Teleoperation of a humanoid robot using full body motion capture, example movements, and machine learning

Christopher Stanton[1], **Anton Bogdanovych**[1] and **Edward Ratanasena**[2]
[1]MARCS Institute, University of Western Sydney
c.stanton@uws.edu.au, a.bogdanovych@uws.edu.au
[2]University of Technology, Sydney
edward.ratanasena@student.uts.edu.au

Abstract

In this paper we present and evaluate a novel method for tele-operating a humanoid robot via a full body motion capture suit. Our method does not use any *a priori* analytical or mathematical modeling (e.g. forward or inverse kinematics) of the robot, and thus this approach could be applied to the calibration of any human-robot pairing, regardless of differences in physical embodiment due to the human's body, the motion capture device, and/or the robot's morphology. Our approach involves training a feed-forward neural network for each DOF on the robot to learn a mapping between sensor data from the motion capture suit and the angular position of the robot actuator to which each neural network is allocated. To collect data for the learning process, the robot leads the human operator through a series of paired synchronised movements which capture both the operator's motion capture data and the robot's actuator data. Particle swarm optimisation is then used to train each of the neural networks. The results of our experiments demonstrate that this approach provides a fast, effective and flexible method for teleoperation of a humanoid robot.

1 Introduction

Teleoperation is the operation of a machine at distance. An early example of teleoperation was introduced in the 1940s, when Goertz [1] developed mechanical master-slave manipulators for work with radioactive material. More recent examples include the control of wheeled robots for tasks such as bomb disposal and the Mars Rovers. Advances in robotic hardware have seen the emergence of sophisticated humanoid robots, such as Honda Asimo, HRP-4C, Robonaut, and Aldebaran Nao. Humanoid robots have advantages over robots with non-human morphologies (e.g. wheeled robots) in that their human-like form allows for the robot to take advantage of urban environments designed by people for people. For example, humanoid robots are well suited to using human tools, opening doors, climbing staircases, and so forth. The unstructured nature of the real world, coupled with limitations in artificial intelligence and autonomous robot behaviour, means that teleoperation of humanoid robots is a more suitable approach to robot control for some situations where dexterous and complex movements are required in environments too dangerous for humans (e.g. mining disasters, war zones, chemical spills, nuclear accidents, space exploration, etc). Furthermore, giving a robot a humanoid form allows intuitive teleoperational control due to the similarities in embodiment between the human master and the robot slave. Initiatives such as DARPA's [1] robotic challenge in which humanoid robots are used in disaster and rescue situations highlight the potential benefits of teleoperation of humanoid robots.

Our aim is provide a general method of mapping human motions to robots, regardless of the robot's form or the device used for capturing human motion. In this paper we present a system for teleoperating a humanoid robot using a full-body motion capture suit. Our system allows for a human to perform natural and fluid motions, and for the humanoid robot to closely mimic these motions in real-time. A key aspect of our approach is the use of a machine learning process to calibrate the human master with the robot slave, thus eliminating the need for explicit kinematic modeling of the relationships between motion capture data and robot actuator commands. Apart from being difficult to establish the mathematical relationships between human sensory inputs and robot motor angle outputs in general, explicit kinematic modeling is always dependent on the particular type of hardware (both robot and motion capture equipment) and any hardware changes would normally require complete redesign of the system. With our approach, however, the system is easily scalable to vari-

[1]http://www.darpa.mil/

ous kind of robotic and motion capture equipment and making equipment changes would only result a quick retraining of the system, while all the software will remain unchanged.

This paper is structured as follows: Section 2 outlines the problem domain of designing and building user interfaces for teleoperation of humanoid robots. Section 3 presents related literature in the use of motion capture and machine learning for teleoperation tasks. In Section 4, our approach is described. Section 5 highlights our results, and in Section 6 we discuss the implications of our work, and present concluding remarks.

2 Problem Domain

The development of humanoid robots has brought increased research interest in teleoperation. As humanoid robots are bipedal and have a large number of degrees of freedom, the main challenges in teleoperating humanoid robots concern how to best satisfy the operator's desired behaviour for the robot given the (dimensional) differences between the input capture device and the robot, while also maintaining the robot's stability.

Early approaches to teleoperating humanoid robots captured user intention through the use of graphical user interfaces (GUIs), joysticks, buttons and keyboards. For example, Takanobu et al [2] developed a GUI for teleoperating a humanoid robot in Italy from Japan. Buttons, slider bars, and text boxes are used to capture user input to control specific parts of the robot (such as the camera/"eyeballs" and neck), or to trigger pre-defined motions of the arms such as signaling "hello" or "bye". Sian *et al.* [3] control a HRP-1S humanoid robot using joysticks. As the joysticks used to control the humanoid capture a smaller number of inputs than the robots number of degrees of freedom, button presses are used to select the particular motors on the robot they wish to control. Limitations of these approaches include lack of complete and simultaneous control over every DOF (degree of freedom) on the robot, and that complex motions or behaviours need to be pre-defined.

One strategy to allow the user to simultaneously control multiple DOF on a humanoid robot is to increase the robot's autonomy, and to allow the operator to select different modes of operation. By having the robot calculate complex trajectories and kinematics, low dimensional commands can be used to control the robot. For example, Stilman et al. [7] teleoperate a 38 degree-of-freedom HRP-2 humanoid robot used to manipulate a caster wheeled table carrying loads as heavy as 55kg using a single three-axis joystick. Through the use of button presses, the user can decide whether to control the robotc hands, move the robot, or to make the robot hold the table and move it around.

