Learning Robot Joint Impedance, focusing on Physical Human Robot Interaction

RC Keely[†]　　Shuhei Ikemoto[†]　　Hiroshi Ishiguro [†‡]

This paper explores applying impedance matching to a robot to maximize the energy transfer, increasing the efficiency of human robot interaction. In contrast with previous work, the system proposed here is a task independent system with no knowledge of the opposing linkage, making the system more flexible. A simulator was built to investigate the problem. It is controlled by a neural network using the dynamic hill climbing algorithm. Several configurations were tested to isolate suitable control parameters.

1 Introduction

Recent years have seen the development of Cognitive Developmental Robotics (CDR) as proposed in [1] and Epigenetic Robotics, as proposed in [22], pushing the idea of embodiment further, towards a more developmentally based approach to robotics. Understanding that the brain is not merely a computer that controls the body, but a carefully balanced biological system, developmental robotics seeks to learn from the brain and from the developmental cycle essential in living creatures to build robots that can learn, based on the models found in nature.

As robotics becomes more mature as a field, robots will need to learn to operate outside of the current controlled environments to which they are currently confined, largely laboratories and factories. In these environments, the humans that the robots interact with are, by and large, trained personnel used to the robot's quirks and idiosyncrasies. Outside of these controlled environments, Human Robot Interaction becomes not only much more likely but even essential when dealing with naive subjects in uncontrolled environments. The task of interacting with naive subjects in an unknown environment can appear, at first, to be almost insurmountable, however human infants face similar obstacles at birth and overcome them. One of the aims of CDR is to build models for robots based on infants. It is hoped that this may make it possible to create a new breed of robots that can overcome these obstacles allowing for an increased integration of robots into society.

Shared use environments have many disadvantages, such as the more stringent safety requirements required and the unpredictable nature of the agents with whom the environment is shared, requiring the use of more complex control. However shared use environments can also have advantages. Physical Human Robot Interaction (hereinafter referred to as Physical HRI), attempts to allow robots to take advantage of the agents with whom the environment is shared, allowing the humans to assist the robot in performing a task and in instructing the robot how best to accomplish said task. One aspect of infants which greatly assists in their subsequent motor learning is the structure of the human muscular skeletal structure and the manner in which an infant's innate reflexes, as discussed in [17] and motor synergies [2] contribute to, and even drive, motor learning. It is felt that the innate compliance of the system is essential in this learning. In an interaction with a caregiver, even if the infant does not know what motion is to be performed, it can still perform motions in coordination with the caregiver because of it's innate compliance.

In general, however, current robots do not have such innate compliance, and it is difficult to implement good compliance for a robot because it tends to be task dependent and, in general, to require *a priori* knowledge of the robots dynamics. These factors make it very difficult to implement a task independent evaluation system based on compliance. However if robots are intended to be used in a shared use environment, it is possible for the robot to learn through interaction with the humans in it's environment, i.e. through Physical HRI. As humans use compliance to teach motions to other humans, it is logical that learning through compliance could be of assistance to robots.

Impedance matching, a concept from electrical science can be applied to Physical HRI because the mechanical and electrical domains are equivalent. The concept of impedance matching should allow a robot to maximize the energy transfer in an interaction and as such increase the efficiency of human robot interaction. Previous work on the application of impedances to ro bots mainly focused on using motions which were known *a priori* to allow interaction with known linkages. In contrast, the system proposed here is a task independent system which has no knowledge of the linkage structure of the opposing system. It is thought that such a system is more flexible. In order to implement this system, a physical simulator was implemented. The simulator was designed such that the processing load can be split across multiple machines and it features high quality visualizations. Control of the simulator is implemented using a feed forward neural network using dynamic hill climbing to learn the output weights. Several configurations are tested to attempt to isolate the most suitable control parameters.The related work section discusses other work that deals with human robot interaction and with work that deals with applications of impedance in robotics as well as its biological basis. The methods section describes the design of the simulator and the learning system used, particularly the learning rule used in the neural network. The experiments section describes the various experiments which were undertaken and the results of those experiments.

2 Related Work

In [5] discusses the various methods available to allow robots to learn skills through interaction with a human caregiver, variously referred to as "Human Robot Interaction", "Robot Programming by Demonstration" and "Human in the Loop Control". In the interactions described in the paper a user demonstrates a motion for the robot and then assists the robot in executing the desired motion. The authors in [3] describe the underlying model for such interactions, taking the example of the primate motor cortex to allow robots to learn tasks by imitating the humans with whom they interact. This

[†]Osaka University
[‡]JST ERATO Asada Project

research applies the ideas of mirror neurons in order to allow the agent to learn how to imitate.

In [6] the authors investigate tactile interaction with a robot in order to directly correct the robot's movements. This has the advantage of being a much more intuitive system than is usually used to generate robot motions, namely a graphical user interface. The research attempts to solve the difficulty that arises in the ambiguous nature of touch by deriving a protocol that generalizes the intuitive gestures of a number of test subjects.

