ARCHR - Apparatus for Remote Control of Humanoid Robots

Martyna Bula, Patrick Early, Eric Eide, Mannan Javid, and Daniel M. Lofaro (Faculty Supervisor)
Electrical Engineering Department, Volgenau School of Engineering
George Mason University, Fairfax, VA, USA

*Submitted for review in the IEEE/RSJ International Conference on Intelligent Robots and Systems 2015

***Abstract*—**
The Apparatus for Remote Control of Humanoid Robots (ARCHR) is an intuitive teleoperation system for high degree of freedom robots with haptic feedback. The purpose of this system is to allow minimally trained individuals to control a complex robot to complete a dextrous task. For example if it is a biohazard handling task then we want a biohazard handling expert operating the robot. However it should not be required that the biohazard handling expert also be a trained robot handler. The robot should be controlled intuitively and give good situational awareness. This is especially important in situations requiring first responders. The ARCHR system uses scaled kinematics of the target robot to incorporate the humans inherent intuitive understanding of joint motion. This setup also allows for hardware self collision detection. A scaled stereoscopic video feed in conjunction with a binocular virtual reality system immures the user into the body of the robot. This immersion is scale independent. Finally haptic feedback is used to extend the immersion to the tactile space. Three ARCHR controllers were created for three distinct humanoid robots, the DRC-Hubo, Baxter, and MiniBot. Each of these robots range from full-size to infant-size. Public testing was done using 30 untrained users on each platform. The results showed that the system does achieve the goal of creating an intuitive and immersive teleoperation system for high degree of freedom robots.

I. INTRODUCTION

The Apparatus for Remote Control of Humanoid Robots (ARCHR) is an intuitive teleoperation system for high degree of freedom robots with haptic feedback. This system leverages the humans' inherent and intuitive understanding of joint motion and tactile sensation. This creates the intuitive and immersive dextrous teleoperation system known as ARCHR. The purpose of this system is to allow minimally trained individuals to control a complex robot to complete a dextrous task including those encountered in *first responder* situations.

A need for such a teleoperation system was shown during the aftermath of the 2011 TEPCO Fukushima Diichi nuclear disaster. During the initial stages of this disaster first responders procured a number of robots to assist them with damage control. These robots include the rugged "gear-footed" iRobot Warrior Prototype equipped with a manipulator to pick up small objects; the waterproof Quince two by Japan's Chiba Institute of Technology equipped with a camera with pan/tilt/zoom features; and the proven iRobot PackBot equipped with a temperature and real-time video sensor packages.[1] Each robot requires a specially trained handler. Though helpful in getting information from areas too dangerous for humans the robots all have limited manipulation abilities. 24 hours after the tsunami roared into the harbor the first responders needed to vent the containment vessel to prevent the hydrogen gas from building up and exploding.[2] Without power the systems valves could not be turned on remotely. Without adequate manipulation the robots could not turn the valves manually. Therein lies the rub.

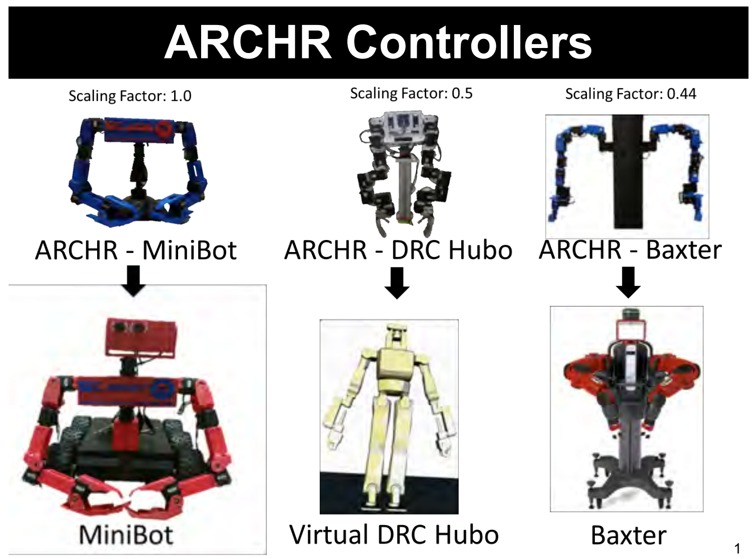

Fig. 1. *ARCHR System* (Apparatus for Remote Control of Humanoid Robots): The *ARCHR System* is a cloud based controller that allows intuitive use of a highly articulated robot. The physical controller is a kinematically scaled version of the target robot: Top Left - ARCHR for MiniBot; Bottom Left - MiniBot; Top Center - ARCHR for DRC-Hubo; Bottom Center - Virtual DRC Hubo; Top Right - ARCHR for Baxter; Bottom Right - Baxter. The ARCHR system is a kinematically scaled

The robots ended up not being deployed on the first day, or even the second. It was not until weeks later when the robots started to assist the relief effort. To quote the TEPCO spokesman Shogo Fukuda "*the company has only now begun using the robots because it took several weeks for crews to learn how to operate the complex devices* [1]". It is evident that for a robot to be effective in a disaster situation it has to be simple and intuitive to use by the experts in the field (first responders) not just the robot handlers.

