28-11-2005

Application of Competitive Clustering to Acquisition of Human Manipulation Skills

S. Dong
University of Wollongong, shen@uow.edu.au

F. Naghdy
University of Wollongong, fazel@uow.edu.au

Application of Competitive Clustering to Acquisition of Human Manipulation Skills

Abstract

The work carried out to explore the feasibility of reconstructing human constrained motion manipulation skills is reported. This is achieved by tracing and learning the manipulation performed by a human operator in a haptic rendered virtual environment. The peg-in-hole insertion problem is used as a case study. In the developed system, position and contact force and torque as well as orientation data generated in the haptic rendered virtual environment combined with a priori knowledge about the task are used to identify and learn the skills in the newly demonstrated task. The data obtained from the virtual environment is classified into different cluster sets using a competitive fuzzy clustering algorithm called Competitive Agglomeration (CA). The CA algorithm starts with an over specified number of clusters which compete for feature points in the training procedure. Clusters with small cardinalities lose the competition and gradually vanish. The optimal number of clusters that win the competition is eventually determined. The clusters in the optimum cluster set are tuned using Locally Weighted Regression (LWR) to produce prediction models for robot trajectory performing the physical assembly based on the force/position information received from the rig. A background on the work and its significance is provided. The approach developed is explained and the results obtained so far are presented.

Disciplines

Physical Sciences and Mathematics

Publication Details

This article was originally published as: Dong, S & Naghdy, F, Application of Competitive Clustering to Acquisition of Human Manipulation Skills, International Conference on Computational Intelligence for Modelling, Control and Automation 2005 and International Conference on Intelligent Agents, Web Technologies and Internet Commerce, 28-30 November 2005, 2, 1092-1097. Copyright IEEE 2005.

Application of Competitive Clustering to Acquisition of Human Manipulation Skills

Shen Dong*
Fazel Naghdy*
(*) School of Electrical, Computer and Telecommunication Engineering,
University of Wollongong, NSW, 2500, Australia
{sd99, fazel}@uow.edu.au

Abstract

The work carried out to explore the feasibility of reconstructing human constrained motion manipulation skills is reported. This is achieved by tracing and learning the manipulation performed by a human operator in a haptic rendered virtual environment. The peg-in-hole insertion problem is used as a case study. In the developed system, position and contact force and torque as well as orientation data generated in the haptic rendered virtual environment combined with a priori knowledge about the task are used to identify and learn the skills in the newly demonstrated task. The data obtained from the virtual environment is classified into different cluster sets using a competitive fuzzy clustering algorithm called Competitive Agglomeration (CA). The CA algorithm starts with an over specified number of clusters which compete for feature points in the training procedure. Clusters with small cardinalities lose the competition and gradually vanish. The optimal number of clusters that win the competition is eventually determined. The clusters in the optimum cluster set are tuned using Locally Weighted Regression (LWR) to produce prediction models for robot trajectory performing the physical assembly based on the force/position information received from the rig. A background on the work and its significance is provided. The approach developed is explained and the results obtained so far are presented.

1. Introduction

The effectiveness of computer simulation can be augmented using haptic rendering. A haptic interface, or force feedback device increases the quality of human-computer interaction by accommodating the sense of touch in computer simulation. It provides an attractive augmentation to visual display and significantly enhances the level of immersion in a virtual world. Haptic interface has been effectively used in a number of applications including surgical procedures training, virtual prototyping, control panel operations, hostile work environments and manipulation of materials.

The primary focus of this paper is on the development of a haptic rendered virtual model for the peg-in-hole assembly and application of the state recognition method to the classification of skills acquired from the virtual environment.

A haptic rendered virtual modeling is used as part of a new paradigm for programming robotics manipulator to perform complex constrained motion tasks. The teaching of the manipulation skills to the machine starts by demonstrating those skills in a haptic-rendered virtual environment.

The study is conducted in the context of the peg-in-hole assembly process, which is often taken as a standard assembly problem. It concisely represents a constrained motion force sensitive manufacturing task, with all the attendant issues of jamming, tight clearances, and the need for quick assembly times, reliability, etc.

In the developed system, a human operator demonstrates both good and bad examples of the desired behaviour in the haptic virtual environment. Position and contact force and torque data as well as orientation generated in the virtual environment combined with a priori knowledge about the task is used to identify and learn the skills in the newly demonstrated tasks and then to reproduce them in the robotics system. The robot evaluates the controller's performance and thus learns the best way to produce that behaviour.