3 Motion Capture and Neural Networks for Teleoperation

3.1 Motion Capture

Motion capture has had a significant impact on robotics. Motion capture has been used for not only teleoperation, but also for improving humanoid locomotion [22] and learning from demonstration [13; 16; 17]. Early approaches to motion capture for teleoperation required large master arms or exoskeletons which impose physical limitations upon the wearer. Later, custom built motion capture systems were built using inertial measurement units, e.g. [14], or flex sensors and photo detectors [23], and shape tape [15]. Most recently, many new motion capture products have come onto the market, ranging from the cheap but somewhat inaccurate (e.g. Microsoft Kinect, Nintendo Wii) to the (relatively) expensive but highly accurate Xsens motion capture suit (described in Section 4.1).

Setapen et al. [21] use motion capture to teleoperate a Nao humanoid robot (with the aim of teaching the robot new skills and motions), using inverse kinematic calculations for finding the mapping between motion capture data and robot actuator commands. Matsui et al. [22] use motion capture to measure the motion of both a humanoid robot and a human, and then adjust the robot's motions to minimise the differences, with the aim of creating more naturalistic movement on the robot. Song et al. [23] use a custom-built wearable motion capture system, consisting of flex sensors and photo detectors. To convert motion capture data to joint angles, an approximation model is developed by curving fitting of 3rd order polynomials.

3.2 Neural networks for teleoperation

The most common approach for teleoperating humanoid robots via motion capture is through the use of kinematic modeling to calculate effector end-points. Forward kinematics refers to the use of equations to compute the position of an end-effector (e.g. a hand) from specified values for the joint parameters (e.g. the angles of the shoulder, elbow, and wrist). Conversely, inverse kinematics refers to the use of the kinematics equations of a robot to determine the joint parameters that provide a desired position of the end-effector. With such approaches, changes in hardware require new analysis and kinematic calculations. Our aim is to remove the need for such calculations.

Artificial neural networks are capable of approximating highly complex nonlinear functions. As such, they are suitable for learning the mapping between the data produced by sensor devices used to teleoperate robots (such as joysticks and motion capture devices) and the actuator command values required to control the robot.

Tejomurtula and Kak [24] demonstrated how neural networks could be used to solve a variety of inverse kinematics problems in robotics, arguing that the benefits of using neural networks to solve these problems include reducing software development labour costs, reducing computational requirements, and that they can approximate solutions to problems where algorithms or rules are not known and cannot be derived.

There are numerous examples of research in which artificial neural networks are trained to control robotic devices. Smagt and Schulten [25] train a neural network to control a rubber robotic arm (designed to resemble a skeletal muscle system). Jung and Hsia [26] use neural networks to fine tune robot input trajectories in simulation. Larsen and Ferrier [27] train a neural network using visual information to map the relationship between the motor position and the pixel location of a large deflection, planar, flexible robot manipulator. Wang and Bai [28] use feed-forward neural networks with back-propagation to improve the positional accuracy of an industrial robot by finding the inverse kinematics of a simulated 2 and 3 link manipulator arms in different configurations, and then using a lookup table to select the appropriate weights at run-time. Neto et al. [18] employ low cost accelerometers attached to a human arm, and use artificial neural networks to recognise gestures for control of an industrial robotic arm. Later, Neto et al. [19] use a similar approach with a Nintendo Wii controller and an industrial robot arm. Morris and Mansor [20] use neural networks to find the inverse kinematics of a two-link planar and three-link manipulator arms in simulation.

The most similar approach to the one presented in this paper is that of Aleotti *et al.* [15], who use neural networks to learn a mapping between the positions of a human arm and an industrial robot arm. In their approach the human operator wears the ShapeTape sensor on an arm and copies a series of pre-programmed robot movements, and a neural network for each DOF is trained using back-propagation to find a mapping between the operator's arm positions and the robot's arm positions. In this work the authors used absolute positions of human arm joints (obtained from the ShapeTape) for training the neural network.

3.3 Limitations of Prior Art

The paper of Aleotti *et al.* [15] reports a relatively small mean error of 8 degrees for the neural network approximating the training examples, but the supporting video and authors' conclusions show that the resulted system seemed to be overall unusable. The resulting movement of the robot was in many situations quite different from the motion demonstrated by the human operator and the similarity between the two was often hard to depict.

The system was trained on absolute hand joint positions resulting from a limited set of poses, where operator tried to perfectly match the pose of the robotic arm. The errors outside of the training examples appeared to be very large, which the authors failed to explain. The use of absolute positions of operator's joints applied significant limitations on the overall usability of such system, as it is required for the human operator to always train and use the system while standing in exactly the same position and orienting all body parts in the same way as in all prior training sessions. As the result of the errors and aforementioned limitations, further continuations of the work on tele-operating robots conducted by the authors and their successors drifted away from machine learning.

3.4 Our Contribution

We made significant progress in comparison to existing work. In contrast to operating a robotic arm with 6 degrees of freedom and a fixed position, the developed system described in this paper allows to successfully teleoperate a mobile humanoid robot with 23 degrees of freedom through full-body motion capture. In contrast to Aleotti *et al.* [15] we use rotational data from human operator joints instead of relying on positions, as a set of joint rotations can be used to uniquely determine/identify any pose of the human, while positions, in many situations, are not enough to capture certain motions (e.g. imagine a human standing upfront and turning the head to the left: there is no position change for any joints, while there is a change in neck rotation).

As we intend to use our approach in disaster recovery scenarios, it is not acceptable to demand for the operator to train and use the system while standing in the same place and always facing the same direction. To overcome this limitation, instead of using absolute positions and rotations - we utilised relative rotations, so that both training and operation of the robot could be performed in a flexible way.