In direct response to the these problems, the Defense Advanced Research Projects Agency (DARPA) started the DARPA Robotics Challenge (DRC) with the goal of "develop[ing] ground robots capable of executing complex tasks in dangerous, degraded, human-engineered environments"[3]. The eight DRC events focus on 7 "capability exercises" which include:

1) **Perception**
2) **Decision Making**
3) Mounted Mobility
4) Dismounted Mobility
5) **Dexterity**
6) Strength
7) Endurance

The ARCHR system (Apparatus for Remote Control of Humanoid

[1]Popular Mechanics - Robots That Braved Fukushima (Visited 2014-02-03): http://www.popularmechanics.com/technology/robots/g795/3-robots-that-braved-fukushima-7223185/

[2]IEEE Spectrum - 24 Hours at Fukushima (Visited 2014-02-05): http://spectrum.ieee.org/energy/nuclear/24-hours-at-fukushima

[3]DARPA Robot Challenge (Visited 2014-04-10): http://www.darpa.mil/Our_Work/TTO/Programs/DARPA_Robotics_Challenge.aspx

Robots) was designed specifically to tackle the problems of *perception*, *decision making*, and *dexterity* with a focus on intuitive control and situational awareness while under teleoperation. Three ARCHR systems were created for 3 distinct robots, the DRC-Hubo (Section IV-A), Baxter (Section IV-B), and MiniBot (Section IV-C). All three systems and target robots can be seen in Fig. 1. Two of the latter were tested on two groups of 30 non-expert (untrained) users on the completion of a "peg-in-hole" task (Section V). A test case for the ARCHR system is then demonstrated: *bio-hazardous material handling using cooperative robots under telepresence control* (Section VII).

II. BACKGROUND

There are currently a variety of methods to meet the demand of remotely controlling a dextrous humanoid robot. These methods range from attaching sensors to the user's joints [2]; utilizing 6 degree of freedom (DOF) sensors on the user's end-effector and inverse kinematics for the robot [3]; using wearable motion-capture suits and mapping the pose to that of the robot [4], [5]; leveraging brain-machine interfacing (BMI) methods [6], [7]; standard GUIs such as RVIZ and Moveit! in conjunction with a kinematic model of the robot [8]–[10]; or by using a simple gaming controller or smartphone/tablate [11]. Many of these methods also utilize a virtual reality system such as the Oculus Rift in order to stream real-time video from the remote location to the user. While these methods are shown to work in lab settings with trained users. Most are complex or non-intuitive thus giving them a steep learning curve.

In the use cases demonstrated on Hubo, Mahru[4], and Telebot [2]–[4] the humanoids mimics the user in real-time based on the wearable suit/tracking devices. The user is required to have proportionally the same amount of work-space as the robot. This is not optimal as it requires the user to physically move through a large work-spaces.

The BMI methods worked however: 1) their command set is limited; 2) the bandwidth is extremely low at 0.2 Hz and 3) it requires substantial training [6], [7].

GUI methods such as RVIZ and Moveit! offers a lot of operational options. These allow for both joint-space and work-space control. Given the latter and that the input is give through the keyboard and mouse the GUI methods are typically non-intuitive and have relatively slow user input respectively [8]–[10]. Because the system is non-intuitive they require a trained user for operation. The extra step of acquiring inverse kinematics (IK) solutions of the *target robot* is also required if controlling in work-space.

Using simple video game controllers or smartphones/tablets is intuitive. However they do not provide the necessary control to make use of the dexterity capabilities of high DOF robots such as humanoids [11].

It is evident that there is a lack of intuitive systems for controlling high DOF robots such as humanoids. There are however multiple intuitive controllers for lower DOF systems. The DaVinci system is one such example It was designed to imitate the controls (kinematic scale of 1:1) of traditional laparoscopic surgery [12], [13]. Because of this imitation, highly trained medical surgeons have been able to use this system with relative ease making it one of the most popular surgical robots. One aspect of the original DaVinci system was its lack of haptic feedback. Due to this limitation surgeons required more training to perform safe and accurate surgeries. The addition of haptic feedback and stereoscopic vision provided the surgeons with a great amount of situational awareness. This then allowed the surgeons to perform the same accurate tasks while only requiring a fraction of the original training on the robot [14]–[16].

[4]IEEE Spectrum - Humanoid Robot Mahru Mimics a Person's Movements in Real Time (Visited 2014-02-18):
http://spectrum.ieee.org/automaton/robotics/humanoids/042710-humanoid-robot-mahru-real-time-teleoperation

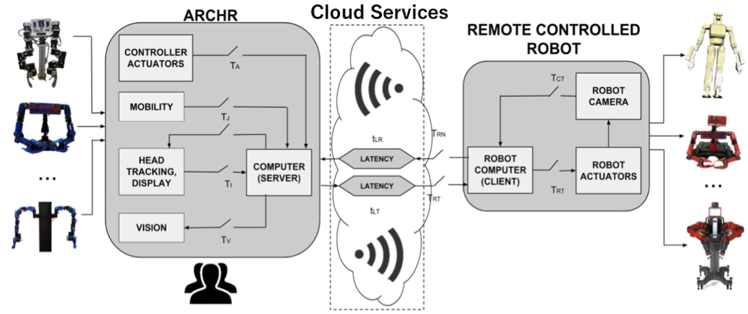

Fig. 2. *ARCHR System:* (left to right) ARCHR physical controllers, ARCHR controller middleware, cloud services, target robot middleware, target robots.

One of the primary reasons why the DaVinci system has such a low learning curve is because it looks and feels like typical laparoscopic tools that the surgeons are familiar with. I.e. it is kinematically scaled to these tool by a factor of 1:1. For this reasons the ARCHR system is made to be kinematically scaled to the target humanoid. As long as the operator knows that if you move joint X on the ARCHR controller the same corresponding joint will move on the *target robot* the system will be as intuitive as the DaVinci system. Like the DiVinci system ARCHR utilizes stereoscopic vision (Section III-D) and haptic feedback (Section III-E). This increases the user's situational awareness regardless of the size (big or small) of the *target robot*.

