RCU when THORMANG was passing through the door. Consequently, we had to stop the operation of the robot to check the computer system. We were imposed a 10-min penalty according to the DRC rules. After intervention, THORMANG went through the door frame without any problems.

7.4. Valve

For the valve task, the robot must open a valve with a circular handle by counterclockwise rotation. Therefore, we focused on recognizing the valve location under degraded communication, as mentioned in Section 3.1. Consequently, our perception algorithms succeeded in finding the valve at all times during the practice tests, as shown in Table IV. During the DRC competition, we successfully approached the valve using automatic perception, and our robot turned the valve using one wrist joint to rotate the valve at once.

7.5. Wall

The wall task consists of two subtasks. First, the robot must hold the drill designed for humans and turn it on. Second, the robot must cut out a hole in the wall using the drill. Using the efforts described in Sections 5.1 and 5.2, we focused on detecting a contact state between the drill and the wall. Although the impedance or force control for the end-effector was a good solution for this issue, we could not use this control during the cutting task because of the limited performance of the FT sensor and actuators in the arm. Instead, we measured the current torque of the actuators in the arm to judge whether the contact state between the drill and the wall is good or not.

Table V shows an analysis of the wall task in our lab tests. During these lab tests, the operator manually supervised whether the drill is turned on or not by checking the sound from the multimedia converter. Also, the operator directly evaluated the contact state, as mentioned above. Thus, the average times of Grasping & Switching and Reaching were longer than that of the other steps in the wall tasks.

The wall task was one of the most difficult tasks during the DRC Finals. Only seven out of the 23 teams successfully performed the wall task. In our case, our robot failed to turn on the drill at the first attempt because the operator misunderstood the distance between the drill and the end-effector. However, we turned on the drill in the second attempt, and we succeeded on the wall task. Figure 21(a) shows the predefined and the measured trajectories at the DRC Finals. The robot could draw a circle very well using our approaches. With our strategy, we became the fastest team in the DRC Finals, as shown in Figure 21(b). For details of the wall mission performance, please refer to our paper for the wall task (Park et al., 2015).

7.6. Overall Performance in the DRC Finals

In the end, we obtained 4 points for 58 min and our team finished in 12th place out of 23 teams. We used up most of the performance time for walking, because the DSP time during walking was greatly increased due to the uneven ground surface in the stadium.

However, the manipulating time for the driving, the door, the valve, and the wall were 00:01:26 (5th), 00:18:16 (19th), 00:01:01 (3rd), and 00:06:40 (1st), respectively. Also, Team SNU’s robot successfully finished the final without falling, while the bipedal robots used by many of the other teams fell down on the ground.

8. LESSONS LEARNED

In the previous sections, we introduced our technical and practical approaches to prepare for the competition and performance in the DRC Finals. Unlike prior humanoid studies, the purpose of the DRC is to develop the whole system,
Table V. Analysis of the wall task during practice runs in a week leading up to the DRC Finals.

<table>
<thead>
<tr>
<th>Operation Type</th>
<th>Drill Detection</th>
<th>Grasping &amp; Switching</th>
<th>Reaching</th>
<th>Cutting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Success Rate</td>
<td>5/5</td>
<td>4/5</td>
<td>5/5</td>
<td>5/5</td>
</tr>
<tr>
<td>Average Time</td>
<td>45s</td>
<td>15s</td>
<td>15s</td>
<td>62s</td>
</tr>
<tr>
<td>Operation Type</td>
<td>Semiautomatic</td>
<td>Automatic</td>
<td>Semiautomatic</td>
<td>Automatic</td>
</tr>
</tbody>
</table>

Figure 21. Analysis of the wall task: (a) comparison of desired trajectory and measured trajectory when THORMANG performed the cutting task at the DRC Finals, (b) time distribution for the wall task.

including the hardware, software, perception, communications, and robot controllers. Therefore, the DRC has taught us a number of lessons that will guide our future work. What we learned at the 2015 DRC Finals is presented as follows from a practical point of view:

**Bipedal walking still remains one of the most difficult tasks.** Bipedal walking is one of the main characteristics of humanoid robots. However, we believe that all participants in the DRC Finals agree that locomotion in the disaster area is still a difficult task. As mentioned in Section 7.6, although there was no falling when our robot performed the tasks, we did not finish all tasks because of our slow walking speed, which was caused by our walking strategy. Therefore, we will concentrate on developing a robust walking algorithm that guarantees fast speed and stability when the robot walks on an unexpected area.

**Falling might be unavoidable; hence, we should focus on surviving after falling.** Compared with robots having wheeled mobility, humanoids are more vulnerable to falling because of the relatively higher position of the COM and the narrower support polygon. Thus, even though getting up from a prone position was one of the qualification tasks in the DRC Finals, most participants may not think the humanoid can stand up all by itself after falling down during the runs (Guizzo & Ackerman, 2015). Figure 22 shows the number of falls of robots with bipedal walking and others at the DRC Finals. There were more bipedal robots than robots with wheeled mobility. Furthermore, no humanoid performed the getting up task by itself because of the structural malfunction caused by the impact from the ground. Therefore, studying survival after falling was necessary from the standpoint of both hardware and software. The humanoid robot should be designed using more durable materials. Moreover, a falling motion control should be developed to minimize physical damage.