During their training, Aleotti *et al.* [15] used a limited set of poses, which we believe was a major cause of system error. In our system we have replaced poses by aerobics style exercises, where a human had to repeat a number of simple movement demonstrated by the robot. Each movement took a few seconds to perform, but it covered 2-3 repetitions of using the entire range of possible motor movement on the robot end and generated hundreds of data samples for training.

As the result, the developed system was successfully tested on performing a wide range of arbitrary motion, manipulating objects and even walking. The video showing the system in action is available at[2] confirms that

[2]http://www.youtube.com/watch?v=ggLge1Rw2z4

there is little visible error in the resulting robot movement, but the system suffers from some lag. Figure 1 shows the robot mimicking the pose of the human operator.

Further we outline the details of our approach.

4 Our Approach

Our immediate aim is to teleoperate a humanoid robot using full body motion capture, but without the use of any *a priori* analytical or mathematical modeling (e.g. forward or inverse kinematics). Our longer term vision is to develop a seamless method for calibrating any human-robot teleoperation pairing, regardless of differences in physical embodiment due to the human's body, the motion capture device, and/or the robot's morphology. Ideally such a system would allow individual users to quickly and easily tailor their idiosyncratic/chosen movements and gestures for accurate and intuitive control of any robotic system.

In this paper we present and evaluate an approach for teleoperating a humanoid robot which involves training a feed-forward neural network for each DOF on the robot to learn a mapping between sensor data from motion capture and the angular position of the robot actuator to which each neural network is allocated. To collect data for the learning process, the robot leads the human operator through a series of repeated paired synchronised movements which capture both the operator's motion capture data and the robot's actuator data. Particle swarm optimisation is then used to train each of the neural networks. The system is tested using an Aldebaran Nao and an Xsens full body motion capture suit. Empirical results detailing the neural networks' kinematics approximations are presented.

4.1 Equipment, Hardware and System Architecture

Robot

We use an Aldebaran Nao humanoid robot. The Nao is approximately 58cm high, with 25 DOF. See Figure 1. In this experiment we used the "H23" version of the Nao robot, which is specifically designed for robot soccer and does not have working wrists or hands (thus it has 23 DOF)[3].

Motion Capture Suit

As an interface to control the Nao robot we employ a high precision full-body motion capture suit, Xsens MVN3[4]. Only recently motion capture suits similar to

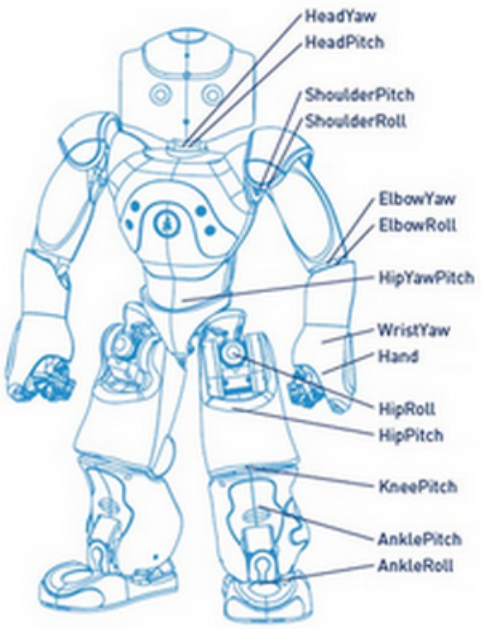

Figure 1: The Nao humanoid robot. The Nao is approximately 58cm high, with 25 degrees of freedom. Picture source: http://www.aldebaran-robotics.com/en/Discover-NAO/nao-datasheet-h25.html

Xsens MVN reached the level of precision when they can correctly capture real-time motion of a human body with no significant data errors. This equipment comes in a form of a Lycra suit with 17 embedded motion sensors, illustrated in Figure 2. The suit is supplied with MVN Studio software that processes raw sensor data and corrects it. It also uses inverse kinematics to cross-verify the data and to estimate the parameters of additional body joints. As the result, MVN studio is capable of sending real-time motion capture data of 23 body segments using the UDP protocol with the frequency of up to 120 motion frames per second. The key elements of the data being transmitted are absolute (X,Y,Z) position of each segment and its absolute (X,Y,Z) rotation.

Figure 2 shows the person wearing the XSENS MVN suit and displays the locations of inertial sensors. Figure 3 outlines the resulting joints that are being computed by the MVN Studio software and used in our experiments.

Architecture

The motion capture suit transfers via wireless communication sensor data to a software application running on a personal/laptop computer. A custom built C++ software application running on the computer performs data preprocessing (detailed in Section 4.2), executes both the training of the neural networks and the operation of the

[3]Note that in our experiments at UTS we used the H23 soccer robot. At UWS we have access to the H25 robot (most photos of robots in this paper are with UWS's H25 model), and future work will likely involve using the robot's hands.

[4]For more information, visit the manufacturer's website http://www.xsens.com/

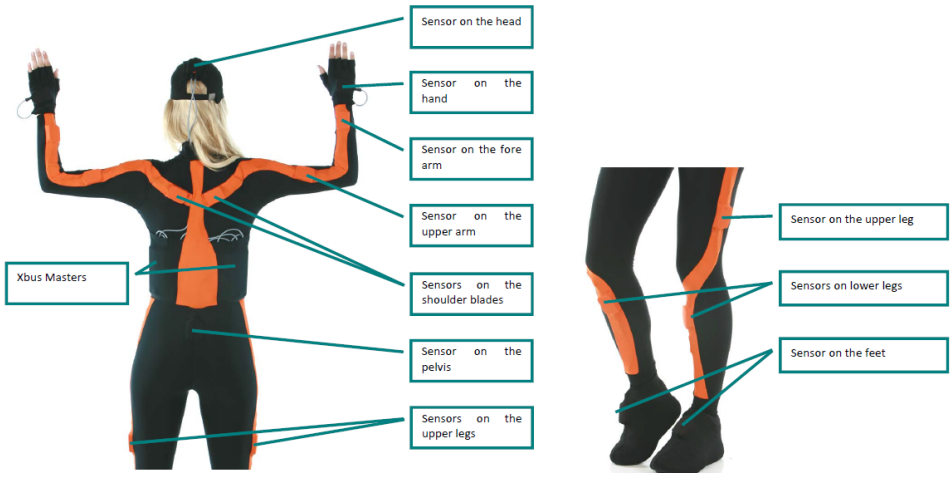

Figure 2: The Xsens full body motion capture suit. Picture source: htttp://www.xsens.com

