2010 IEEE International Conference on Robotics and Automation
Anchorage Convention District
May 3-8, 2010, Anchorage, Alaska, USA

Transfer of Skills between Human Operators through Haptic Training with Robot Coordination

Chung Hyuk Park, Jae Wook Yoo, and Ayanna M. Howard

Abstract— **In this paper, we discuss a coordinated haptic training architecture useful for transferring expertise in teleoperation-based manipulation between two human users. The objective is to construct a reality-based haptic interaction system for knowledge transfer by linking an expert's skill with robotic movement in real time. The benefits from this approach include 1) a representation of an expert's knowledge into a more compact and general form by learning from a minimized set of training samples, and 2) an increase in the capability of a novice user by coupling learned skills absorbed by a robotic system with haptic feedback. In order to evaluate our ideas and present the effectiveness of our paradigm, human handwriting is selected as our experiment of interest. For the learning algorithms, artificial neural network (ANN) and support vector machine (SVM) are utilized and their performances are compared. For the evaluation of the performance of the output of the learning modules, a modified Longest Common Subsequence (LCSS) algorithm is implemented. Results show that one or two experts' samples are sufficient for the generation of haptic training knowledge, which can successfully recreate manipulation motion with a robotic system and transfer haptic forces to an untrained user with a haptic device. Also in the case of handwriting comparison, the similarity measures result in up to an 88% match even with a minimized set of training samples.**

I. Introduction

HAPTIC transference of an expert's skill to non-experts is a well-established methodology [1], [10]. By capturing the sophisticated operation of a human expert using high degree-of-freedom haptic interfaces, a user's skill can be stored, analyzed, and transferred to other human operators. The ability of the haptic device to capture and generate forces positions the haptic device as both a mediator and a trainer in the skill transfer process.

What we find common in the various architectures used in this transfer process is that the expert's data is usually not altered – i.e. the ground truth that is used without modification. It has led to a common architecture that primarily copies the expert's data, instead of compiling the generalized representation of the expert's skills. As such, we focus on an approach for manipulating the expert's data for the purpose of compacting and generalizing the haptic data into a form of "haptic knowledge."

The next aspect of the conventional haptic transfer system we examine is the utilization of a robotic system in the transfer process. Although in some cases two identical haptic devices function as a master-slave system, the expert's skill is not always directly identifiable. We propose the bidirectional linkage of a robotic system be coupled with a haptic interaction process that allows more realistic operation and increases understanding of a real-world practice through a direct observation of the robot's performance while operating the haptic device.

Our research is similar in nature to the concept of "learning from demonstration" (also called imitation learning or programming by demonstration) in the robotics community, as we need to both capture human motions as well as mimic/recreate them with a robotic/haptic system. As illustrated in Fig. 1, our intention with this work is to integrate the learning path with the haptic knowledge flow, to increase the efficiency of the haptic training loop and to broaden the modality in human-robot interaction with the haptic pathway.

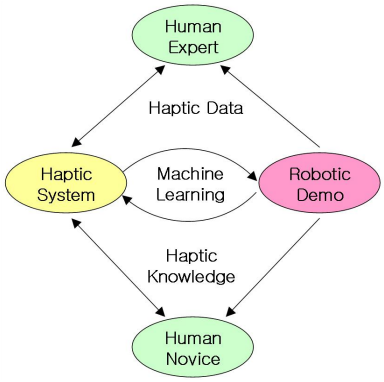

Fig. 1. Our paradigm of robot-coordinated haptic training.

We select human handwriting as the subject of our experiments, since handwriting functions as a good testbed for showcasing the complexity of human dexterity. Handwriting is a research topic that is studied in the haptics area for training applications, and it is also a challenging area for robotics [15], [16]. The reason that handwriting is difficult is that it is a three dimensional task, which increases to six dimensions if changes in orientation are considered. Most importantly, the position and velocity of the writing utensil (or pen) can change abruptly over the time-domain, making it more difficult to generalize the pattern. For example, to write a simple letter 'b', one needs to approach a paper with a pen, make contact with the pen tip, apply a straight down-stroke motion followed by a circular stroke, stop at the same spot as

This work was supported by the National Science Foundation under Award Number IIS-0705130.

C. H. Park and A. M. Howard are with the Human Automation Systems (HumAnS) Lab., School of Electrical and Computer Engineering, Georgia Institute of Technology, U.S.A. J. W. Yoo is with the College of Computing, Georgia Institute of Technology, U.S.A.