The data obtained from the virtual environment is classified into different cluster sets using Competitive Agglomeration (CA) algorithm [1]. The optimal number of clusters that win the competition is determined when the fuzzy prototype-based object function is minimized. Fuzzy maximum likelihood estimation algorithm (FMLE) [2], which can divide given data set into different shapes and sizes, is used as a distance measure in the fuzzy prototype-based object function.

Each cluster represents a contact state between the peg and the hole. The data in the clusters consists of fourteen different parameters including force and position variables. The method has been successfully applied in the past to pattern recognition. The clusters in the optimum cluster set are tuned using Locally Weighted Regression (LWR) to produce prediction models for robot trajectory performing the physical assembly based on the force/position information received from the rig.

The rest of the paper is organized as follows. The development platform employed in the work is explained and the algorithms developed to perform the peg-in-hole insertion with a 6 DOF haptic device are described in Section 2. The assembly state classification training by Competitive Agglomeration (CA) algorithm combined with Fuzzy Maximum Likelihood Estimation (FMLE) algorithm is explained in Sections 3. The employment of the locally weighted regression (LWR) on skill acquisition from the virtual environment is presented in section 4 and the physical manipulation results are provided in Section 5. Finally, some conclusions on the work are drawn and suggestions for future work are provided in Section 6.

2. Haptic rendered virtual environment

The data used by the robot to acquire basic manipulation skills is generated from a haptic rendered virtual environment. This environment consists of a six degree-of-freedom (6DOF) haptic device PHANToM Premium 1.5 and Reachin API along with VRML and Python, which are used to construct the virtual haptic manipulation environment of the peg-in-hole insertion.

The PHANToM family of haptic devices, manufactured by SensAble Technologies (Boston, MA), is currently the most widely used force feedback interface on the market. The PHANToM has 3 or 6 degree-of-freedom (DOF) and it can provide wrist up to shoulder motion depending on the model. The 6 degree-of-freedom (6DOF) haptic device, PHANToM Premium 1.5, provides torque feedback in addition to force display within a large translation and rotational range of motion, and provides the user with the much needed dexterity to feel, explore and manoeuvre around other objects in the virtual environment. A PHANToM can produce a maximum transient force of up to 22 N, and a sustained force of 3 N.

In the six degree-of-freedom (6DOF) device, the maximum torques generated is 670 mNm, being produced by actuators placed in the handle. The produced continuous torque is 104 mNm [3]. The characteristics of the PHANToM make it well suited for point interaction, for example, operated by a single virtual finger, a pencil or a peg.

ReachIn API is a C++ application programming interface for creating multi-sensory applications. It can handle the complex calculations required for the touch simulation and the synchronization with graphic rendering, freeing the user to focus on more important issues such as developing application behaviour or experimenting with haptic algorithms [4].

VRML and Python are integrated with ReachIn's own force/torque rendering technology in this project. The graphic model is constructed by VRML-based scene-graph and application's behaviour is described by using ReachIn's unique event-based programming model and Python script code [4].

Fig. 1 shows the developed virtual environment. The peg is a dynamic rigid object in the haptic rendered virtual environment. The force and torque reacted to the peg are transferred to PHANToM Premium 1.5 through the spring damper system. The hole is static in the environment while the peg can be translated and rotated.

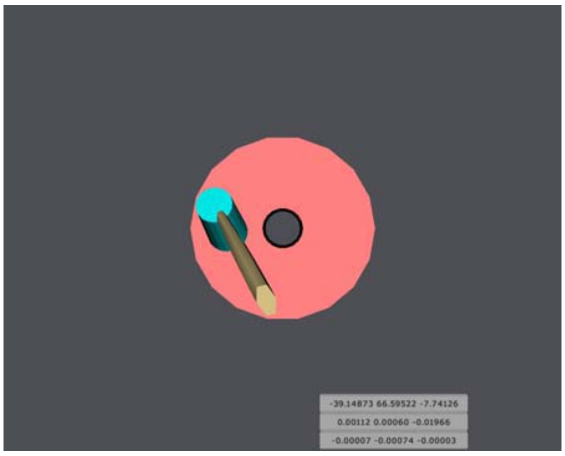


Fig. 1. 6DOF Peg-in-hole haptic virtual environment

The haptic rendered model of the peg-in-hole assembly insertion generating force and torque data is constructed using the virtual proxy method [5]. The virtual rigid peg is defined as a virtual proxy and controlled by the physical ReachIn probe in the haptic rendered virtual environment. The position of the virtual proxy is changed according to alteration in the probe's position.