The authors in [11] undertake an analysis of Physical HRI in order to allow for human robot interaction in a shared use environment, using the CB^2 robot. The authors [10] subsequently expanded on this, investigating the behavior of the robot in tasks which require close physical contact with an actively controlled robot. This research stresses the importance of flexible, compliant joints in a shared use environment and the dangers which such an environment can have. Again, in [10] focused on using machine learning techniques to allow the robot to learn how to interact with a human partner. The efficiency of the system is evaluated using an interaction where a human helps the robot to stand.

The Partner Ballroom Dance Robot (PBDR) robot, developed by Kosuge et. al [12], is designed to allow humans to interact with a robot which dances. The robot has a force sensor to allow it to determine the applied force, but the system is designed solely for the desired task, namely ballroom dancing, and all of its postures have been specifically designed for the execution of that task. The nature of the interaction lends itself well to accomplishing the goal, however it seems unlikely that it could be extended far beyond the ritualized nature of a formalized dancing style.

The importance of impedance in the central nervous system is shown in [4]. The authors prove that the brain uses selective control of the mechanical impedances to generate forces to stabilize agains unstable movements. This was shown by the comparison of human subjects moving a manipulator with no force applied versus moving the manipulator through a dynamic force field. After a brief training period, the subjects to exhibit no significant difference between the two trajectories, as they had learned to compensate for the motion of the dynamic field. In [20], the authors build upon previous work[4] to analyze the uses of these impedances in a redundant muscle system, such as the human arm, and how these impedances allow for the selection between different muscle groups to accomplish different tasks.

Applying impedance to robotic motion has been researched to various degrees in previous work. The authors in [14] proposed an impedance matching method for serial link manipulators which modeled the dynamics of the manipulators and calculated the correct compliance for a given task. The work presented is based on earlier work by the authors, [13] which describes a method expressing the manipulator performance of a linkage by means of its dynamic manipulability its manipulating force, together referred to as the inertia matching ellipsoid. The authors [14] expanded upon this earlier work by moving from an inertia matching ellipsoid to an impedance matching ellipsoid. The impedance matching ellipsoid is seen as an expression of the efficiency of the transfer of energy from the actuators to the load on the system. The authors present several numerical examples illustrating their system.

In [18] the authors explore the applicability of using a pneumatic system in robotic physiotherapy. They find the use of a pneumatic system to be advantageous because it has a good power to weight ratio and is backdrivable. The system presented in the paper uses a PID controller to control the system and the system is calibrated by human testing. The system is, as such, incapable of learning and will be suitable only for the designated task.

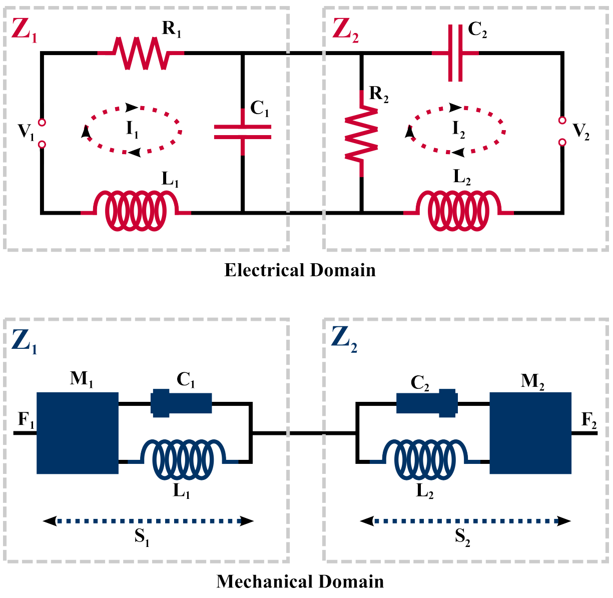


Figure 1: Impedance Matching

3 Methods

In contrast with previous research, the aim of this research is to implement a task independent, compliant learning system. In order to implement such a system, it was felt that the use of impedance matching, a concept borrowed from electrical science, would be beneficial. In impedance matching the energy transfer between two circuits can be maximized if the impedance of the two circuits is matched, i.e. the better matched the impedances are, the more efficient the energy transfer is. The electrical and mechanical domains can be seen as equivalent as shown in Fig. 1, where the Voltage (V) is equivalent to the Force(F), the Current (I) is equivalent to the Displacement (S), the Resistance (R) is equivalent to the Elasticity (L), the Capacitance (C) is equivalent to the Damping (C) and the Inductance(L) is equivalent to the Mass (M). By using impedance matching to distribute compliance throughout all of the joints being actuated at a given time, it should be possible to maximize the energy transfer between the teacher and the learner, increasing the efficiency of the interaction. In this research it is thought that impedance matching in a linkage can be obtained by maximizing the elastic energy of the linkage. For this to be accomplished, the system must learn the function which describes the linkage energy and then find its maximum.

Implementation of the learning is done by means of a fully connected feed forward neural network. The network features bias nodes in the input and hidden layers. The activations of the neural network are hyperbolic tangent functions, scaled in the range $0 \leq \phi(v) \leq 1$. The hyperbolic tangent function was chosen over the sigmoid function as it is believed that the hyperbolic tangent function is more linear. An illustration of the neural network can be seen in Fig. 2.