E-mail: {chungpark, ayanna.howard, jyoo}@gatech.edu

the end of the straight stroke, then finally lift up the pen from the paper. This simple depiction is representative of the complexity inherent in the motion, variability in velocity, and changes in position associated with a handwriting task.

To successfully tackle the skill transfer issue, we propose a haptically-linked human-robot interactive learning architecture. In Section II, we describe previous research related to this subject matter. We explain the architecture of our system and the algorithms in Section III, and we display the experimental setup in Section IV. We discuss the corresponding results and analysis in Sections V and VI. We then conclude our work with a brief summary in Section VII.

II. Related Works

There are a number of research efforts that have successfully showcased robotic learning of behaviors. Nicolescu and Matarić [2] divided the sequence of tasks being performed by a robot into multiple behaviors and accomplished the learning and generalization process using the longest common subsequence algorithm. Campbell et al. [3], [4] introduced the sensory-motor coordination and behavioral superposition, which enabled a humanoid robot to perform 3D manipulation tasks such as grasping and handling. However, their work required many repeated training samples for each short period of subtasks (e.g. 45 reruns for grasping). Mayer et al. [5] suggested the use of Recurrent Neural Networks (RNNs) with long short-term memory cells for training the robot to do more complex shaped manipulation tasks such as tie-knotting. The drawbacks with this approach are that RNNs are hard to train and are unable to meet the performance criteria with respect to the run-time due to the noise, and only a portion of a subtask is learnable. One research effort that is similar to our proposed work, in terms of human trajectory learning, was conducted by C. Lee in his thesis [6], in which principal curves were collected and regenerated using a spline smoothing method to find the best-fit for a human trajectory data. His work though required the existence of a model to derive principal curves of the trajectory.

Handwriting has been continuously studied in the haptics arena, with the primary purpose of aiding in teaching or rehabilitation, mainly through the use of haptic guidance. Among many haptic guidance related studies, Feygin et al. evaluated the effectiveness of haptic and visual guidance in the training of perceptual motor skills [9], and Liu et al. compared performance associated with combining haptic guidance and visual guidance in training new movements for rehabilitation [10]. Wang et al. developed a Chinese character teaching system using a haptic interface and experimented with sequential training accompanied by haptic and visual guidance which evaluated the trainee's memorization level [11]. Much more work can be found in the past decade with regard to this matter, but the most common characteristic in these efforts is that the expert data is carefully recorded and just replayed in the training process. Even when the data was preprocessed with learning methods, the number of example data was usually high (at least more than10 samples).

In this work, we adopt supervised learning algorithms to process the haptic task performing data provided by a human expert, human-handwriting in this case. The data is then transformed into a generalized haptic knowledge that is both applicable for the haptic system and for the robotic system, and in turn is used to transfer skills to the human novice by combining the output from both systems.

III. System Architecture

There are three primary components that govern our robot coordinated haptic training architecture: the human control loop, the robotic learning loop, and the haptic training loop (Fig. 2). At first, the human teacher tele-operates the robot to perform a task, that is, write a letter in this case. After the robot executes the commands from the human operator, the learning modules are trained over the temporal sequences of the trajectory. Then, the learned modules are used to control the robotic system and to autonomously perform the trained task. During this process, the haptic device simultaneously generates force guidance input through the haptic device to the human novice, synchronizing the robotic motion and the haptic interface.

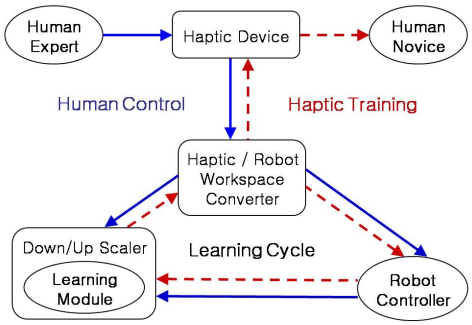

Fig. 2. Teaching data flow for the robot coordinated haptic training system architecture. (Blue arrow: human expert's teaching flow; Red dashed arrow: novice teaching flow with the learning module)

A. Control Flow

In the human control loop, a human expert controls a pen-like haptic device (Phantom Omni) which can receive continuous six-dimensional position inputs over its workspace. The control input is linearly converted to match the differences between the workspace of the input device and the workspace of the robotic manipulator. The mapped pattern then updates the robot controller which transmits back the status of the robot, the position and velocity vectors in Cartesian space, to the learning module.