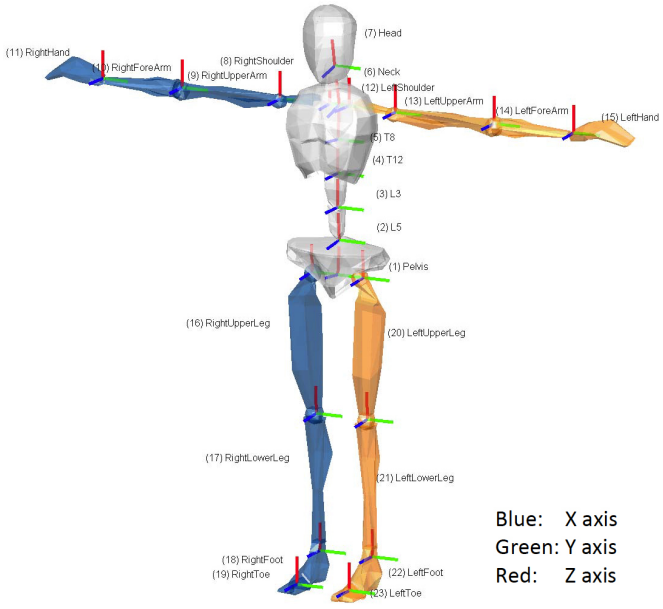

Figure 3: Human body joints used in motion capture. Picture source: XSENS MVN User Manual.

neural networks during teleoperation mode[5]. Communication between the Nao robot and the off board computer is via TCP/IP protocol (both wireless and Ethernet communication was used).

[5]The neural network code could just as easily run on board the robot, as demonstrated by our previous work [29]. For the purposes of rapid prototyping an off board training system was implemented for this research.

4.2 Experimental Setup

Data Collection

To find a mapping between human movement and robot movement, motion data for both robot and human needed to be collected and synchronized. To do this, the robot was preprogrammed to perform a number of slow, symmetric repetitive motions. The human watches the robot, and is asked by the robot's text-to-speech system to copy the robot's movements. Thus the robot, much like a gym or fitness instructor, leads the human through a series of slow, repetitive movements designed to demonstrate the robot's range of physical motion. The human imitates the robot's motion, and both the robot's motors' angular position data and the human's motion capture data is streamed to an off-board computer software application, where it is logged for use in the machine learning process. An example robot motion can be viewed online[6].

The robot's motions were simple to implement, and required specifying a start position, an end position, and a speed of movement. Many of the motions resembled exercises a person would perform during sports like weight training or aerobics. For example, to collect data for the elbow a "bicep curl" is performed, while to collect knee and hip data "squats" are performed. Figure 4 illustrates the start and end points of some of these motions. The motion would be demonstrated by the robot to the user before the data logging process began, which allowed the user the time to find a comfortable, repeatable corresponding motion. When user is ready to imitate the robot, each motion is repeated by the robot for approximately 5 to 10 repetitions (an arbitrary number chosen to balance the competing goals of collecting numerous

[6]http://youtu.be/RgIQPtMIOXg

examples while minimising the data collection time).

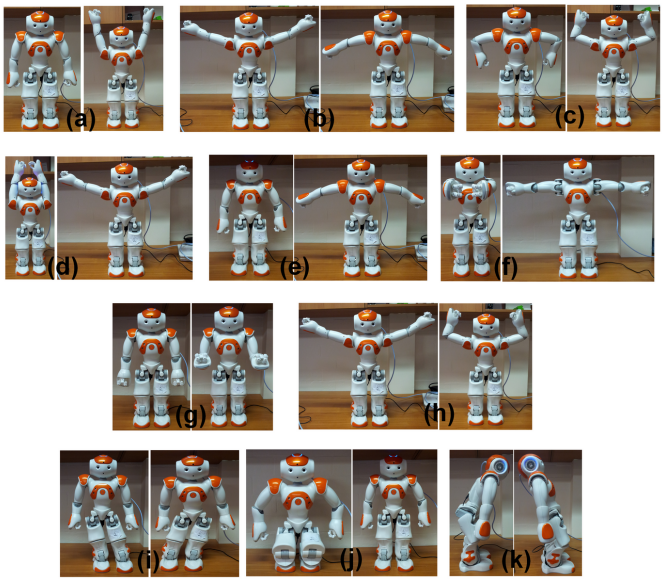

Figure 4: Snapshots of robot motions which the human would mimic during the training process. Each pair of pictures represents the start pose and end pose of the movement. The motors used during the transition from start pose to end pose for each of the motions are as follows: (a), (b) and (c) shoulder pitch; (d), (e), and (f) shoulder roll; (g) and (h) elbow roll; (i) ankle roll, hip roll; (j) ankle pitch, knee pitch, hip pitch (k) hip pitch. An example video of the robot performing the "squat" motion (j) can be found at "http://youtu.be/RgIQPtMIOXg".