In Physical HRI, the relationship takes the form of a teacher and a learner. The user, taking the role of

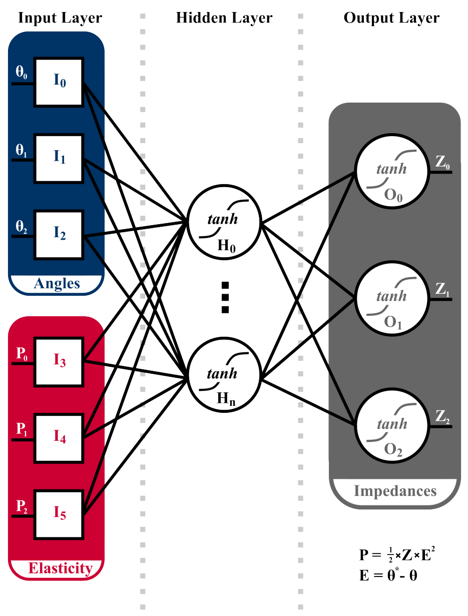


Figure 2: Neural Network

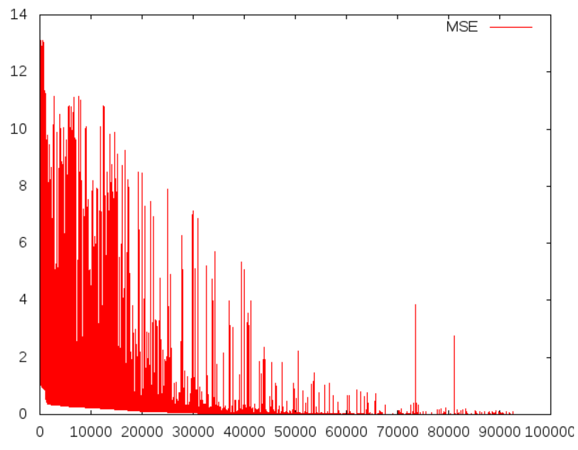


Figure 3: Neural Network Convergence

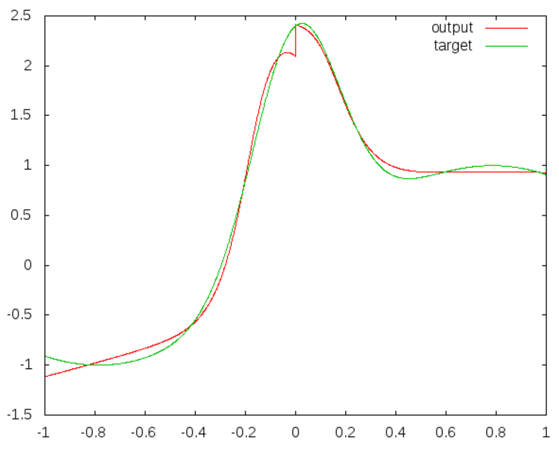


Figure 4: Neural Network Output

the teacher, performs a movement in conjunction with the robot, taking the role of the learner. In this system the aim is to develop a flexible task independent system. As such the robot can assume no *a priori*knowledge of the user who it is interacting with. This implies that all of the impedances must be learned and the learning cannot make be trained by the use of a gradient. Normally in learning problems when there is supervisory data available it is customary to use the back propagation algorithm [19] as the learning rule. Unfortunately the absence of supervisory data requires the use of a different algorithm.

Dynamic Hill Climbing, described in [23] and [9] is a flexible algorithm that combines concepts from hill climbing and from genetic algorithms to solve the problem of optimization. It uses a dynamic co-ordinate frame, which facilitates the finding of ridges that would be incredibly difficult to locate using a standard co-ordinate frame. It uses two loops, an outer loop which selects areas in which to search and an inner loop which uses a set of $2n$ vectors to traverse the search space, where n is the number of dimensions of the search space. The sizes of the vectors are incremented or decremented by a factor of two depending on their success. In Fig 5, an example of DHC applied to a spherical function can be seen. The spherical function is one of the five functions described in [8] as test functions for genetic algorithms:

$$f_1(\vec{x}) = \sum_{i=1}^{N} x_i^2$$

In order to test the combination of a feed-forward neural network and DHC, the learning method was applied to a functional approximation benchmark. The target function was defined as follows:

$$f(x) = \sin(2x) + 2\exp(-16x^2). \tag{1}$$

The area of the function in $-1.0 \leq x \leq 1.0$ was used to approximate the value. Fig. 3 shows the results of the approximation with 5 hidden neurons. It can be seen that the output of the neural network accurately approximates the target function. This indicates that DHC is sufficient to optimize the neural network parameters. The other prospective problem in employing DHC for neural network optimization is the calculation cost. To compare, back propagation needs only $O(W)$ calculations where W indicates the number of weights. Methods based on numerical derivation with parameter perturbation, however, take on the order of $O(W^2)$ calculations to optimize a neural network, indicating that DHC is far slower than back propagation. Fig. 3 shows the convergence of the network. This graph shows that the mean squared error of the functional approximation converges quickly. As such DHC is considered adequate for the problem at hand.