III. METHODOLOGY

ARCHR - Apparatus for Remote Control of Humanoid Robots is a cloud based controller that allows intuitive use of a highly articulated robot. The physical controller is a kinematically scaled version of the target robot. The joint space coordinates are mapped directly from the physical ARCHR controller (θ_{ac}) to the *target robot* (θ_{tr}). The appendage lengths of the ARCHR controller (L_{ac}) is a scaler of the *target robot's* appendage lengths (L_{tr}) by a factor of K_s, see (1). In short when joint A' on the ARCHR controller is rotated θ rad, joint A on the target robot is also rotated θ rad. If joint B on the target robot hits an object and can not move joint B' on the ARCHR controller stiffens. This lets the user know the *target robot* has hit an object or itself. This allows the user to "*back off*" or "*continue with*" their original motion depending on if their goal was to create the collision or not. This keeps the *human in the loop*.

$$\theta_{tr}(n) = \theta_{ac}(n) \qquad\qquad L_{ac}(m) = K_s L_{tr}(m) \qquad (1)$$

The ARCHR system is comprised of:

1) **Target Robot:** the robot performing the task.
2) **ARCHR Controller:**
   - kinematically scaled version of the *target robot*.
   - stereoscopic virtual reality head tracking system.
   - haptic feedback from *target robot*.
3) **Cloud Services:**
   - processes feed-forward data (joint commands)
   - processes feed-back data (vision, haptics, state)
   - throttling for bandwidth/latency control

See Fig. 2 for system configuration.

*Submitted for review in the IEEE/RSJ International Conference on Intelligent Robots and Systems 2015

A. Target Robot:

The target robot is the robot performing the given task. This robot can be big or small. It can be legged, wheeled, flying, submersible or have any other method of movement. The minimum sensor capabilities of the target robot is stereo vision and joint-space current/torque feedback. The target robot runs a “process based” middleware which is a variant of the Hubo-Ach controller [17]. This layer abstracts the target robot to a common convention expected by the *cloud services*. The middleware is target *robot specific*, however the *cloud services* is not. This translation allows for use on a wide range of robots.

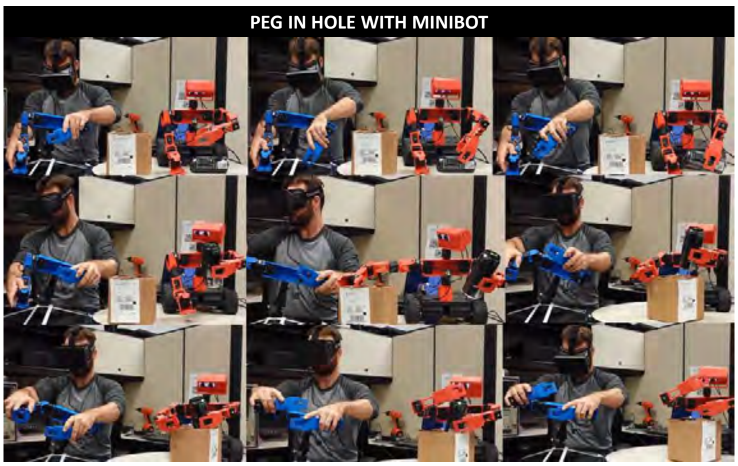

Fig. 3. *ARCHR System* controlling the *MiniBot* robot for *peg-in-hole* task. Time moves from left to right, top to bottom.

B. ARCHR Controller:

The ARCHR Controller needs to have the same number of DOF as the manipulation portion as the target robot. The mobile portion of the target robot is controlled by a high level controller. The configuration of the physical controller should be kinematically scaled to that of the target robot. This means that arm segment lengths should all follow the same linear conversion as seen in (1). i.e. if K_s is the conversion factor then the length (L_{tr}) of apendage A on the target robot will map to the corresponding apendage A' with length $K_s L_{tr}$. Joint order and orientation remains the same between the *target robot* and the ARCHR controller. Inherent hardware self collision detection is achieved by having all dimensions of the ARCHR controller the same scaled as the *target robot*. Each joint on the ARCHR controller must be able to have its output torque adjusted in real-time. Like the *target robot* the ARCHR Controller runs a “process based” middleware which is a variant of the Hubo-Ach controller [17]. This layer abstracts the ARCHR Controller to a common convention expected by the *cloud services*. The middleware is target robot specific but the *cloud services* is not. This translation allows for use on a wide range of controllers.

The ARCHR controller also consists of a stereoscopic vision system with head tracking capabilities. The orientation of the head is transmitted over the reference channel as described above. See Fig. 3 for a visual of the head tracking system working with the MiniBot *target robot*. The stereo vision system and configuration is described in Section III-D.

The user is not expected to give the robot joint level commands for translation. Instead a high level controller is used for translational purposes. The base of the kinematically scaled portion of the ARCHR controller is attached to a 6DOF sensor (high DOF version of a joystick). This is gives high level translation commands (forward, backward, turn, etc.) based on user input (push, pull, rotate, etc.)

C. Cloud Services:

The cloud services includes model based self collision detection [18], object detection/image augmentation, and automatic feed-forward/feed-back channel throttling. This is done to offload processing and for bandwidth/latency control. The cloud services receive the joint space command data (feed-forward channel) from the ARCHR controller at period T_{ru} and re-transmits the the data to the *target robot* at period T_{cr}. The rate T_{cr} is adjusted based on: 1) the latency between the cloud server and the ARCHR controller; and 2) the latency between the cloud server and the *target robot*.

The same process happens in reverse order for the feed-back channel. The target robot transmits state date to the cloud server at period T_{rs} and is then re-transmitted to the ARCHR Controller at period T_{cs}. This process is again repeated for the visual feedback channel with the receive and transmit periods of T_{rv} and T_{cv} respectively.

To conserve bandwidth the periods are limited to the following:

$$T_{ur} \leq T_{cr} \qquad \mathit{and} \qquad T_{rs} \leq T_{cs} \qquad \mathit{and} \qquad T_{rv} \leq T_{cv} \tag{2}$$

The specifics about the cloud services will be described in future publications. For this document we consider the cloud server to have a consistent latency and bandwidth.