During the paired movements between human and robot, the robot's sensor data was logged at 5 frames per second for the entire duration of each repeated motion. To ensure motion capture data and robot sensor data logs are synchronised, a "3, 2, 1" countdown is used before commencing each repetitive motion. A slow but arbitrarily chosen speed was used to allow the user to closely follow the robot's example movements. As with the robot sensor data, motion capture data is logged at 5 frames per second. After the final repetition of each motion is completed, the data from both robot and human is merged into a single log file containing a time series of matched robot motor position sensor data and human motion capture data, the format of which is described in the next section.

Motion capture data preprocessing and transformation to "relative rotations".

The Xsens MVN motion capture suit produces raw data in an absolute coordinate space. That is, its output values are specified in relation to a global origin coordinate

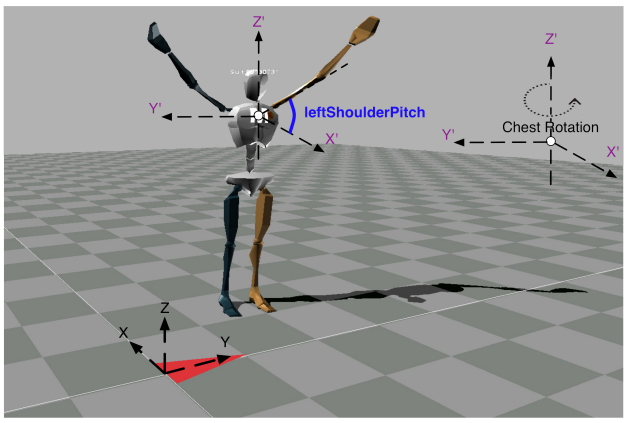

Figure 5: Calculating Relative rotations

in 3-dimensional physical space. In this 3-dimensional space the origin corresponds to the physical world position in which the motion capture suit was last calibrated, Y axis points to magnetic north and Z axis points up. As a consequence if the user is standing in two different locations with the exact same posture, the output values of the suit will be different due to the two different physical locations. Similarly for rotations, orienting the suit operator body differently in space (while holding the same pose) would modify the values of absolute rotations of body joints.

To reduce the complexity of the learning problem, and to allow our teleoperation system to be used regardless of the user's location and orientation, data pre-processing was performed, which involved calculating positions and rotations of the user's body joints relative to other joints. Rotational data was the main source of system input as such data is enough to represent any human pose. Transforming the raw rotational data involved calculating a relative rotation, representing a comparison of two specific joints from the motion capture suit. For example, if we consider the robot's elbow roll motor and want to use motion capture input for training this motor, there is no exact corresponding sensor on the motion capture suit. To approximate an "elbow" sensor value, we compared the relative rotational movements of the forearm sensor and the upper-arm sensor (see Figure 2). Likewise the motion capture's "knee" is derived by comparing the upper leg with lower leg.

Table 1 describes the correspondences between suit input values and robot output values. Please refer to Figure 3 for understanding the inputs from the suit.

Figure 5 outlines the idea behind this preprocessing step. In order for us to obtain the data that corresponds to the leftShoulderPitch motor on the robot, we use the rotational angle of the sensor located on the left elbow of the human operator (labelled as LeftForeArm in Fig-

ure 3), but the rotation of this sensor is computed in the coordinate system represented by absolute rotation of the chest sensor (T8 in Figure 3), shown as (X', Y', Z') in Figure 5. Thus, if operator turns the chest in space, while holding the same pose, it will not affect the relative rotation angle as the angle will be computed in the updated coordinate system set by chest.

All the rotations in the system are represented in the quaternion form [30]. In order to compute relative rotations, we use the following equation:

$$Qrot_{relative}(a,b) = \frac{Qrot_b}{Qrot_a} \quad (1)$$

If we apply the above equation to the example from Figure 5, $Qrot_{relative}(a,b)$ corresponds to the resulting relative rotation (in the quaternion form) of the left elbow sensor (a) in relation to the chest segment (b). The values of $Qrot_a$ and $Qrot_b$ represent the quaternions defining the absolute rotations of each corresponding body segment.

The resulting quaternion rotation (that is shown as headRot in the picture) can be represented as a matrix of the following form:

$$Qrot_{relative}(a,b) = [q_0, q_1, q_2, q_3]^T \quad (2)$$

The values of this matrix are used as the training input of the neural network.

Ensuring Robot Stability

To ensure robot stability, the robot's ankle pitch and ankle roll motors did not employ neural networks for calculating their actuator commands. Instead, the position of the robot's ankles was calculated relative to the position of the robot's hips and knees, so that the robot's feet always remained horizontal/parallel with the ground at all times. In particular, the value of the ankle pitch motor was determined by the value of the hip pitch and knee pitch values of that particular leg. Similarly, ankle roll was determined by the value of the hip roll motor on the same leg of the robot. Lastly, the range of movement of all of the leg motors on the robot was restricted to ensure robot stability. In this initial study, no balance controller used, nor was walking attempted. The robot however was in standing position, and capable of movements such as squatting, shifting weight from one foot to the other, leaning forwards or backwards, and so forth.

Table 1: Relative 3-Dimensional Angular Rotations derived from motion capture data.