While a sizable majority of robots use servo motor systems, which are not backdrivable due to the fact that small motions which occur at the effector are largely lost to the friction forces within the motor, it is necessary to use either a robot which features direct drive motors or to use a pneumatic robot, such as CB^2. Pneumatic motors have a high degree of non-linearity, making them difficult to work with. As such it was decided that it would be best to initially simulate the robot and to then move to a pneumatic air robot after proof of concept in the simulator. In the development the simulator, several important design decisions had to be made.

It was decided early on that it would be advantageous to include the use of a 3D modeling program in the simulator pipeline. The use of a 3D modeling pro-

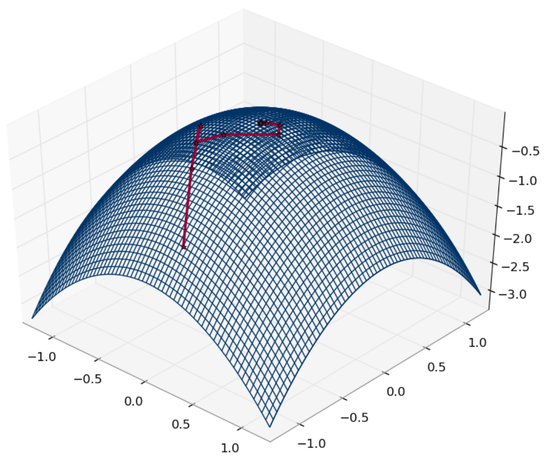


Figure 5: DHC applied to a spherical function

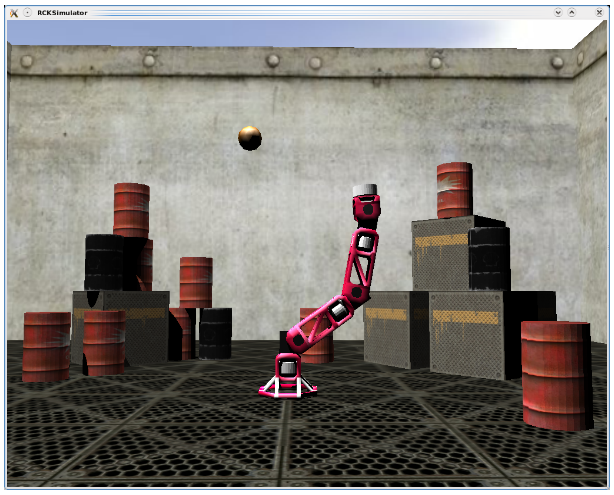


Figure 6: Screenshot of simulator

gram to create meshes for use in a physics simulator has many advantages. Defining the models in the code is a laborious, time consuming process and effectively limits the complexity of the models used to compounds of primitive objects. In a 3D modeling program however, it is possibly to render high quality meshes on par with those used in the latest video games, which improves both the realism and the visual appeal of the simulation.

Blender[1] is thought to be the best choice of 3D modeling program for several reasons. Blender, an open source 3D graphics application was investigated to ascertain its suitability as a basis for a physics simulator. Blender is a mature modeling tool, in development since 1998. Blender is a multipurpose application which enables 3D modeling, animation, texturing and several other features. There are many other 3D modeling programs available, such as 3DSMax and Maya, however these programs are proprietary software, with prohibitively expensive licenses. Blender was also initially a proprietary shareware program, but since 2002 has been distributed under the GNU General Public License. Development of Blender is largely overseen by the Blender Foundation, a not for profit organization based in the Netherlands.

Additionally, because it is open source, and it is possible to script it with Python, export and import scripts can be easily written which allow Blender to both import models built in other 3D modeling programs and to export models to any filetype the user wishes to use. Recent versions of Blender also support Collada.[2] Collada is an open standard, interchange file format, which allows whole scenes to be stored in an xml file which can be passed between different programs. Simulator support of Collada would allow users to quickly and easily rearrange scenes or create new scene in a 3D environment, which could then be easily loaded into the simulator. A screenshot of the simulator can be seen in Fig. 6.

[1]http://www.blender.org

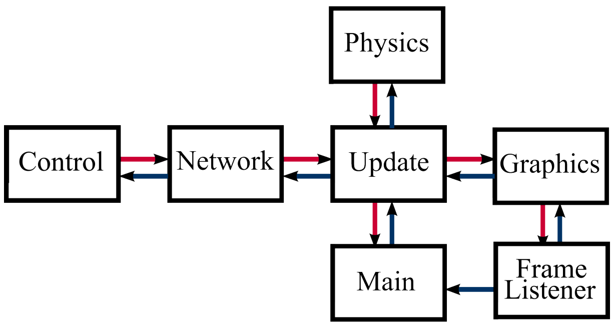


Figure 7: Simulator Design

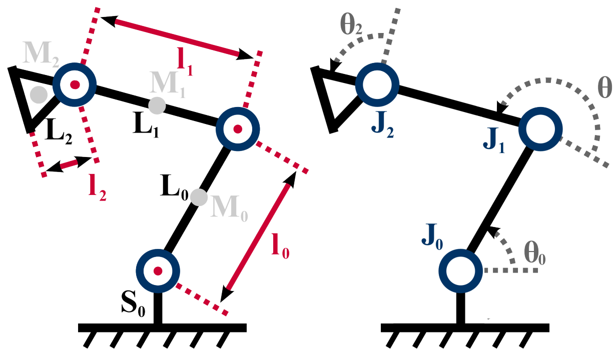
One of the major strengths of simulation over working directly with a robot is the ability to run multiple simulations in parallel on different machines. With this in mind, and paying attention to good software engineering practices, it was felt that designing the simulator to be composed of a number of modules which could be run concurrently on the same machine or distributed throughout several machines throughout a network would be the best design. As such the different aspects of the simulator were divided along functional lines, as can be seen in Fig. 7.