**An efficient warning system is necessary to minimize operator error.** Many teams spent a great deal of time training the operators despite insufficient development time and manpower in order to minimize operator error. However, even well-trained operators at the DRC Finals made vital mistakes\(^{16}\) by misunderstanding the situation of the robot under poor communication conditions. In our case, no one in the operation center was able to realize the malfunction in THORMANG when it went through the door, as mentioned in Section 7.3, because the operators overlooked the current status of the robot in OCU1. These results imply that the operation tools should have a more efficient notification system for dangerous robot situations. For example, warning systems with haptic devices will help the operator understand the circumstances of the robot by transmitting not only visual effects but also vibrations and sounds.

**Operators want to intuitively control the robot.** However, our operators reported that input devices such as keyboards and mice were limited in intuitively operating the robot, because these devices were specialized for running

\(^{16}\)http://www.cs.cmu.edu/~cga/drc/events.html
two-dimensional operations. The input system using a 3D mouse and a haptic device was developed to construct an intuitive UI. However, this was not used at the DRC Finals because of the lack of time for operator adjustment. Therefore, we will concentrate on developing a new input system for operation tools to enhance the human-robot interaction.

Finding the level of autonomy is difficult. Although there are many algorithms for decision-making, developing an autonomous system like the human brain in the near future was impossible. On the other hand, although the manual system helps in the decision-making for the robot through the human-robot interaction, this system has a disadvantage because the performance of the robot varies from operator to operator. We therefore used event-based FSM to construct the semiautonomy system. However, we must answer the following question before constructing an efficient semiautonomy system: how far can we trust the results from the decision-making algorithm? In summary, more research is needed to find the optimal level between autonomy and semiautonomy systems.

9. CONCLUSIONS

This paper presents a detailed description of the technical and practical approaches of Team SNU in the 2015 DRC Finals. With THORMANG provided by ROBOTIS, we concentrated on developing a software architecture, which includes operation tools and robot controllers, and modifying the hardware. In particular, there are two philosophies in the development direction to overcome the hardware limitations of THORMANG and create the capabilities of the robot for the disaster situation: increasing stability and modularization.

Based on these philosophies, our technical and practical approaches have a number of unique features compared with other teams in the DRC Finals. First, we implemented various recognition algorithms depending on targets to enhance the automatic perception performance. In particular, the HOG features obtained from the depth image were used to find the valve. The effectiveness of the proposed strategy was verified through experiments, which demonstrate better precision and recall score than those of other conventional algorithms. Second, we concentrated on developing the multilayer software system, which includes various modules, to eliminate software system complexity and programming errors. We also built an efficient communication model based on the publish/subscribe model, to manage the software system and operation tools under the degraded communications. Finally, and most importantly, our controllers were developed considering stability as the top priority. In particular, we implemented the IMU-based compensator to overcome the joint elasticity. Our walking controller automatically changes the double support duration to enhance stability by considering the robot’s status. With these efforts in development, our approaches were verified at the 2015 DRC Finals, where Team SNU obtained 4 points for 58 min without falling.

However, our performance at the DRC is not perfect due to many reasons, including the hardware limitation and lack of development time. As mentioned in Sections 7 and 8, our robot could not complete all the missions, because walking speed was slow to maintain the robot’s stability. Moreover, our operation tools are limited in performing the complex tasks, because of the absence of a warning system and efficient input devices. Therefore, our future works will concentrate on developing a robust walking algorithm to overcome uncertain environments and modifying the robot, which is more suitable for whole-body control. We believe that the hard lessons learned from our rich experiences will help us to solve any remaining software and hardware issues in the future.

ACKNOWLEDGMENTS

This research was supported by the MOTIE under the robot industry core technology development project (No. 10050036) supervised by the KEIT. Also, this work was partially supported by the National Research Foundation of Korea (NRF) grant funded by the MSIP (No. NRF-2015R1A2A1A10055798). We would like to thank Team ROBOTIS for providing THORMANG and technical support. We also would like to thank Jeeho Ahn, Joonwoo Ahn, and Seungyeon Kim who provided support during development and competition.

Figure 22. Analysis of mobility types: (a) the number of robot types with respect to mobility types, (b) the number of falls of the robot for two runs with respect to mobility types.
REFERENCES

Burke, J., & Murphy, R. (2004). Human-robot interaction in user technical search: Two heads are better than one. In 13th IEEE International Workshop on Robot and Human Interactive Communication (pp. 307–312). IEEE.


Hartanto, R., & Eich, M. (2014). Reliable, cloud-based communication for multi-robot systems. In IEEE International Conference on Technologies for Practical Robot Applications (ToPRA) (pp. 1–8). IEEE.