Joint Rotations	Sensor 1	Sensor 2
Left Shoulder	LeftUpperArm	T8
Right Shoulder	RightUpperArm	T8
Left Elbow	LeftUpperArm	LeftForearm
Right Elbow	RightUpperArm	RightForearm
Left Knee	LeftLowerLeg	LeftUpperLeg
Right Knee	RightLowerLeg	RightUpperLeg
Left Hip	LeftUpperLeg	Pelvis
Right Hip	RightUpperLeg	Pelvis
Left Ankle	LeftFoot	LeftLowerLeg
Right Ankle	RightFoot	RightLowerLeg
Neck	Head	T8

DOF Restrictions

The Nao robot possesses motors for which there is no corresponding joint on the human body. In particular, Nao has two elbow motors, allowing the robot's forearm to rotate about the elbow in a manner that is impossible for a normal human being. To simplify the learning problem, the Nao's elbow yaw rotation was disabled (but elbow roll is enabled) so the Naos elbow range of motion more closely resembled that of a human's. The Nao also possesses two dependent hip yaw-pitch motors, which allow the robot's knees/feet to point inwards or outwards. However, they are not independent motors; i.e. these motors mirror each other, whereas a human's legs are independent. For this reason the hip yaw-pitch motors were disabled (permanently set at an angular position of "0" radians for all motions).

4.3 Learning

The output of the data collection process is a series of datasets containing the actuator values for every motor actuator on the robot, and for the person a set of relative rotational angles of different parts of the body, as described in Table 1. Initially there is a dataset for every motion performed by the robot, but one new large dataset if formed by merging the datasets. One feed forward neural network is allocated for each actuator on the robot (except those that have been disabled or overridden, as described in previous sections). Each neural network has one hidden layer of 20 neurons. The neural networks are trained to approximate a function which outputs the actuator value (a) based on the relative rotations (q1, q2, q3, q4) of the corresponding sensors on the motion capture suit, i.e. *a = f(q1, q2, q3, q4)*. The function is represented by a matrix of values which represents the weights of each node on each other. These weights are optimised to approximate the function (training) as closely as possible using particle swarm optimisation (50 particles and 50 generations). For this initial work, the neural networks were trained manually offline, rather than on board the robot. The training process for each network is fast (a matter of a few seconds on a standard personal computer). In previous work we have used the same software tools (neural networks trained by PSO) to successfully model non-linear real-time motor data for robot proprioception [29].

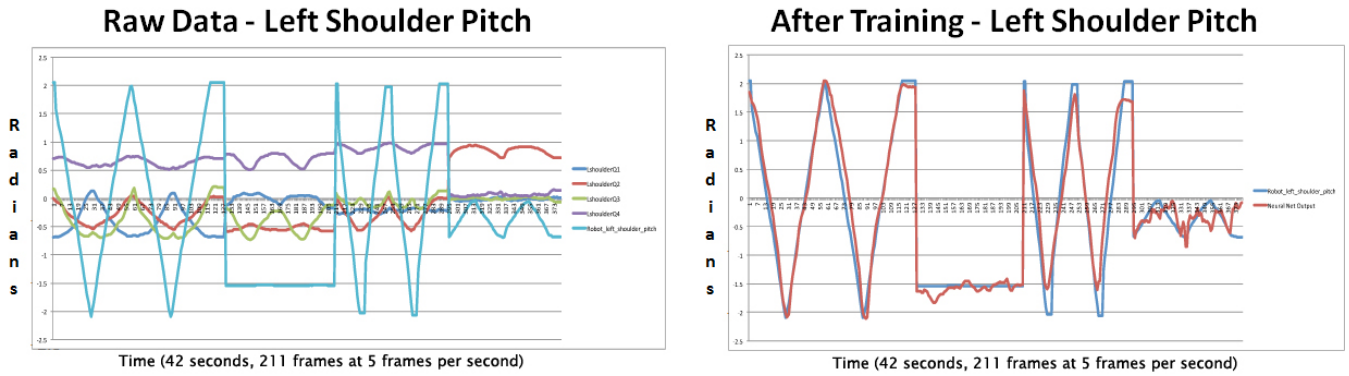

Figure 6: Left Shoulder Pitch training data and neural net approximation for a dataset comprised of 3 different motions.

5 Results

5.1 Data

Examples of neural network approximations of the relationship between motion capture data for a particular human joint and its corresponding robot motor joint are displayed in Figure 6 and Figure 7. Figure 6 represents using relative quaternion rotations for the shoulder sensor in the motion capture data to train the shoulder pitch motor of the robot. Figure 7 shows the input motion capture data and the comparison between the trained neural network and the correct robot motor output for the shoulder roll motor.

As can be seen in all data, a difficulty encountered in the learning process of the neural networks was accurately modeling both large movements and "stillness". For example, in Figures 6 to 7 there are movements/fluctuations in the motion capture data while the robot's equivalent joint is stationary. This may suggest that in approximating the robot's movement, other articulation points on the human's body are being used, and this information is not being provided to neural network in our current model. It also reflects that while it easy for a robot to remain perfectly still, it is very difficult for a human being. Another source of noise is that the user's performance of the individual repetitions of each motion varies considerably, as humans don't have perfect control over their bodies and are unable to repeat movements with perfect precision.

Despite the aforementioned difficulties, the resulting graphs show that the neural network was able to approximate the training motions relatively well. Due to human not being able to hold still poses, we expected slight movement to result in significant approximation errors, but our experiments showed that the mean error of neural network approximation ranged between 4.4 to 7 degrees.

5.2 Experiments

The trained neural network weights are used to control the robot in real-time. Figure 8 shows the tele-operated robot and the human operator testing the system. During our experiments the user tried to perform a number of complex movements, such as picking up cardboard boxes, using the robot's fist to touch and hit objects, and by making dynamic movements such as "shadow boxing". The user had sufficient control of the robot to perform these tasks with almost no visible targeting errors. All movements were perfectly recognisable and most of the time it was hard to determine errors in the robot movement. The majority of visible differences were due to embodiment dissimilarities (e.g. a human being able to lift his arms higher than the robot because of motor restrictions). The system suffered from a small lag on occasions (in order of roughly 0.5 to 1 second), which was further discovered to be resulting from using TCP/IP protocol on the robot end rather than UDP. A video of the results can be found here[7]. Our approach provided a reasonable level of control (as demonstrated in the video). Lastly, the user also attempted to walk, but was only capable of taking small side-steps as some of the foot motors were disabled to ensure the robot doesn't fall over. The main focus of this work was on the upper body and we haven't done much leg training yet.