1. Main Module The main module loads the initial simulation configuration from a file. This file contains a list of all of the objects in the simulation, their locations, rotations, graphical properties and physical properties (mass, dimensions etc.) as well as a list of all of the constraints between these objects and the constraint properties (constraint type, initial angle etc.). The configuration file serves to maximize the reusability and flexibility of the simulator, as all that is in order to add additional objects to the simulator or to adjust the properties of the current objects one must simply edit the configuration file. It is envisaged that this configuration file could, in the future, be replaced with a Collada parser to increase its flexibility still further.

   The simulation then initializes the other local modules: the physics module, the graphics module, the listener module, the update module and the network module. The modules are initialized using the values from the configuration file. After it completes the initialization procedures, the main module executes the simulation until instructed to finish.

2. Physics Module: The physics module contains functions for initializing the physical world, helper functions for creating physical objects in that world, as specified by the configuration file parsed by the

[2]http;//www.collada.org



(a) Links (b) Joints

Figure 8: Arm Structure

Link	Length	Mass
S_0	3	
L_0	6	1
L_1	6	1
L_2	3	1

(a) Link Configuration Paramters

Joint	Angular Range	Joins
J_0	$-\frac{\pi}{2} \leq \theta_0 \leq \frac{\pi}{2}$	(S_0, L_0)
J_1	$-\frac{\pi}{2} \leq \theta_1 \leq \frac{\pi}{2}$	(L_0, L_1)
J_2	$-\frac{\pi}{2} \leq \theta_2 \leq \frac{\pi}{2}$	(L_1, L_2)

(b) Joint Configuration Parameters

Figure 9: Configuration Parameters

main module and functions to update the properties of physical objects as each simulation step is executed.

3. Graphics Module: The graphics module contains functions for loading and texturing the meshes used to visualize the physical simulation and creates lights and cameras, necessary for the visualization. It also contains functions for moving the meshes as instructed by the update module. The graphics module can be disabled at compile time to enable users to run the simulation without graphical visualization, which is very useful for running simulations remotely.

4. Listener Module: The listener module allows users to move around the simulator visualization as one would control most 3D video games. This allows for visual debugging and increases the intuitiveness of the simulation. If the graphics module is disabled, the listener module is also disabled.

5. Network Module: The network module allows the control module, implemented in a separate program to communicate with the main simulator. It updates the parameters of the simulation as instructed by the command module and logs information about the simulation to the control module.

6. Update Module: The update module contains functions allowing the parameters of the simulation to be updated, as well as functions for synchronizing the physical simulation with its graphical rendering. The use of an update module greatly simplifies the implementation of the simulator, as the vast majority of the variables which need to be shared between classes are kept in the update module, reducing the likelihood of corrupting the data and simplifying the interactions between the classes.

7. Control Module; The control module is designed to allow fast efficient communication with the simulator from another process, which may be running on a separate machine. This allows for the simulation to easily be run in parallel on multiple machines all of which can be co-ordinated from a single machine. Additionally the communication over a socket allows for greater flexibility in the implementation as the control module, which is separate to the main simulation, can be written in an interpreted language whereas the main simulator, because of the large number of numerical calculations required,needs to be written in a compiled language.

4 Experiments and Discussion

The first obstacle in implementing the basic linkage (a render of which can be seen in Fig. 6 and a diagram in Fig. 8) is the difficulty in implementing soft joints in a physical simulation. The parameters of the linkage can be seen in Fig 9a and 9b. Both Bullet Physics, the physics engine used, and ODE, another similar physics engine, require joints to be controlled using angular velocity which is applied as motors. Unfortunately the documentation for both physics engines is very poor and as such the parameters required considerable tweaking. Bullet also features soft bodies. A brief investigation into the feasibility of implementing the desired functionality through the use of soft bodies was conducted, however the poor documentation and unintuitive API precluded their use in the simulator. A brief description of soft bodies is included in the Appendices.

In order to implement joints which can be controlled it was deemed necessary to implement a PID controller. In a PID controller, or Proportional / Integrative / Derivative controller, the system calculates the difference between the target value and the actual value, known as the error, and attempts to minimize this error by adjusting the values of the control parameters, k_p, k_i and k_d which represent the proportional, integrative and derivative parameters respectively. This allows the simulator control program to specify an angle which the simulator will then attempt to reach by moving the arm according to the parameter values. The parameters are coefficients that multiply the error, the sum of errors and the rate of change of the error to calculate a corrective action.