The role of the learning module during the robotic learning cycle is to train the learning algorithms over the expert's haptic data and generalize it into suitable parameters for subsequent task implementation. In this learning cycle, haptic movement sequences are scaled down to increase the performance of the learning algorithm, and passed to the learning module along with the scaled data of the robot's previous status. The haptic movement sequence is a sampled

trajectory of human handwriting taken over the workspace of the haptic device at every haptic device update cycle. After each haptic operation is finished, the training data is updated and the learning module is trained over the data.

The processes of up-scaling and down-scaling the input data have two merits. First, it transforms the data into a fixed uniform range so any learning module can be used interchangeably. Secondly, since the 3D motions are transformed to fit into a confined volume space, it allows the learning module to be trained over any type of motion, accounting for variability in ranges of movement and velocity constraints, thus minimizing the chance of losing the detailed features that define the task.

Finally in the haptic training loop, the learned module is provided with the robot's previous and current status (which are also scaled down), and its output is scaled up and handed to both the haptic controller and the robotic controller. The output consists of the velocity vector for the next movement, corresponding to the position and velocity of each timestep, and the sequential output vector controls the robot while, at the same time, providing input into the haptic device for generating the guiding forces to make the same trajectory.

B. Learning Algorithms

Learning a three-dimensional data pattern involves predicting the next robot command sequence given its previous sequence and its current position at a certain time. This was studied previously in [14] only without the effect of the time variable. In this work, we trained the learner with and without the time variable as an input and observed the difference in performances.

We investigate the performances derived from implementing the multi-layer feed-forward neural network (NN) and the support vector machine (SVM) as our learning algorithms, which are selected based on the fact that these two supervised learning algorithms possess good generalization capability over continuous range space of input data with high-dimensional characteristics.

1) Neural Networks

For learning in three-dimensional Cartesian space, we first implement the neural network regression algorithm. The time-series data patterns are collected from the human control sequences, and fed into our multi-layer feed-forward neural network (NN) modules to train them recursively over the pattern.

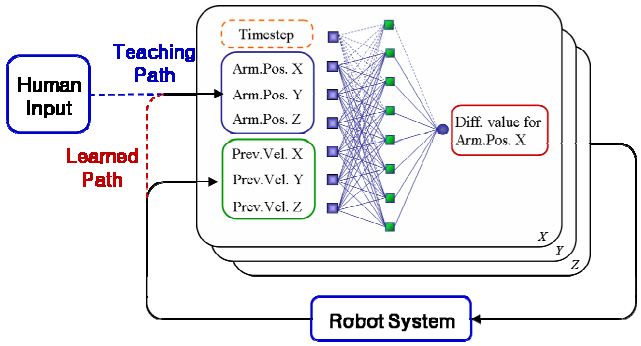

Fig. 3. Training loop by a human teacher and robot's self-writing loop.

As illustrated in Fig. 3, the NN module takes as inputs the current arm's end position in 3D space, the differential values of current position and previous position (representing the velocity of the arm's end position), and the timesteps from the beginning of the sequence. The timestep input is needed to provide additional dimension, since the time value can differentiate between states when there are intersections or circular motions in writing.

We train 3 networks (corresponding to each X,Y,Z space) for each letter dataset. The parameters for the network configuration are primarily based on our *a priori* knowledge, and only a few parameters are changed to evaluate over complex letters. The back-propagation [12] method is adopted to optimize the networks and the training is terminated if the error term reaches a certain allowable criteria—details are discussed in the results section. The networks then go through a preliminary evaluation process and a real-system evaluation in sequence.

2) SVM

SVM [7], [13] is another popular machine learning algorithm that shows good performances in data classification and dimensional reduction. The SVM uses a kernel method to find the best hyperplane between high dimensional training datasets in a relatively fast time compared to other classification algorithms. Since it is crucial in robotic learning problems to guarantee accurate control and strong resilience against disturbances while maintaining short training time, we choose to utilize SVM for our human trajectory regression problem.

We use ε-SVR approach to map our 7 dimensional real valued input data to continuous real valued 3 dimensional outputs, forming a set of SVM classification over continuous output range space. The objectives of ε-SV regression are in a given training set of N instance pairs (x_i, y_i), $i = 1, ..., N$ where x_i in R^N, firstly to find a function f(x) that has at most ε deviation from the actually obtained targets y_i for all the training data, and secondly to make the function as flat as possible at the same time. Specifically, it requires solving the following optimization problem:

$$\min_{w,b,\xi} \frac{1}{2} w^T w + C \sum_{i=1}^{N} \xi_i \quad \text{that is subject to} \tag{1}$$

$$y_i \left(w^T \varphi(x_i) + b \right) \geq 1 - \xi_i, \quad \xi_i \geq 0 \tag{2}$$