6 Conclusion

We have constructed a teleoperation system based upon using machine learning to find relationships between paired human and robot example motions. From approximately 10 minutes worth of example paired movements, a time series of motion capture inputs and robot actuator outputs is used by neural networks and particle swarm optimisation to find kinematic mapping functions between the physical pose of the user and the physical

[7] http://www.youtube.com/watch?v=ggLge1Rw2z4

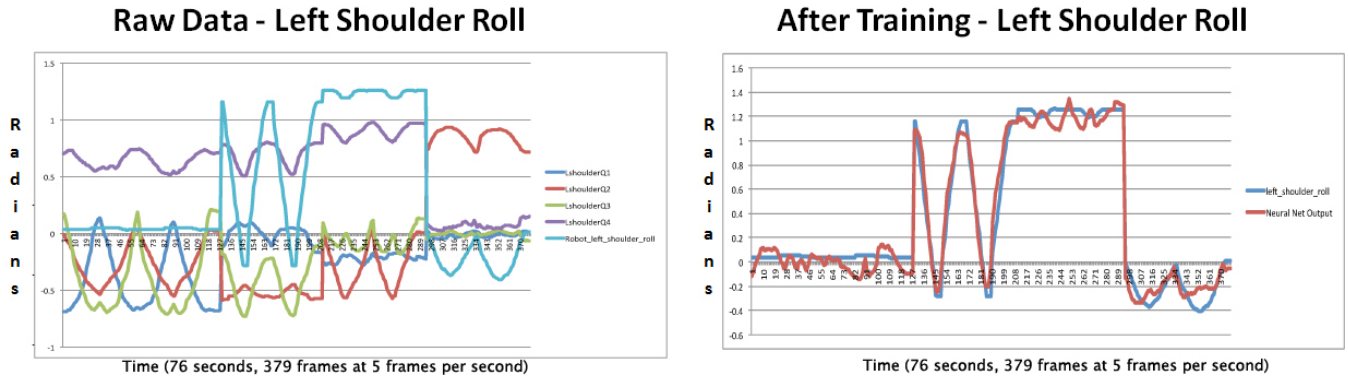

Figure 7: Left Shoulder Roll training data and neural net approximation for a dataset comprised of 3 different motions.

pose of the robot. Initial results suggest that this is a promising general approach to building intuitive and flexible teleoperation systems. Benefits of this approach include lack of robot-specific kinematic modeling, the ability to adapt to the idiosyncrasies of individual users, and that minimal work is required to use the system with new motion capture sensors or new robots. Possible applications include teleoperation of any form, teleoperation of reconfigurable robots, or teleoperation of systems by people with disabilities or unusual human form.

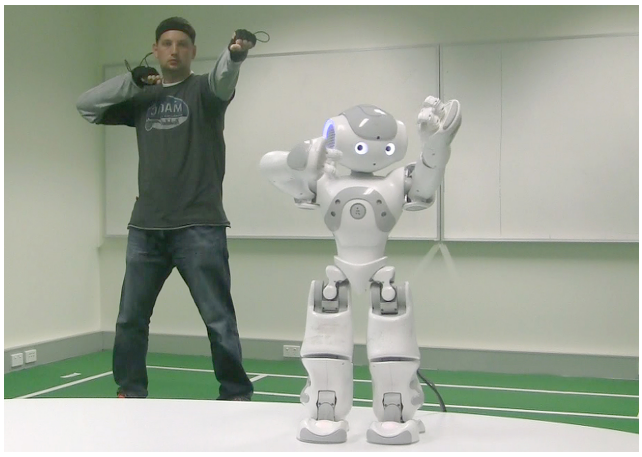

Figure 8: Teleoperating a robot in real time

Acknowledgments

Much of this research was conducted while the first author was an employee of the University of Technology, Sydney.

References

[1] Raymond Goertz and R. Thompson. Electronically controlled manipulator. Nucleonics, Vol.12, No.11, pp.46-47, 1954.

[2] H. Takanobu, H. Tabayashi, S. Narita and A. Takanishi. Remote Interaction between Human and Humanoid Robot. Journal of Intelligent and Robotic Systems 25: 371385, 1999.

[3] Whole Body Teleoperation of a Humanoid Robot - Development of a Simple Master Device using Joysticks. N. Sian, K. Yokoi, S. Kajita, F. Kanehiro, K. Tanie. 2002. Proceedings of the 2002 IEEE/RSJ Intl. Conference on Intelligent Robots and Systems.

[4] A tele-operated Humanoid Robot Drives a Lift Truck. H. Hasunuma, M. Kobayashi, H. Moriyama, T. Itoko, Y. Yanagihara, T. Ueno, K. Ohya, and K. Yokoi. 2002. IEEE International Conference on Robotics and Automation.

[5] A tele-operated humanoid robot drives a backhoe. H. Hasunuma, K. Nakashima, M. Kobayashi, F. Mifune, Y. Yanagihara, T. Ueno, K. Ohya, K. Yokoi. 2003. Proc of the 2003 IEEE International Conference on Robotics and Automation.