Fig. 2 illustrates the motion of the virtual proxy. In the absence of an obstacle, the virtual peg moves directly towards the hole. When the peg encounters an obstacle or the surface of the hole, direct movement is impossible. The operator can still reduce the distance of the peg relative to the goal by moving the peg along one or more of the constraint surfaces of the obstacles or the holes. The motion is chosen to locally minimize the distance relative to the goal. The proxy stops when it is unable to decrease its distance relative to the goal for reasons such as jamming.

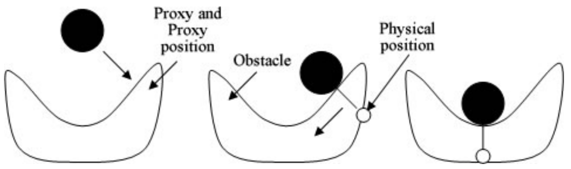


Fig. 2. Motion of the virtual proxy [5]

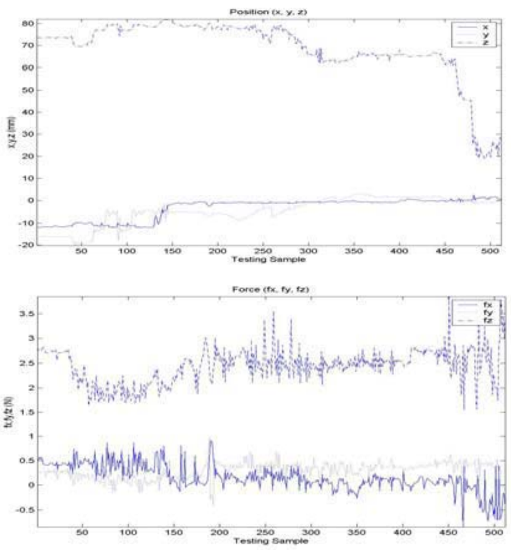
The surfaces of the peg and the hole are constructed using polygons. This results in numerical errors, which produce gaps in the common edge of the peg and hole. The size of the virtual proxy is chosen large enough to prevent it from falling into the gaps.

The force generated at the proxy is the sum of the Coulomb and the friction forces. The generated torque is the product of contact force vector applied at the contact point and the distance vector from the contact point to the rotating centre of the object [6], [7]. The full rotation of the proxy is recorded as (f_x, f_y, f_z, θ). This describes an arbitrary rotation about an axis vector, where (f_x, f_y, f_z) are the axis vectors and θ is the angle in radians in the right-handed direction. The axis vector is of unit length. Rotation matrix is calculated by:

$$Rot(f,\theta)=\begin{bmatrix} f_x f_x v\theta + c\theta & f_y f_x v\theta - f_z s\theta & f_z f_x v\theta + f_y s\theta & 0 \\ f_x f_y v\theta + f_z s\theta & f_y f_y v\theta + c\theta & f_z f_y v\theta - f_x s\theta & 0 \\ f_x f_z v\theta + f_z s\theta & f_y f_z v\theta + f_x s\theta & f_z f_z v\theta + c\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Where $v\theta = 1-\cos\theta$, $c\theta = \cos\theta$, $s\theta = \sin\theta$

Fig. 3 shows the position, force and torque data as well as change in the rotation angle from the last step to the current, obtained from the haptic rendered virtual environment.



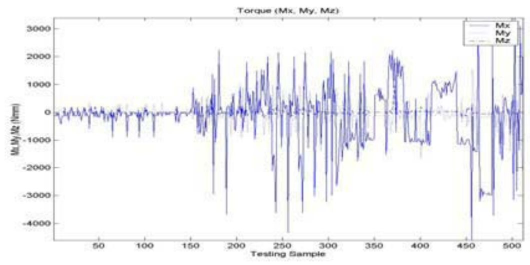


Fig. 3. The training data obtained from the virtual environment

3. Manipulation skill acquisition

Competitive Agglomeration (CA) algorithm is applied to the offline data obtained from the haptic rendered virtual environment to acquire the basic skills applied by the human operator during virtual manipulation. It has been necessary to remove the noise in the data before the analysis. The following algorithm [8] is applied to refine the data, in which u is the action vector:

1. Remove irrelevant actions that do not enhance an action. This is defined by $\|u\| \le \delta_s, \delta_s \ge 0$, where δ_s is an application specific threshold.
2. Remove the operator's rough control. If the difference between two continuous actions is too large, their average value is used.