where w is the weight coefficient vector, ξ_i the slack variable, and $K(x_i, x_j) = \varphi(x_i)^T \varphi(x_j)$ the kernel function which maps the training vectors x_i to a higher dimensional space by the transformation function $\varphi(x_i)$. Then, SVM is used to find a separating hyperplane with the maximal margin in this higher dimensional space. The kernel functions we use are as follows:

$$\text{Linear kernel: } K(x_i, x_j) = x_i^T x_j \tag{3}$$

$$\text{RBF kernel: } K(x_i, x_j) = \exp(-\gamma \| x_i - x_j \|^2), \gamma > 0 \tag{4}$$

Throughout Eq. (1) to (4), C and γ affect the accuracy of the SVM module most significantly, where C>0 is the penalty parameter of the error term and γ is a kernel parameter. In our experiment, to find good values for C and γ in a reasonable amount of time, we adapted the grid search method [8].

C. Haptic Guidance

Forces for haptic guidance are generated both in the human control cycle and in the haptic training cycle. During the human control cycle, the haptic device creates a passive guidance force feedback which generates potential-like centering forces to the operator's hand position. The primary objective of this guidance is to provide a reliable control environment, since it is difficult for a human operator to maintain a steady position in a 3 dimensional workspace without any support in the device. The haptic device is updated every 10 ms (100 Hz), so as the human operator moves the haptic device, the passive potential force follows the human operator's position and creates continuous holding forces, enabling a passive haptic support.

In the haptic training loop, after the learning cycle, a guiding force field is also applied to the device, except that this time the learning module generates a potential-like force field based on the current position and velocity. In this regard, the haptic force guidance becomes an active guidance toward the next position, thus guiding the human trainee to follow the same trajectory as the human expert - i.e. guidance in writing the same letter.

D. Validation Algorithm (Similarity Measure)

As the last step of the experiment, the robot's writing trajectories are processed to evaluate the robot's performance given the specific human training example. We adopt and implement a LCSS (Longest Common Sub-Sequence) algorithm, which was originally designed to compare common phrases in texts.

We modified the algorithm to measure the differences in two vectors by extracting the longest sequence of similar trajectory slices. This algorithm increases the number of 'hits' if the error between the two trajectory slices is within a certain threshold, and keeps the maximal value of the 'hits' if not. With this method, we measure the similarity between the original pattern and the pattern created by the robot, and provide a computational match between the two writings.

The algorithm is described in detail in Eq. 5. For the comparing of a sequence *A(i)* and an original sequence *B(i)*,

$$LCSS(A(i),B(i)) = \begin{cases} 0, & if\ A(i),B(i) = null \\ 1 + LCSS(A(i-1),B(i-1)), & if\ \Delta_{x,y,z}\left|A(i)-B(i)\right| < \delta_{x,y,z} \\ \max(LCSS(A(i-1),B(i)), & otherwise \end{cases} \quad (5)$$

IV. Experiment

A. System Setup

The robotic platform used for our haptic training system consists of a 5 DoF (degrees-of- freedom) robotic manipulator with a 6 DoF input haptic device, the Phantom Omni. A human operator, or namely the human teacher, manipulates the Omni to control the manipulator to perform a sophisticated task, and a PC connected to the system performs the required calculations such as running the machine learning algorithms.

The data to be learned are constructed as a set of arrays of real valued data taken over time. The human control data consist of a set of trajectories over 3D space, and the related data that we learn are the set of motion vectors corresponding to the trajectories from current positions to the next positions. Computed over the time-series data, the final learned result becomes a function F such that

$$\begin{aligned} &For\ i = 1,6\ \text{, N, N = size of the pattern (total timesteps)} \\ &P_i^C = (p_x^C, p_y^C, p_z^C)_i \ : \text{current position at time } i \\ &V_i^P = (v_x^P, v_y^P, v_z^P)_i \ : \text{previous motion vector at time } i \\ &V_i^N = (v_x^N, v_y^N, v_z^N)_i \ : \text{next motion vector to be taken at time } i \\ &F(P_i^C, V_i^P) = V_i^N \ : \text{the learned function with NN / SVM} \end{aligned} \quad (6)$$

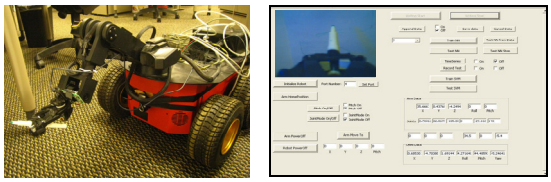

Fig. 4. Pioneer3AT mobile robot with 5DoF robotic arm, and the GUI.