D. Stereoscopic Vision:

When using stereoscopic vision the baseline, or the distance between the cameras, is a key factor in determining depth. The closer the cameras are together, the further away things look. For this reason the baseline of the stereo pair on the *target robot* should be proportional to the distance between the average human's eyes (L_{EP}) and the ratio of the *target robot's* total arm length (L_{Ax}) to that of the average human's arm (L_{AP}). This conversion allows the user to see at the same scale as the robot. Thus the operator can use the robot's arms more effectively/intuitively. Fig. 4 and (3) shows the creation of the scaling ratios for the baseline using the Baxter and the MiniBot as examples. L_{EB} is the baseline for the vision system on the Baxter robot and L_{EM} is the baseline for the vision system on the MiniBot. L_{AB} is the arm length of the Baxter and L_{AM} is the arm length of the MiniBot.

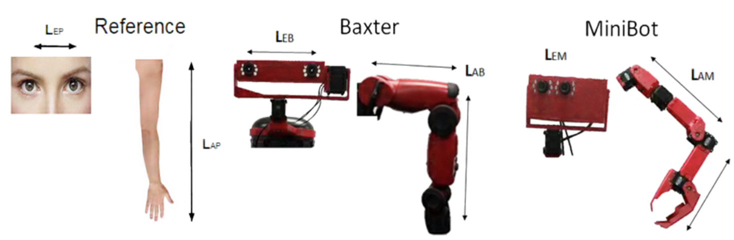

Fig. 4. *ARCHR System* Kinematically scaled

$$L_{EB} = L_{AB}\frac{L_{EP}}{L_{AP}} \qquad \mathit{and} \qquad L_{EM} = L_{AM}\frac{L_{EP}}{L_{AP}} \tag{3}$$

To increase situational awareness and utilize the human brain's inherent ability to calculate depth, the optics of the *target robot's* vision system should be similar to that of the human. Wide angle lenses are used to accommodate for the human's $\sim 180^o$ field of view (FOV) capabilities. The FOV of the lens is *target robot* scale independent. However the focal length is scale dependent.

All depth calculations are done naturally by the human. To assist this process the *cloud services* augment the image depending on the given optics of the *target robot*. Augmentation includes but is not limited to image barreling, enhancing, etc.

E. Haptic/Force Feedback

The ARCHR system implements haptic and force feedback. This is done to allow the user to intuitively know when the robot has hit something. It also gives the user a sense of weight when the *target robot* is lifting something of mass. This is done by feeding back the joint-space current/torque from each of the *target robot's* joints. A scaled version of that torque (τ_n) is then applied to the normally *free wheeling* ARCHR controller joints. τ_n is the *target robot's* measured torque where n is the given joint. τ_n' is the applied torque to the corresponding joint on the ARCHR controller.

$$\tau_n' = \begin{cases} 0 & : \quad |\tau_n| > \tau_{lower} \\ K_\tau \tau_n - \tau_{lower} & : \quad \tau_{lower} \leq |\tau_n| \leq \tau_{upper} \\ K_\tau \tau_{upper} - \tau_{lower} & : \quad |\tau_n| < \tau_{upper} \end{cases} \tag{4}$$

K_τ is a positive torque conversion gain. τ_{lower} is the lower torque threshold allowing for effects of gravity, inertia, and joint friction to be ignored. τ_{upper} is the high torque threshold which is set so the user will still be able to move the joints on the ARCHR controller no matter how much torque is being applied to the *target robot*. This allows the user to "back off" or "continue with" their original motion depending on if their goal was to create the collision or not. These gains and thresholds are *target robot* dependent and user adjustable.

IV. CONTROLLERS AND ROBOTS

The ARCHR system is designed to give intuitive control of highly articulated robots to a minimally trained user. The ARCHR system was implemented on three separate and distinct systems. One legged (DRC-Hubo), one stationary (Baxter), and one wheeled (MiniBot). Each ARCHR system and corresponding *target robot* can be seen in Fig. 1.

This section describes the different controllers, robots, and equipment used for the experiments in Section V.

A. ARCHR: DRC-Hubo

The ARCHR system for the DRC-Hubo (Section IV-D) was designed using Dynamixel MX-28 actuators. The ratio of the ARCHR controller to the *target robot* is 0.5. The baseline ratio for the stereoscopic vision is 1.15. Though the robot is shorter then a human the arms are longer thus the ratio greater then 1.0. This ARCHR controller was primarily tested on the Virtual DRC-Hubo which is based on OpenHubo [18]. Both the DRC-Hubo and the Virtual DRC-Hubo use Hubo-Ach for control. No modifications to the ARCHR controller is needed to switch between virtual and real robots. The ARCHR system for the DRC-Hubo was the prototype. Full OpenSource instruction and tutorials for the creation of the ARCHR DRC-Hubo is located on the LofaroLab's wiki.[5]

B. ARCHR: Baxter

The ARCHR system for the Baxter robot (Section IV-E) was designed using the smaller Dynamixel AX-12 actuators. Unlike the ARCHR DRC-Hubo prototype the ARCHR Baxter utilized 3D printing technology for greater scaled appendage accuracy. Fig. 5 (right) shows the 3D printed parts and final design/construction the ARCHR Baxter. The ratio of the ARCHR controller to this *target robot* is 0.44. The baseline ratio for the stereoscopic vision is 1.30. Full OpenSource instruction and tutorials for the creation of the ARCHR DRC-Hubo is located on the LofaroLab's wiki.[6]

[5]ARCHR DRC-Hubo: http://wiki.lofarolabs.com/index.php/Hubo_DRC_Peg-In-Hole_%26_Valve_Turn

[6]ARCHR Baxter: http://wiki.lofarolabs.com/index.php/Archr_Baxter

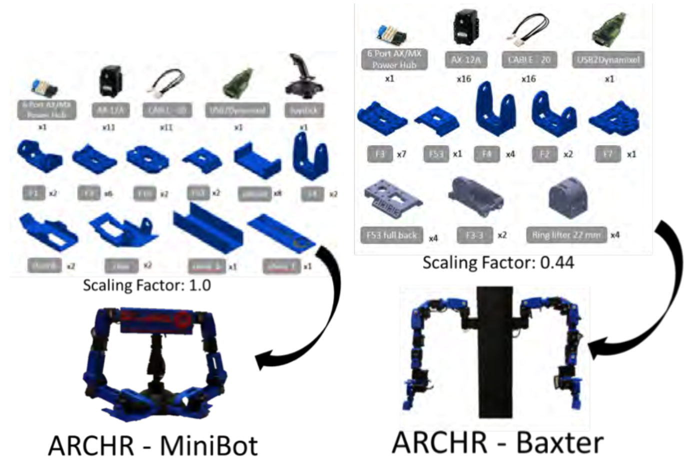

Fig. 5. All of the parts needed to make the Baxter and MiniBot ARCHR System. Tutorials and parts can be seen/downloaded from the Lofaro Lab's wiki.