Initially it was thought that the impedance would correspond to the parameters of the PID controller. To test this hypothesis, the simulator was configured with a simple linkage. A spring connects the end effector of the arm to a body which describes a lemniscate trajectory while the arm attempts to follow a random interpolated trajectory. The system control can be seen in Fig. 10. The target angles generated by the signal generator and the impedances generated by the neural network are fed into the simulator which executes for 360 steps, which is one full revolution of the lemniscate pattern. The error between the target angle and the actual angle is

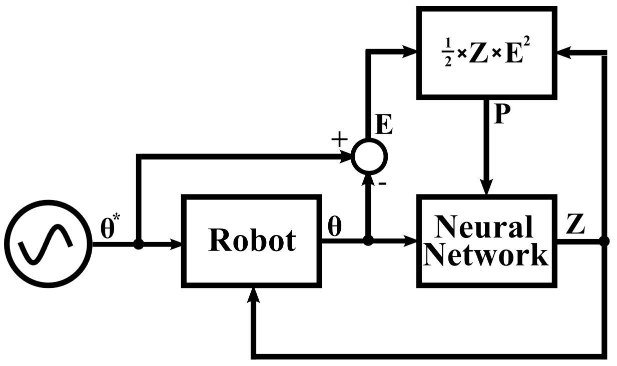


Figure 10: Control Diagram for Initial Experiment

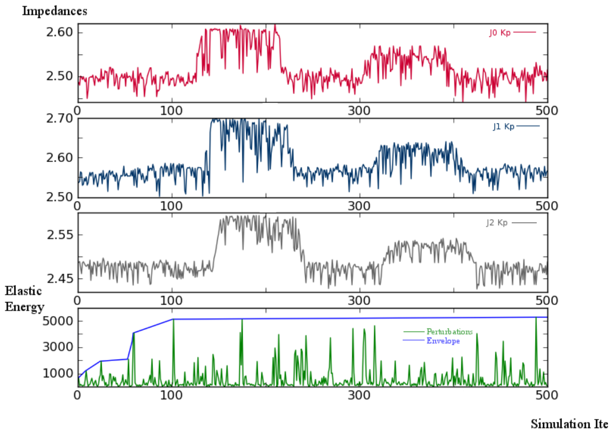


Figure 11: Experiment 1: k_p

recorded for each step, and the average elastic energy of the system is calculated.

Several different values of the parameters were tested. A graph of the system's behavior with just the proportional parameter can be seen in Fig. 11 and of the behavior of the system using all three PID parameters were tested, as can be seen in Fig. 12. On first impressions both graphs look promising, as the behavior of the system is similar, and the elastic energy appears to converge quite quickly. The relationship between the joints expresses itself as expected, the most proximal joint leads middle joint, which in turn leads the most distal joint. However the average energy reveals a flaw in the experimental design. The body which was used to implement the endpoint of the spring was static, in effect an immovable object, as such the energy of the system was not finite. The complications with this configuration led to the design of a second configuration.

The second configuration, attempted to overcome the issues encountered in the first configuration by applying a force directly to the end effector to ensure the force was finite. The force was generated using a benchmark test. During experimentation with this configuration it was discovered that the PID controller parameters would not yield the impedances. Simulations were run with a wide variety of different parameters however the linkage failed to learn the desired impedances. At this point re-examination of the API led to the hypothesis that perhaps rather than the PID parameters, if the maximum impulse of the motor were learned it would possible to implement impedance matching. Several simulations were run with the arm maintaining a fixed posture while the force was applied to its end ef-

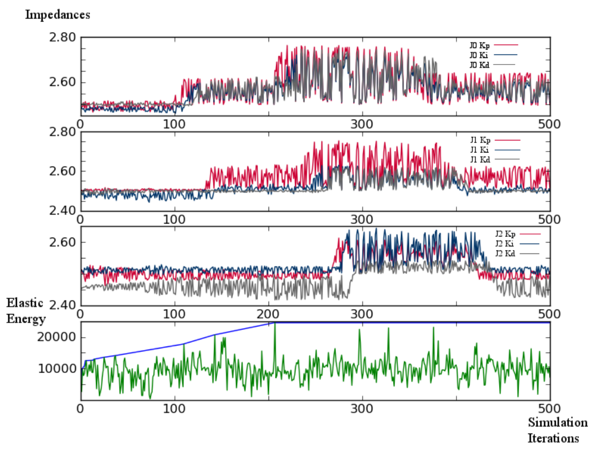


Figure 12: Experiment 2:k_p, k_i and k_d

fector. However the setup of the learning system meant that because the simulator ran for a number of steps and yielded an average elastic energy to the neural network, in this configuration the error became constant.

A third configuration was also briefly investigated. A screenshot of this linkage in the simulator can be seen in Fig. 13. In this configuration, two arms were connected together to allow for a direct implementation of the teacher/learner relationship. However because of the way in which physical simulations are constructed, such a linkage is incredible difficult to configure. If the maximum impulse values are set too high, the simulation error increases to the point where the objects vibrate constantly, if the maximum impulse is set too low, the linkage is not able to move. The range of available motion to such a linkage is very low, meaning that in order to generate meaningful results, much effort would need to put into tweaking the parameters to the correct ranges.