[6] A tele-operated humanoid operator. K. Yokoi, K. Nakashima, M. Kobayashi, H. Mihume, H. Hasunuma, Y. Yanagihara, T. Ueno, T. Gokyuu, and K. Endou. 2006. The International Journal of Robotics Research, 206, 25: 593-602.

[7] Humanoid Teleoperation for Whole Body Manipulation. Mike Stilman, Koichi Nishiwaki and Satoshi Kagami. ICRA 2008.

[8] A teleoperation system for a humanoid robot with multiple information feedback and operational modes. L. Zhang, Q. Huang, T. Liu, and Y. Lu. 2005. IEEE International Conference on Robotics and Biometrics.

[9] Cooperation of Humanoid Robots using Teleoperation for Transferring an Object. Muhammad Usman Keerio, Weimin Zhang and Ali Raza Jafri. Interna-

tional Journal of Advanced Robotic Systems, Vol. 5, No. 1 (2008).

[10] Wholebody Teleoperation for Humanoid Robot by Marionette System. T. Takubo, K. Inoue, T. Arai and K. Nishii. Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, October 9 - 15, 2006, Beijing, China.

[11] Task Autonomy for a Teleoperated Humanoid Robot. Kensuke Harada, Hitoshi Hasunuma, Katsumi Nakashima, Yoshihiro Kawai and Hirohisa Hirukawa.Springer Tracts in Advanced Robotics, 2008, Volume 39/2008, 533-540.

[12] Intercontinental, multimodal, wide-range telecooperation using a humanoid robot. Evrard, P., Mansard, N., Stasse, O., Kheddar, A., Schauss, T., Weber, C., Peer, A., Buss, M. pp. 5635-5640. Intelligent Robots and Systems, 2009. IROS 2009.

[13] Whole Body Humanoid Control From Human Motion Descriptors. Behzad Dariush, Michael Gienger, Bing Jian, Christian Goerick, and Kikuo Fujimura. 2008 IEEE International Conference on Robotics and Automation, 2008.

[14] Motion capture from intertial sensing for untethered humanoid teleoperation. N. Miller, O. C. Jenkins, M. Kallman, M. J. Mataric. Proceedings of the IEEE-RAS International Conference on Humanoid Robotics (Humanoids), Santa Monica, CA, Nov. 2004.

[15] Position Teaching of a Robot Arm by Demonstration with a Wearable Input Device. Jacopo Aleotti, Alexander Skoglund and Tom Duckett. Proc. IMG04, International Conference on Intelligent Manipulation and Grasping. 2004.

[16] Robonaut Task Learning through Teleoperation. Richard Alan Peters II, Christina L. Campbell, William J. Bluethmann and Eric Huber. Proceedings of the 2003 IEEE International Conference on Robotics and Automation, p.2806-2811. 2003.

[17] A survey of robot learning from demonstration. Brenna D. Argall, Sonia Chernova, Manuela Veloso, and Brett Browning. Robotics and Autonomous Systems. Volume: 57, Issue: 5, Pages: 469-483, 2009.

[18] Accelerometer-Based Control of an Industrial Robotic Arm. Pedro Neto, J. Norberto Pires, and A. Paulo Moreira. The 18th IEEE International Symposium on Robot and Human Interactive Communication, 2009.

[19] High-level programming and control for industrial robotics: using a hand-held accelerometer-based input device for gesture and posture recognition. Pedro Neto, J. Norberto Pires, and A. Paulo Moreira. Industrial Robot: An International Journal, 37/2 (2010) 137147.

[20] Finding the inverse kinematics of manipulator arm using artificial neural network with lookup table. A. S. Morris and A. Mansor. Robotica (1997) volume 15, pp 617625.

[21] Beyond Teleoperation: Exploiting Human Motor Skills with MARIOnET. Adam Setapen, Michael Quinlan, and Peter Stone. AAMAS 2010 Workshop on Agents Learning Interactively from Human Teachers (ALIHT).

[22] Generating Natural Motion in an Android by Mapping Human Motion. Matsui, D., Minato, T., MacDorman, K.F. and Ishiguro, H. Proc. of Intelligent Robots and Systems, 2005. (IROS 2005).

[23] Tele-operation between Human and Robot Arm using Wearable Electronic Device. HeeBae Song, Doik Kim, Mignon Park and JoonByung Park. Proceedings of the 17th World Congress The International Federation of Automatic Control, Seoul, Korea, July 6-11, 2008.

[24] Inverse kinematics in robotics using neural networks. Sreenivas Tejomurtula and Subhash Kak. Information Sciences 116 (1999) 147-164.

[25] Smagt, P.V.D. and Schulten, K., 1994. Control of pneumatic robot arm dynamics by a neural network. Network, 3, p.1-5.

[26] Neural network reference compensation technique for position control of robot manipulators. IEEE International Conference on Neural Networks, 1996. S. Jung and T. C. Hsia.

[27] Larsen, J.C. and Ferrier, N.J. A case study in vision based neural network training for control of a planar, large deflection, flexible robot manipulator. Proceedings. 2004 IEEE Intelligent Robots and Systems, 2004. (IROS 2004). vol.3, pp. 2924- 2929.

[28] Dali Wang and Ying Bai. Improving Position Accuracy of Robot Manipulators Using Neural Networks. Proceeding of 2005 IEEE Instrumentation and Measurement Technology Conference, pp. 1524–1526, May 2005.

[29] Christopher Stanton, Edward Ratanasena, Sajjad Haider, and Mary-Anne Williams. Perceiving forces, bumps and touches from proprioceptive expectations. In Proc of RoboCup 2011, LNCS 7416, p.377-389.

[30] Kuipers, J.B.. Quaternions and Rotation Sequences. Princeton University Press, Princeton, 1999.