B. Experimental Setup

To test ouralgorithms, we select letters '3', 'b', and a word 'ML'. The numbers '3' and 'b' are chosen since the '3' is written in one continuous stroke consisting of two similar curves, and 'b' is selected since it contains two basic strokes, a straight line and a circle with a sharp turn in between, sharing an intersection. These two letters are thus suitable for evaluating whether the NN and the SVM learning modules are capable of learning spatiotemporal patterns. The word 'ML', abbreviation for 'Machine Learning', is selected to validate if the learning modules can learn to write multiple letters as a whole sequence.

For the SVM training and implementation process, we utilize the LibSVM [8] to implement the SVM learner in our system. LibSVM provides basic SVM functions in an integrated software package for support vector classification, regression, and distribution estimation.

V. Results

A. Preliminary Results

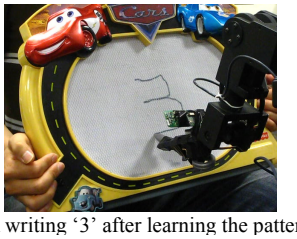

Fig. 5. Robot Arm writing '3' after learning the pattern wth SVM.

After the system was setup, a preliminary experiment was initiated to see whether our haptic system was able to learn a trajectory pattern. One training example of writing a number

'3' was given to the learners, and after training, outputs from the learning modules were observed given the same input data. This was not a cross-validation, and it would require more validation dataset to be a realistic test of performance. However, the primary goal was to check if the algorithms could actually perform regression over complex three-dimensional patterns with only a single training data example.

The results were quite satisfactory, and as expected, the NN module showed a tendency to produce smoother output compared to the initial data, while the SVM tried to follow more details in the pattern as shown in Fig. 6.

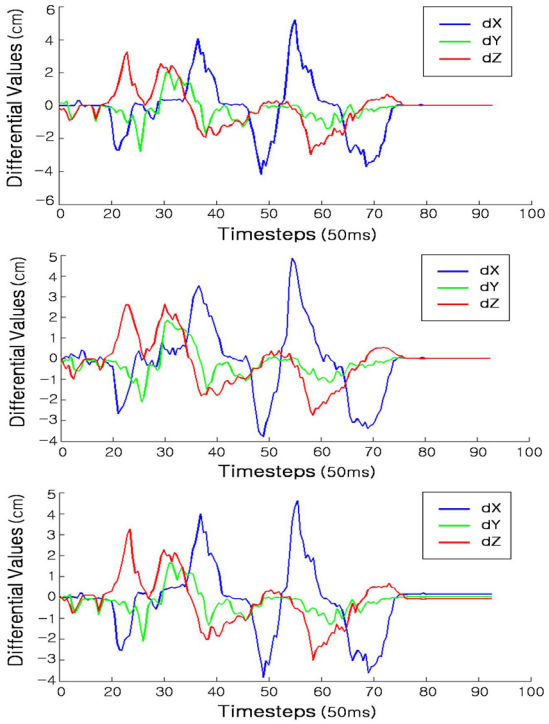

Fig. 6. Top: original data for letter '3', middle: pattern generated by NN for letter '3', bottom: pattern generated by SVR_Linear for letter '3'.

B. *Neural Network Results*

With the NN as a learning module, the commonly used parameters were 7 input nodes, {4,8,12,16} hidden layer nodes, and one output node for each NN structure. Also, the learning rate of 0.15, momentum term of 0.85, error limit of 5%, and iteration count of 3000 were the other parameters used, with minor variances depending on the dataset.

As shown in Fig. 7, the robot learned the pattern of writing '3' with a single input pattern (Fig. 7, left). Although the trajectory became smoother than the human pattern, it promptly regenerated (Fig. 7, center) the two sequential curves for the number '3' as the human had written.

Then, the letter 'b' was demonstrated twice for the robot (Fig. 8, top), and the robot's execution, with the timestep as an input to the NN and without the input parameter of timestep, were recorded. When ran without the timestep information, the robot miscalculated at the first straight down-stroke and drew a slightly bent curve, and after finishing the circle it kept circling again since the velocity and orientation matched again whenever it returned. However, when the time parameter was used as an input vector to the NN, it drew a perfect straight line at the first stage, and reduced speed after drawing one circle (Fig. 8, bottom left). The NN module even resulted in a more smooth and generalized pattern than the original data.