C. ARCHR: MiniBot

The ARCHR system for the MiniBot robot (Section IV-F) was also designed using Dynamixel AX-12 actuators. ARCHR MiniBot utilized 3D printing technology for greater scaled appendage accuracy. Fig. 5 (left) shows the 3D printed parts and final design/construction the the ARCHR MiniBot. The ratio of the ARCHR controller to this *target robot* is 1.0. The baseline ratio for the stereoscopic vision is 0.35. Full OpenSource instruction and tutorials for the creation of the ARCHR DRC-Hubo is located on the LofaroLab's wiki.[7] Ratios for all ARCHR systems can be found in Table I.

TABLE I

RATIOS FOR ARCHR SYSTEMS TESTED

	Physical Controller *controller* to *target robot*	Stereoscopic Baseline *robot* to *human*
DRC-Hubo	0.50	1.15
Baxter	0.44	1.30
MiniBot	1.00	0.35

D. Target Robot: DRC-Hubo

The DRC-Hubo is a 150 cm tall, 60 kg, humanoid robot with two 7 DOF arms, two 6 DOF legs, a 3 DOF head with stereo vision, and a 1 DOF waist. DRC-Hubo is a walking humanoid robot that is powered by a single 48V Li-ion battery. This robot was used in the 2014 DRC-Trials in Homestead Florida, USA.

E. Target Robot: Baxter

The Baxter (with platform) is a 185 cm tall humanoid robot with two 7 DOF arms, and a 2 DOF head with stereo vision. It is a stationary robot that runs off of wall power. Designs, schematics, and tutorials on how to make the stereo head for Baxter is available on the Lofaro Labs wiki.[8]

F. Target Robot: MiniBot

In order for others to validate our ARCHR controller we created a fully OpenSource, 3D printed wheeled humanoid *target robot* named MiniBot. All of the parts needed to make the MiniBot can be seen in Fig. 6. Tutorials and parts can be seen/downloaded from the Lofaro Lab's wiki[7].

[7]ARCHR MiniBot: http://wiki.lofarolabs.com/index.php/Minibot/ax-12a/ax-18a

[8]Baxter Stereo Head Tutorial: http://wiki.lofarolabs.com/index.php/Baxter_head

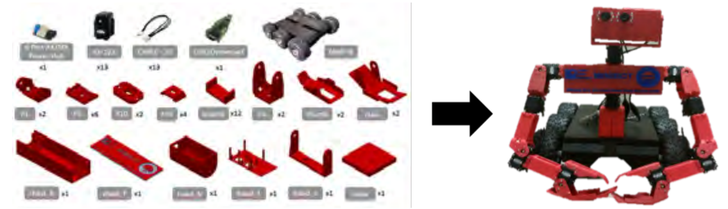

Fig. 6. All of the parts needed to make the MiniBot can be seen in Fig. 6. All of the parts needed to make the MiniBot. Tutorials and parts can be seen/downloaded from the Lofaro Lab's wiki[7].

The MiniBot is a 14 cm tall wheeled humanoid with two 5 DOF arms, a 2 DOF head with stereo vision, and a 1 DOF waist. The lower body is a six wheeled skid steering platform. The system is powered by a single 12V NiMH battery.

V. EXPERIMENT

Two test cases were implemented to determine the accuracy of the ARCHR system. These test cases focused on *accuracy* and *ease of use*.

A. Accuracy:

Accuracy was determined by analyzing the ARCHR controllers' end effector work-space position and joint-space positions with respect to the *target robot's* corresponding (scaled) work-space position and joint-space angles. Two ARCHR systems were tested, ARCHR Baxter and ARCHR MiniBot.

Each ARCHR controller was posed in a given position. Work-space and joint-space information was recorded for both the ARCHR controller and *target robot*.

1) ***Baxter****:* Fig. 7 shows the *ARCHR System* controlling the *Baxter* robot. The plot shows the commanded work-space positions and the actual work-space positions. The commanded work-space position is calculated by the use of forward kinematics and the commanded joint-space values from the ARCHR controller.

$$T = T_0^N = \prod_{n=1}^{N} T_{n-1}^{n}(\theta_n) \tag{5}$$

N is the number of joints in the kinematic chain and θ_n is the joint-space angle of joint n.

The actual position of the Baxter's arm is lower then that of the commanded position from that of the ARCHR controller. This is due to the low gains for gravity compensation on the *target robot*. The maximum error was approximately 6.0 cm.

2) ***MiniBot:****:* Fig. 8 shows the *ARCHR System* controlling the *MiniBot*. The plot shows the commanded work-space positions and the actual work-space positions. Like the Baxter the commanded work-space position for the MiniBot was calculated by using forward kinematics and the commanded joint-space values from the ARCHR controller. The MiniBot's joints are much stiffer then that of the Baxter and thus the error seen between the commanded position and the actual position is much lower. The maximum error was approximately 0.5 cm.