5 Conclusions and Future Work

From the related work, it can be seen that impedance is important for biological systems and impedance matching has been applied to robots. These interactions have, for the most part, been in limited interactions which are largely task dependent and require *a priori* knowledge of the dynamics. The system proposed in this work is task independent and requires no knowledge of the robot's dynamics.

Issues with the way in which constraints are implemented in physical simulation is thought to be a large part of the reason that the experiments performed did not yield conclusive results. Poor simulator documentation was a contributing factor to many of the issues which arose during the course of the experimentation. Additionally it is thought that by redesigning the control module of the simulator to move to a fully online system, allowing the individual elastic energies to be fed into the neural network, rather than the average of a number of trials the simulation would yield useful results because the periodic nature of the applied force averages out over the course of the simulation run. Both the PID parameters and the maximum impulses were investigated separately to determine whether they corresponded to the impedance of the system, as neither yielded conclusive results it may be possible that a combination of the two factors together would operate as the impedance of the system.

The problems which arose with the experiments per-

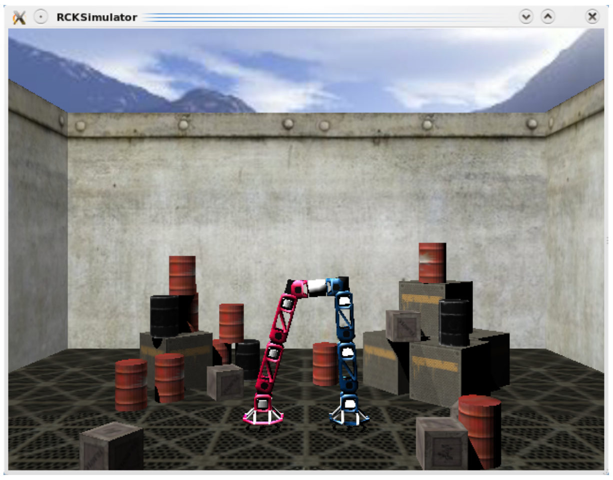


Figure 13: Simulator Screenshot of Two Linkage Simulation

formed are perhaps a sign that because of the nature of the experiments, this problem may not be suitable for simulation. This is thought because the joints used in the simulation require torques to be applied and the metric that was being investigated may be a component of that torque. The simulator itself is robust, flexible and can be applied to many tasks because of the distributed design in combination and the use of Blender to create meshes. The use of the Bullet Physics engine allows for future expansion to GPGPU programming and Ogre provides high quality visualizations. With the addition of a Collada parser, the simulator has the potential to become a very useful platform for simulation of robotics in many areas.

One avenue worth exploring it to implement the system on an arm with pneumatic air actuators as moving to a real robot would remove a large number of the problems encountered during the experimentation phase. It is expected that other issues would arise during the implementation on a real robot, especially given the high nonlinearity of air actuators. If the system could be shown to work on such an arm, then it would seem that a logical extension would be to implement the impedance matching system on a robot with compliant joints. Robots featuring pneumatic air actuators or direct drive motors would be ideal for implementing such a system because of their inherent compliance. One such robot is CB^2. CB^2 is a pneumatic air actuator based robot with 52 degrees of freedom. Implementing an impedance matching system on CB^2 would also allow for comparison in the effects of impedance matching with the methods use in the uprising experiments performed in [11]. There are many advantages in applying the proposed system to a full real robot, the compliance makes the robot much less susceptible to damage, as well as make interactions safer for humans in shared use environments. Additionally, no known learning system can be applied to a real robot in long term learning without limiting the kinds of physical interactions possible. The use of a robot will allow for the learning system to be used in real Physical Human Robot Interactions, something which is not possible within the confines of the simulator. This approach is considered promising, as while the simulator is useful for testing the basic premise of the hypothesis, a simulation is by its very nature a simplified abstraction of the real world, so testing the hypothesis on a real robot in conditions of genuine physical human robot interaction is a far better test of the hypothesis.

Another possible extension to this work is to investigate the possibility of combining an impedance matching system with the idea of freeing and fixing investigated in [16] and [15]. In this work, the possibility of teaching movements to a robot by first freezing all of the joints except the most proximal joint and then freeing the joints one at a time from the most proximal to the most distal. In this manner the robot learns how to perform a motion in a rough approximate manner and with the addition of more distal joints slowly learns to perform the motion in ever more precise approximations. This approach has been observed by humans acquiring new skills, such as infants learning to walk or people learning to ski. It is thought that it would be possible to augment the freezing and fixing with impedance matching and apply it to Physical HRI in order to allow for more intelligent interactions between humans and robots.