Lastly, 'ML' was provided to the NN learner, and after a few minutes of training, the robot started to write a deformed 'M' with a smoothed out 'L' quickly (Fig. 9).

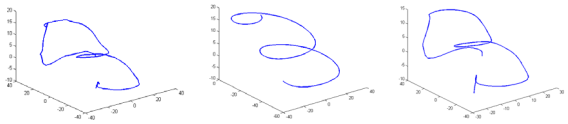

Fig. 7. Letter '3': human writing pattern (left) and robot's writing pattern (center, right). Center: result with NN learner, right: result with SVR learner.

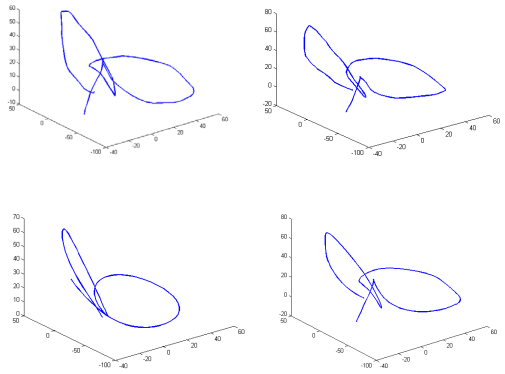

Fig. 8. Letter 'b': human writing patterns (top) and robot's writing patterns (bottom). Bottom left: result with NN learner, bottom right: result with SVR learner.

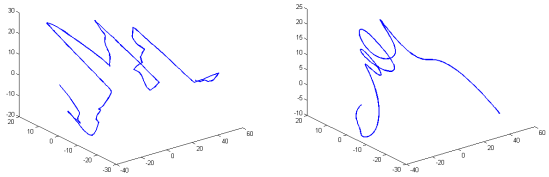

Fig. 9. Original data (left) and the ANN result for the word 'ML' (right).

The only parameter we usually had to change for the NN was the number of hidden layer nodes. The common rule of thumb for the middle layer node number is the average of input and output node numbers, which in our structure is 4. The 4 middle nodes worked fine with connected characters such as 'b' and worked ok with '3', but with more complex letters such as 'M' or 'A', the nonlinearity in the pattern increased and required a larger number of hidden layer nodes. Yet, training over 4 different ranges of middle layer nodes of {4,8,12,16} was enough to find a well trained network.

C. *SVM Results*

In grid search for the SVM learner, a set of (C, γ) pairs are tried over the range and the one with the best cross-validation accuracy is selected. Specifically, the pair that generates the least MSE (mean squared error) is kept per each iterations.

Exponentially growing sequences of C and γ is a well-known method to identify good parameters. In our experiment, the tested C and γ ranges are 2^{-10} to 2^{10}, and they are searched exhaustively. One of the advantages of this approach is that the search can easily be parallelized, because each (C, γ) pairs are independent.

Scaling is also used to derive better performances. The dataset is scaled down before training and scaled up when the learning module is in control. The scaled range is -1 to 1. By means of scaling method, we were able to get two positive results. One, it created more elaborate output data, and two, it dramatically shortened the training time. Especially, when we used a linear kernel for training, the order of training time decreased from minutes to seconds.

As shown in Fig. 7-8, for the characters '3' and 'b', the SVM module showed successful learning with more elaborate drawings, recreating the details of human hand-writing. Especially, when the robot wrote 'b', the SVM learner stopped moving exactly when it finished writing the letter, but the NN module looped the last stroke (a circle) again.

However, for complex patterns such as 'ML', the SVM learner didn't work; the grid search couldn't find a proper parameter combinations and the module couldn't manage to write any letters, while the NN tried to write 'ML' although the letters were smoothed out excessively.

D. Haptic Guidance Result

Haptic guidance forces were generated with a potential-like field, shifting the centers to the sequential points resulting from the learning algorithms (NN / SVM) thus creating a series of guidance for writing a letter. So the difference between the expert's trajectory and the haptic guidance becomes the same as the difference between the expert's data and the data created from the learning algorithms. We leave the discussion of the user study for a future paper, and visualize the resulting haptic guidance trajectory in Fig. 10.

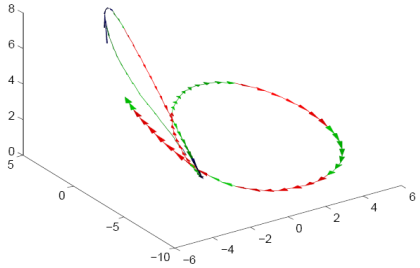

Fig. 10. Haptic guidance for 'b' with NN module. Each arrow represents the change in the center of haptic guidance with the size corresponding to the speed in between guiding points, together with color representation(blue=slow, green=modest, red=fast change).