These results were then confirmed for each robot via the use of a second comparison technique. The work-space values (x', y', z') of the controller and the work-space values of the *target robot* (x, y, z) were measured. The ARCHR controller to *target robot* ratio was then applied. See Table I and (6).

$$\sqrt{(x' - \alpha x)^2 + (y' - \alpha y)^2 + (z' - \alpha z)^2} \leq 1cm \tag{6}$$

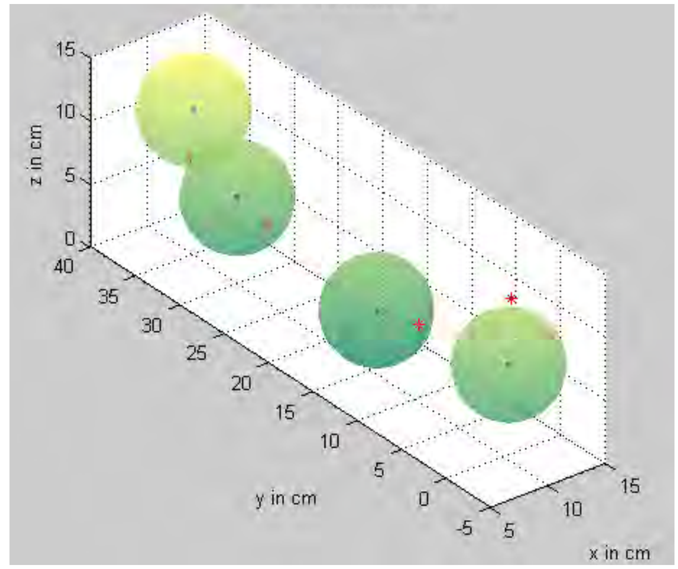

Fig. 7. *ARCHR System* controlling the *Baxter* robot. Showing the commanded work-space positions and the actual work-space positions. (Red ($*$) = Commanded; Blue ($\bullet$) = Actual)

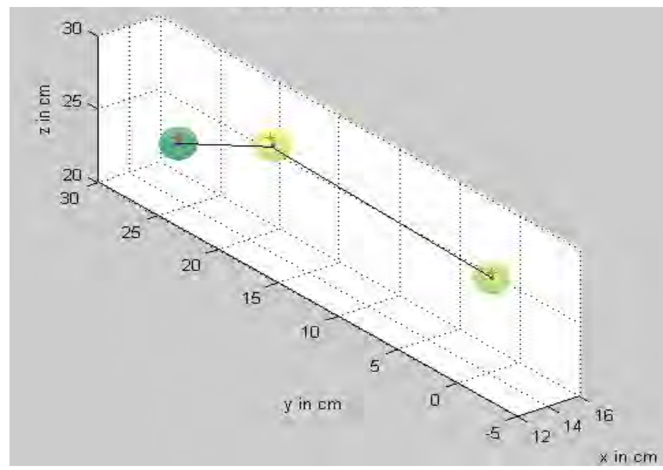

Fig. 8. *ARCHR System* controlling the *MiniBot* robot. Showing the commanded work-space positions and the actual work-space positions. (Red ($*$) = Commanded; Blue ($\bullet$) = Actual)

α is the *physical controller* ratios found in Table I. (x, y, z) denote the work-space position of the *target robot's* end-effector and (x', y', z') is the work-space position of the ARCHR controller's end-effector.

B. Ease of Use:

To determine the ease of use of the system, 30 tests among untrained users was given. The simple task given was to be performed after only one minute of training. Each subject used the ARCHR system to complete a peg-in-hole type task. Initial tests were done on ARCHR DRC-Hubo (see Fig. 9). Baxter and MiniBot ARCHR Systems were used for the 30 subject trials. The data obtained in this test case was as follows:

1) Time to object pickup
2) Time to object insertion
3) Number of dropped objects
4) Number of missed holes
5) Inadvertent robot joint movement
6) Loss of control of the robot
7) User experience from a Q&A

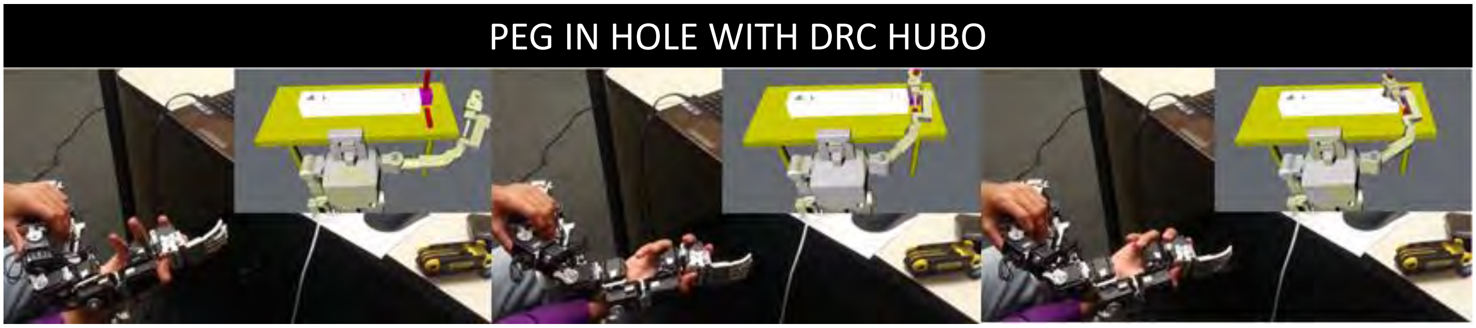

Fig. 9. *ARCHR System* controlling the *DRC-Hubo* robot for *peg-in-hole* task in the virtual environment OpenRAVE using Virtual DRC-Hubo.

8) System failures

Items 1-3 are all measurable objective data quantities. Items 4-8 are based on a person's judgment, thus they are subjective.

The Q&A was a list of questions was given to each subject and recorded. These questions related to: the ease of use; perception of the concept; and the overall experience The testing was done in public with a random set of subjects over the course of three hours.