The algorithm classifies the trajectory into a number of clusters, each representing a state. In the first stage of the process, the constraints on the state variables are learned. These constraints determine the corresponding desired state and choose the best trajectory to the next state. The Competitive Agglomeration classifier, obtained by training the data form the haptic rendered virtual environment, will be used to estimate assembly states in on-line analysis. Actions that determine the best trajectory from current state to next state are computed using knowledge of the system dynamics, acquired through Locally Weighted Regression (LWR) method and encoded as the on-line non-linear function approximator for the trajectories.

Fig. 4 illustrates the skill acquisition procedure. The first two rows represent the offline training procedure. There are 30 groups of data obtained from the haptic virtual environment are used in Competitive Agglomeration classifier training. In this diagram thin lines represent the flow of data in the training progress, the third row represents the online data analysis, thick lines represent the sensory data in the physical assembly procedure from the experimental rig, and dotted lines represent the use of previously trained module.

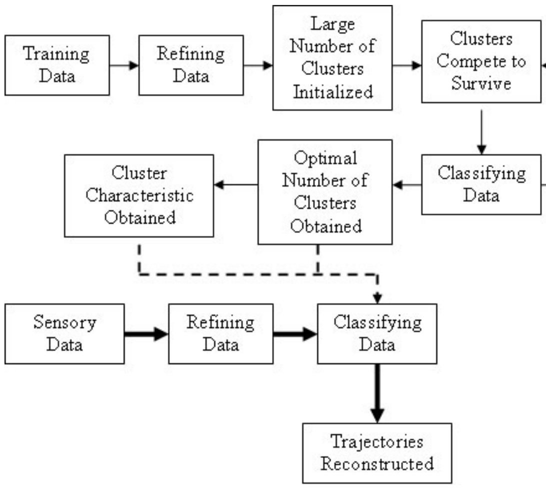

Training Data
Refining Data
Large Number of Clusters Initialized
Clusters Compete to Survive
Classifying Data
Optimal Number of Clusters Obtained
Cluster Characteristic Obtained
Sensory Data
Refining Data
Classifying Data
Trajectories Reconstructed

Fig. 4. Skill acquisition procedure

The Competitive Agglomeration algorithm has several advantages over other clustering algorithm:

(a) The clustering does not suffer from initialization and the local minimum.
(b) The optimal number of clusters is determined after minimization of the fuzzy prototype-based object function.
(c) The points are dynamic and can move from one cluster to another to minimize the fuzzy prototype-based object function.
(d) The algorithm can be used to find clusters of different sizes and shapes by using an appropriate distance measure in the fuzzy prototype-based object function.

The Competitive Agglomeration (CA) algorithm starts by classifying the data set into a large number of clusters. As the training proceeds, clusters compete for survival. Those with large cardinalities survive and clusters that lose the competition are removed. The training process gradually decreases the number of clusters. This will result in an optimal number of clusters when the fuzzy prototype-based object function is minimized.

Frigui and Krishnapuram [1] first introduced Competitive Agglomeration (CA) algorithm as one of the clustering methods mostly for use in image segmentation.

The Competitive Agglomeration (CA) algorithm searches the optimal cluster prototypes in order to minimize the following prototype-based object function:

$$J(Z,U,V)=\sum_{i=1}^{M}\sum_{j=1}^{N}u_{ij}^{2}d^{2}(z_j,v_i)-\alpha\sum_{i=1}^{M}[\sum_{j=1}^{N}u_{ij}]^{2} \qquad (1)$$

$$\text{Subject to } \sum_{i=1}^{M}u_{ij}=1,\ \text{for } j\in\{1,\dots,N\}. \qquad (2)$$

where $Z=\{z_1,\dots,z_N\}$ is a set of N data objects represented by n-dimensional feature vectors. $V=[v_1,\dots,v_M]$ represents M cluster prototypes each of which have to be determined, and v_i is the centre of the cluster. $U=[u_{ij}]$ is the membership degree of the data vector z_j in the ith cluster. $d^2(z_j,v_i)$, representing the distance from feature vector z_j to the cluster center v_i.