The clear and smooth trajectory of this writing training sample is due to the good training result of the NN learner, while the actual robot's trajectory in the attached video seems to have unnecessary movements. The reason for this discrepancy lies in the following facts: First, the learning process is performed over the user's haptic data, and the generalized haptic knowledge is recreated as a control input to the robotic and the haptic systems; Secondly, the manipulator with our robotic system is a low-cost educational device, having a slow response-time and a low accuracy movement with back-lashes, thus limiting us from achieving perfect closed-loop control. With these limitations, however, we prove that our algorithms can successfully perform actual learning and transference of the haptic knowledge.

VI. Analysis

A. Data Compressibility

The first factor we looked for, in order to show the effectiveness of applying machine learning algorithms on our problem, was the ability to compress the data. As shown in Table I and Fig. 11, the original pattern data for the writing samples of 'b', '3', and 'ML' were 27kB, 22kB, and 36kB each. However, the trained NNs noticeably downsized them to 2kB, 4kB, and 6kB each, while the SVMs only reduced the size by 2/3, resulting in 17kB, 14kB, and 21kB each.

TABLE I
DATA COMPRESSION RATIO OF ANN AND SVM (UNIT=KB)

Learner	'3'	'b'	'ML'
NN	2	4	6
SVM	17	14	21
Original Data	27	22	36

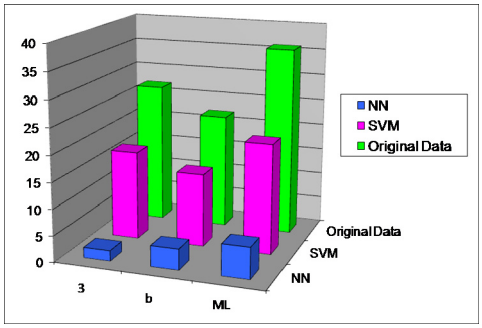

Fig. 11. Data compression ratio of ANN and SVM over 3 datasets.

B. Training Time Comparison

The second feature we observed is the training time. Compared to the NN, which is known for its long training time, the training time for the SVM regression was surprisingly fast (Table II). The NN usually took 10~30 times longer than the SVM's training time, while the SVM usually finished training in a few seconds.

However, the additional fact we gathered is that the SVM learners are very sensitive to the parameters (especially with C and γ) on each dataset, while the NNs usually trained over several datasets with the same parameters. So we had to run grid-search for every datasets we collected, and the time taken for the parameter search summed up with the time for training a single NN, which was around 1 minute.

TABLE II
TRAINING TIME FOR ANN AND SVM OVER 3 DATASETS (UNIT=SEC)

Learner	'3'	'b'	'ML'
NN	57	62	183
SVM	1.39	5.62	9.31

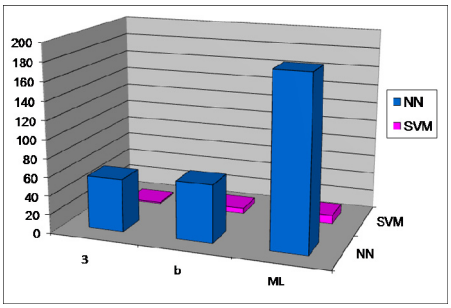

Fig. 12. Training time for ANN and SVM over 3 datasets.

C. Validation

The last parameter that we observed was the measure of similarity, that is, how well the learning module had learned from what the human teacher had taught (and only with a few teaching trials). As shown in Table 3, the LCSS results for the character '3' were 65.7% and 87.7% for each ANN and SVM learners. Although both learners created letter '3' that was legible, the SVM's result were closer to the features of the original data which is the 'human hand writing', and the results from our similarity measure algorithm support that. For 'b', both algorithms created good results, but the SVM's LCSS result was higher, due to the fact that the SVM learned more details such as the crooked circle shape or the edge at the end.

However, both algorithms have limitations as well. When challenged to learn to write a word 'ML', we only achieved 42.9% match from the ANN and failure in training with the SVM.

TABLE III
LCSS RESULT COMPARISON FOR ANN AND SVM

Learner	'3'	'b'	'ML'
NN	65.7 %	83.5 %	42.9 %
SVM	87.7 %	85.6 %	-

VII. DISCUSSION

We have proposed a human-robot coordinated interactive haptic training architecture, in which the demonstration of the robotic system is linked with the haptic control flow and the machine learning algorithms transform the haptic data into the compact haptic knowledge. Results show that our system successfully creates generalized haptic knowledge from only 1 or 2 training examples, and the system is capable of transferring the haptic knowledge both through the haptic force guidance as well as through the robotic demonstration.