30 Trials (Unique Users)	Baxter	MiniBot
	Untrained User	Untrained User
Time to retrieve peg	53 sec	60 sec
Time to complete the test	1 min 18 sec	1 min 45 sec

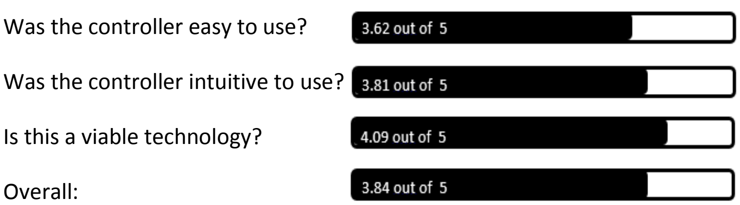

Fig. 10. *ARCHR System* controlling the *Baxter* and *MiniBot* robot. Triles done by 30 unique (untrained) users per robot. Time to peg/ball pickup and completion time as well as user survey results.

	Baxter		MiniBot	
	Female	Male	Female	Male
Time to retrieve peg	1 min 10 sec	45 sec	40 sec	64 sec
Time to complete the test	1 min 40 sec	1 min 8 sec	1 min 11 sec	1 min 51 sec
	Gamer	Non-Gamer	Gamer	Non-Gamer
Time to retrieve peg	46 sec	1 min 7 sec	1 min 7 sec	17 sec
Time to complete the test	1 min 6 sec	1 min 47 sec	1 min 54 sec	50 sec
	Right Handed	Left Handed	Right Handed	Left Handed
Time to retrieve peg	59 sec	36 sec	1 min 3 sec	40 sec
Time to complete the test	1 min 26 sec	59 sec	1 min 52 sec	1 min 6 sec

Fig. 11. *ARCHR System* controlling the *Baxter* and *MiniBot* robot. Triles done by 30 unique (untrained) users per robot. Subject results divided by attributes.

VI. RESULTS

A. Public Testing:

Fig 10 and 11 shows the average task times for each robot as well as the results from the Q&A. In addition errors such as dropped peg or missed occurred (five times or more) occurred in 9 of 30 cases. 26 of 30 users successfully completed the task. 4 users lost control of the system and were not able to complete the task (2 users controlling MiniBot and another 2 controlling Baxter).

B. Bandwidth:

The feed-forward channel to the *cloud services* was measured to consumed 2.24 *kbps* of bandwidth and the feed-back channel consumed 4.48 *kbps*. Both the feed-forward and feed-back channels were updated at 20 *hz*. A full breakdown of the measured feed-forward and feed-back channel bandwidths can be found in Table II.

TABLE II
ARCHR FEED-FORWARD/FEED-BACK CHANNEL BANDWIDTH

	Update Rate (Hz)	Feed-Forward (kbps)	Feed-Back (kbps)
DRC-Hubo	20	2.56	5.76
Baxter	20	2.24	4.16
MiniBot	20	2.08	3.84

Our video was streamed using H.264 / MPEG-4 AVC compression and a resolution of 320x240 at 12 frames per second per camera. This achieved a measured video bandwidth of 130.91 *kbps*. This resolution and rate pair was chosen because of the subjective "*good balance*" between bandwidth, resolution and update rate. Other tested/measured pairs can be found in Table III.

TABLE III
ARCHR VIDEO FEEDBACK BANDWIDTH USING H.264 / MPEG-4 AVC COMPRESSION

Resolution Per Camera (pixels)	Frame Rate (Hz)	Stereo Pair Bandwidth (kbps)
160x120	12	32.72
320x240	12	130.91
640x480	12	523.64
1280x720	12	1576.36
1920x1080	12	3534.55

C. Q&A Feedback

Out of a top score of 5 the ARCHR controller was rated 3.62 in the category of "*ease of use*," 3.81 in "*intuitive to use*," and 4.09 for "*valuable technology*." The overall score for the ARCHR system was 3.84 out of a possible 5. The full breakdown can be found in Fig. 10.

Crowd sourced "*Use Cases for the ARCHR System*" were:

1) **disaster areas**
2) mining sites
3) hospitals
4) exploration
5) **quarantine sites**
6) war zones
7) **radioactive sites**
8) bomb defusing

D. OpenSource

The final project is fully OpenSource. Tutorials on how to create the ARCHR system and the MiniBot are available on the Lofaro Lab's website.[9] All software and 3D printable CAD models are also available on the website for download (http://lofarolabs.com/projects/archr/).

VII. EXPERIMENT VALIDATION:

Bio-hazardous material handling using cooperative robots under telepresence control

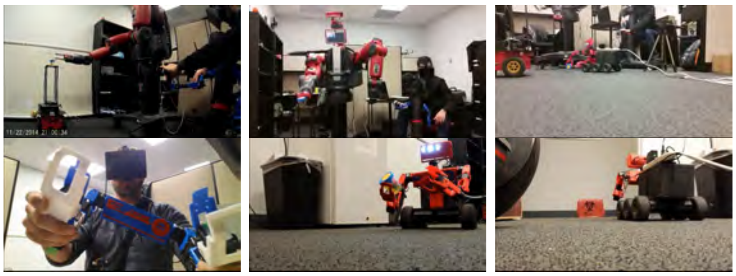

Fig. 12. Simulated bio hazardous scenario: Use the *ARCHR System* to control the *Baxter* and *MiniBot* to perform the cooperative task of picking up and disposing of a bio hazardous container without putting humans in harm. Time goes from left to right, top to bottom. See Fig. 13 for link to full video.