It would also be interesting to explore combining reflexes with a compliant learning system. Motor learning in infants has long been know to be heavily dependent on primitive reflexes [17], which have in recent years been seen to be motor synergies with very high initial activations [21]. It is thought that by combining a simple reflex with the compliant linkage, it should constrain the search space allowing for faster learning of motions. In order to investigate this it would be useful to add a simple effector with a grasp reflex, similar to the palmar grasp reflex exhibited by human infants. This grasp effector could be used in conjunction with a compliant joint to get the linkage to grasp objects. Additionally it is thought likely that the use of compliance would allow the arm to grasp soft objects as well as rigid objects. The work reported in [20] shows the importance of impedance matching in a redundant muscular system. It is thought that investigating possible relationships between impedance in the muscular system in conjunction with the work shown in [7] about the construction of behaviors using muscle synergies could lead to interesting results.

References

[1] Minoru Asada, Karl F. MacDorman, Hiroshi Ishiguro, and Yasuo Kuniyoshi. Cognitive developmental robotics as a new paradign for the design of humanoid robots. *Robotics and Autonomous Systems*, 37:185–193, 2001.

[2] Nikolai Bernstein. *The coordination and regulation of movements*. Oxford, Pergamon Press, New York, 1967.

[3] Aude Billard. Learning motor skills by imitation: a biologically inspired robotic model. *Cybernetics and Systems*, 32:155–193, 2001.

[4] Etienne Burdet, Rieko Osu, David W. Franklin, Theodore E. Milner, and Mitsuo Kawato. The central nervous system stabilizes unstable dynamics by learning optimal impedance. *Nature*, 414:446–449, 2001.

[5] Sylvain Calinon and Aude Billard. What is the teacher's role in robot programming by demonstration? towards benchmarks for improved learning. *Special Issue on Psychological Benchmarks in Human-Robot Interaction*, 8, 2007.

[6] Fabio DallaLibera, Takashi Minato, Ian Fasel, Hiroshi Ishiguro, Enrico Pagello, and Emanuele Menegatti. A new paradigm of humanoid robot motion programming based on touch interpretation.

Robotics and Autonomous Systems, 57(8):846–859, 2008.

[7] Andrea d'Avella, Philippe Saltiel, and Emilio Bizzi. Combinations of muscle synergies in the construction of a natural motor behavior. *Nature Neuroscience*, 6:300–308, 2003.

[8] Kenneth A. De Jong. *Analysis of the behavior of a class of genetic adaptive systems*. PhD thesis, University of Michigan, Ann Arbor, 1975.

[9] Michael de la Maza and Deniz Yuret. Dynamic hill climbing. *AI Expert*, pages 26–31, 1994.

[10] Shuhei Ikemoto, Heni Ben Amor, Takashi Minato, Hiroshi Ishiguro, and Benhard Jung. Physical interaction learning: behavior adaptation in cooperative human-robot tasks involving physical contact. In *IEEE RO-MAN 2009*, 2009.

[11] Shuhei Ikemoto, Takashi Minato, and Hiroshi Ishiguro. Analysis of physical human-robot interaction for motor learning with physical help. *Applied Bionics and Biomechanics*, 5:213–223, 2008.

[12] Kazuhiro Kosuge, Tomohiro Hayashi, Yasuhisa Hirata, and Ryosuke Tobiyama. Dance partner robot - ms dancer. In *Proc. IEEE/RSJ Intl. Conf. Intell. Robots and Syst.*, pages 3459–3464, 2003.

[13] Ryo Kurazume and Tsutomu Hasegawa. A new index of serial link manipulator performance combining dynamic manipulability and manipulating force ellipsoids. *IEEE Transactions on Robotics*, 1, 2002.

[14] Ryo Kurazume and Tsutomu Hasegawa. Impedance matching for a serial link manipulator. In *Proc. of the 2004 IEEE Intl. Conf. on Robotic and Automation*, pages 4802–4808, 2004.

[15] Max Lungarella and Luc Berthouze. Adaptivity through physical immaturity. In *Proceedings of the 2nd International Workshop on Epigenetic Robotics*, pages 79–86, 2002.

[16] Max Lungarella and Luc Berthouze. Adaptivity via alternate fixing and freeing of degrees of freedom. In *Proceedings of the 9th International Conference on Neural Information Processing*, pages 482–487, 2002.

[17] Jean Piaget. Piaget's theory. In *Handbook of child psychology*, volume 1. Wiley, New York, 4 edition, 1963.

[18] Robert Richardson, Michael Brown, Bipin Bhakta, and Martin Levesley. Impedance control for a pneumatic robot - based around pole placement, joint space control. *Control Engineering Practice*, 13:291–303, 2005.

[19] David E. Rumelhart, Geoffery E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323:533–536, 1986.

[20] Keng Peng Tee, David W. Franklin, and Mitsui Kawato. Concurrent adaptation of force and impedance in the redundant muscle system. *Biological Cybernetics*, 102:31–44, 2010.

[21] Claes von Hofsten. An action perspective on motor development. *Trends in Cognitive Science*, 8:266–272, 2004.

[22] Juyang Weng, James McClelland, Alex Pentland, Olaf Sporns, Ida Stockman, Mriganka Sur, and Esther Thelen. Autonomous mental development by robots and animals. *Science*, 291:599–600, 2001.

[23] Deniz Yuret and Michael de la Maza. Dynamic hill climbing: overcoming the limitations of optimization techniques. In *Second Turkish Symposium on Artifical Intelligence and Neural Networks*, pages 208–212, 1993.