As expected, the SVM results in learning more detailed and accurate strokes than the NN, since the SVM algorithm tries to find more optimal classifiers by keeping specific features. However, the NN exhibits better results if the sequence gets more complex, due to the NN's powerful capabilities in generalization.

The next step we are planning to take is to apply this learning process to other human-motor skills, such as object manipulation or basic exercise patterns. To expand the area of application, we will also expand the modalities of haptic feedback. The enhanced haptic learning process will be able to increase the performance of haptic training, such as medical training system and skill transfer for the visually impaired. Future research will also focus on applying our methodology to robot-assisted rehabilitation and training.

REFERENCES

[1] Y. Yokokohji, R. L. Hollis, T. Kanade, K. Henmi, and T. Yoshikawa, "Toward Machine Mediated Training of Motor Skill," in *5th International Workshop on Robot and Human Communication*, Nov. 1996, pp. 32-37.

[2] M. N. Nicolescu and M. J. Matarić, "Natural methods for robot task learning: Instructive demonstrations, generalization and practice," in *Proceedings of the Second International Joint Conference on Autonomous Agents and Multi-Agent Systems*, 2003, pp. 242-248.

[3] C. L. Campbell, R. A. Peters II, R. E. Bodenheimer, and W. J. Bluethmann, "Superpositioning of Behaviors Learned Through Teleoperation," *IEEE Transactions on Robotics*, Vol.22, No.1, Feb.2006.

[4] R. A. Peters and C. L. Campbell, "Robonaut Task Learning through Teleoperation," in *Proceedings of the IEEE International Conference on Robotics & Automation*, Sept. 2003, Vol.2, pp. 2806-2811.

[5] H. Mayer, F. Gomez, D. Wierstra, I. Nagy, A. Knoll, and J. Schmidhuber, "A System for Robotic Heart Surgery that Learns to Tie Knots Using Recurrent Neural Networks," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robot and Systems*, Beijing, China, Oct. 9 - 15, 2006, pp. 543-548.

[6] C. Lee, "Learning Reduced-Dimension Models of Human Actions," Ph.D. dissertation, tech. report CMU-RI-TR-00-17, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, May 2000.

[7] V. Vapnik, *The Nature of Statistical Learning Theory*. New York: Springer-Verlag, 1995.

[8] C. C. Chang and C. J. Lin. LIBSVM: a library for support vector machines, 2001. Available: http://www.csie.ntu.edu.tw/~cjlin/libsvm

[9] D. Feygin, M. Keehner, and F. Tendick, "Haptic Guidance: Experimental Evaluation of a Hpatic Training Method for a Perceptual Motor Skill," in *Proceedings of 10th International Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems,* Orlando, FL, Mar. 2002, pp. 40-47.

[10] J. Liu, S. C. Cramer, and D. J. Reinkensmeyer, "Learning to perform a new movement with robotic assistance: comparison of haptic guidance and visual demonstration," *Journal of NeuroEngineering and Rehabilitation*, Vol. 3, Issue 20, Aug. 2006.

[11] D. Wang, Y. Zhang, and C. Yao, "Machine-mediated Motor Skill Training Method in Haptic-enabled Chinese Handwriting Simulation System," in *IEEE International Conference on Intelligent Robots and Systems*, Beijing, Oct. 2006, pp. 5644-5649.

[12] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," *Parallel Distributed Processing: explorations in the microstructure of cognition*, Vol. I, pp. 318-362, MIT Press, 1986.

[13] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*, Cambridge University Press, 2000.

[14] S. Remy, C. H. Park, and A. M. Howard, "Improving the Performance of ANN Training With an Unsupervised Filtering Method," in *International Joint Conference on Neural Networks*, 2009, pp. 2627-2633.

[15] V. Potkonjak, S. Tzafestas, D. Kostic, G. Djordjevic, M. Rasic, "The Handwriting Problem [Man-machine Motion Analogy in Robotics]," *IEEE Robotics & Automation Magazine*, Vol. 10, Issue 1, pp. 35-46, March 2003.

[16] V. Potkonjak, D. Kostic, S. Tzafestas, M. Popovic, M. Lazarevic, G. Djordjevic, "Human-like behavior of robot arms: general considerations and the handwriting task Part II: the robot arm in handwriting," *Robotics and Computer Integrated Manufacturing,* pp. 317-327, Vol. 17, 2001.