To validate the system we picked a use case from the *Q&A Feedback* in Section VI-C. The use case picked was *quarantine sites*. The situation is as follows:

" *A bio hazardous agent was released in the building. No one can enter or leave in fear of contamination. The area will not be safe for Level C hazmat suits until the agent is contained.* "

It is explicit that the situation is dire due to the quarantine. Even wearing a hazmat suit would keep the wearer's life in immediate danger. To mitigate this risk to human lives we send in two robots, a Baxter and a MiniBot. Both robots are controlled by bio hazard containment specialists. They are both using the ARCHR System. Fig. 12 shows how the ARCHR operator of the Baxter easily picks up the bio hazardous container and hands it to the more mobile MiniBot. The ARCHR operator of the MiniBot easily takes it from the Baxters claw and places it in the bio hazardous containment vestal. All of this is done while using the experts for the situation (bio hazard containment) as the robot operators. Due to the intuitiveness of the system the operators only require minimal training, all of which was done on site. A video of this demonstration can be found on the QR-Code link in Fig. 13 on the ARCHR Systems' home page: http://lofarolabs.com/projects/archr/

Fig. 13. Experiment Validation - Demonstration Video: Bio-hazardous material handling using cooperative robots under telepresence control. Link: http://lofarolabs.com/projects/archr/video/biohazard

VIII. CONCLUSION

The ARCHR System was shown to drastically reduce operator training time by creating an intuitive and immersive system. This is proven through the high task completion rate of the untrained subjects in Section VI. The system was implemented on multiple (unique) highly articulated robotic systems including the DRC-Hubo, Baxter, and MiniBot thus showing its flexibility. An experimental validation test in the form of *performing a simulated task in a "quarantine site"* was successful. The entire system is OpenSource from software to the mechanical 3D CAD designs. This invites others to apply and validate the ARCHR system on their robots.

[9] Lofaro Lab's Home Page: http://lofarolabs.com/

REFERENCES

[1] S. Kawatsuma, M. Fukushima, and T. Okada, "Emergency response by robots to fukushima-daiichi accident: summary and lessons learned," in *Industrial Robot: An International Journal*, 2012, pp. 428–435.

[2] D. Limbu, W. Alvin, C. Yuanwei, A. Adiwahono, T. A. Dung, and B. S. Han, "A pilot study on telebot, a internet-based teleoperated robot," in *Control, Automation and Systems (ICCAS), 2013 13th International Conference on*, Oct 2013, pp. 845–850.

[3] R. O'Flaherty, P. Vieira, M. Grey, P. Oh, A. Bobick, M. Egerstedt, and M. Stilman, "Humanoid robot teleoperation for tasks with power tools," in *Technologies for Practical Robot Applications (TePRA), 2013 IEEE International Conference on*, April 2013, pp. 1–6.

[4] D. Lofaro, C. Sun, and P. Oh, "Humanoid pitching at a major league baseball game: Challenges, approach, implementation and lessons learned," in *Humanoid Robots (Humanoids), 2012 12th IEEE-RAS International Conference on*, Nov 2012, pp. 423–428.

[5] K. D. Nguyen, I.-M. Chen, S. H. Yeo, and B.-L. Duh, "Motion control of a robotic puppet through a hybrid motion capture device," in *Automation Science and Engineering, 2007. CASE 2007. IEEE International Conference on*, Sept 2007, pp. 753–758.

[6] A. Finke, A. Knoblauch, H. Koesling, and H. Ritter, "A hybrid brain interface for a humanoid robot assistant," in *Engineering in Medicine and Biology Society, EMBC, 2011 Annual International Conference of the IEEE*, Aug 2011, pp. 7421–7424.

[7] C. Bell, P. Rawichote, R. Chalodhorn, and R. Rao, "Control of a humanoid robot by a noninvasive braincomputer interface in humans," in *Journal of Neural Engineering*, vol. 5, num. 2, 2008.

[8] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "Ros: an open-source robot operating system," in *ICRA Workshop on Open Source Software*, 2009.

[9] D. Gossow, A. Leeper, D. Hershberger, and M. Ciocarlie, "Interactive markers: 3-d user interfaces for ros applications [ros topics]," *Robotics Automation Magazine, IEEE*, vol. 18, no. 4, pp. 14–15, Dec 2011.

[10] S. Chitta, I. Sucan, and S. Cousins, "Moveit! [ros topics]," *Robotics Automation Magazine, IEEE*, vol. 19, no. 1, pp. 18–19, March 2012.

[11] R. Gutierrez and J. Craighead, "A native iphone packbot ocu," in *Human-Robot Interaction (HRI), 2009 4th ACM/IEEE International Conference on*, March 2009, pp. 193–193.

[12] P. Steinberg, P. Merguerian, W. Bihrille, J. Heaney, and J. Seigne, "A da vinci robot system can make sense for a mature laparoscopic prostatectomy program," in *Journal of the Society of Laparoendoscopic Surgeons*, Jan-Mar 2008, pp. 9–12.

[13] A. Baheti, S. Seshadri, A. Kumar, G. Srimathveeravalli, T. Kesavadas, and K. Guru, "Ross: Virtual reality robotic surgical simulator for the da vinci surgical system," in *Haptic interfaces for virtual environment and teleoperator systems, 2008. haptics 2008. symposium on*, March 2008, pp. 479–480.

[14] C.-H. King, M. Culjat, M. Franco, C. Lewis, E. Dutson, W. Grundfest, and J. Bisley, "Tactile feedback induces reduced grasping force in robot-assisted surgery," *Haptics, IEEE Transactions on*, vol. 2, no. 2, pp. 103–110, April 2009.

[15] W. McMahan, J. Gewirtz, D. Standish, P. Martin, J. Kunkel, M. Lilavois, A. Wedmid, D. Lee, and K. Kuchenbecker, "Tool contact acceleration feedback for telerobotic surgery," *Haptics, IEEE Transactions on*, vol. 4, no. 3, pp. 210–220, May 2011.

[16] F. Volonte, N. Buchs, F. Pugin, J. Spaltenstein, M. Jung, O. Ratib, and P. Morel, "Stereoscopic augmented reality for da vincii robotic biliary surgery," in *International Journal of Surgery*, Feb 2013, pp. 365–367.

[17] N. Dantam, D. Lofaro, A. Hereid, P. Oh, A. Ames, and M. Stilman, "Reliable software for humanoid robots," in *IEEE Robotics and Automation Magazine*, 2014.

[18] D. Lofaro, R. Ellenberg, P. Oh, and J. Oh, "Humanoid throwing: Design of collision-free trajectories with sparse reachable maps," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, Oct 2012, pp. 1519–1524.