Fuzzy Maximum Likelihood Estimation (FMLE) algorithm [2] is used to divide the given data sets into clusters of different sizes and different shapes. FMLE interpret the data set as a p-dimensional normal distribution. The distance of a datum to a cluster is inversely proportional to the posterior possibility (the probability of selecting the ith cluster given the jth feature vector) that a datum z_j is the realization of the ith normal distribution.

$$d^{2}(z_j,v_i)=\frac{(\det(F_i))^{\frac{1}{2}}}{P_i}\exp\left(\frac{(z_j-f_i)F_i^{-1}(z_j-f_i)^{T}}{2}\right) \qquad (3)$$

$$F_i=\frac{1}{N_i}\sum_{j=1}^{N}u_{ij}^{m}(z_j-v_i)(z_j-v_i)^{T} \qquad (4)$$

$$P_i=\frac{1}{N}\sum_{j=1}^{N}u_{ij} \qquad (5)$$

The first part of the prototype-based object-function $J(Z,U,V)$, as the sum of squared distances to the prototypes weighted by constrained memberships, is used to control the shapes and sizes of the clusters and to obtain the compact clusters. The global minimum of the first component is achieved when the number of clusters M is equal to the number of samples N. The second part of prototype-based object function $J(Z,U,V)$, used to control the number of clusters, is the sum of squares of the cardinalities of the clusters. The global minimum of this part is achieved when all points are in one cluster. When both components are combined and α is chosen properly, the final partition will minimize the sum of cluster distances and divide the data set into the smallest possible number of clusters.

The parameter α is chosen by:

$$\alpha(\theta)=\eta_0 e^{\frac{-\theta}{\tau}}\frac{\sum_{i=1}^{M}\sum_{j=1}^{N}u_{ij}^{2}d^{2}(z_j,v_i)}{\sum_{i=1}^{M}[\sum_{j=1}^{N}u_{ij}]^{2}} \qquad (6)$$

Where θ is the iteration number, η_0 is the initial value and τ is a decay factor.

The equation for membership u_{ij} update is given by:

$$u_{ij}=u^{FMLE}+u^{Comp} \qquad (7)$$

$$u^{FMLE}=\frac{1/(2d^{2}(z_j,v_i))}{\sum_{i=1}^{M}1/(2d^{2}(z_j,v_i))} \qquad (8)$$

$$u^{Bias}=\frac{\alpha}{d^{2}(z_j,v_i)}\left(N_i-\frac{\sum_{i=1}^{M}N_i/(2d^{2}(z_j,v_i))}{\sum_{i=1}^{M}1/(2d^{2}(z_j,v_i))}\right) \qquad (9)$$

u^{FMLE} is the membership term in FMLE algorithm which takes into account the relative distance of the feature point to all cluster, u^{Bias} is a signed bias term which depends on the difference between the cardinality of the clusters of interest, and the weighted average of cardinalities from the point of view of feature points.

The value of α is initially set high and then decreased slowly in each iteration to help the Competitive Agglomeration algorithm to seek an appropriate partition with an optimal number of clusters. As the algorithm proceeds, the second part of equation 7 enables the cluster to include as many points as possible. Through the process, a few clusters eventually survive and the rest disappear.

4. Trajectory reconstruction

A common approach used to acquire manipulation skills from the force/torque data generated from the 6DOF haptic rendered virtual environment is known in the literature as behaviour cloning [9]. In this method, the control traces of the operator manipulating the virtual system are used. This approach has been applied to a number of problems such as pole balancing, plane flying and crane operating [10].

The manipulation skills can be learned by any nonlinear function approximator from operators' control traces. In this work, initially, the locally weighted regression (LWR) method is encoded as the approximator. In most learning methods, a single global model is used to fit all the training data, while local models attempt to fit the training data only in a region around the location of the query point. Locally weighted regression is one of the methods, which uses a distance weighted regression to fit nearby points, giving them a high relevance. Locally weighted regression is a form of lazy and memory based learning, since it stores the training data in memory and finds relevant data from the database to answer a particular query point [10]. When a locally weighted linear model is computed, the stored data points are weighted according to the distance from the query point.

The recorded training data are classified into several groups according to contact states between the peg and the hole. Specific indexes are assigned to each group. The index speeds up the search process in the physical manipulation by looking for the actions associated with a particular state in a sub-training data set rather than the whole database.

The following issues are considered when locally weighted regression learning is applied [11]:

(a) Distance function: the relevance between data points is measured using diagonally weighted Euclidean distance.

$$d_m(x,q) = \sqrt{\sum_j (m_j(x_j - q_j))^2}$$
$$= \sqrt{(x-q)^T M^T M(x-q)} = d_E(M_x, M_q)$$

m_j is the feature scaling factor for the jth dimension.
M is a diagonal matrix with $M_{jj}=m_j$.

(b) Separable criterion: the weights of training points are computed using Gaussian kernel, which has infinite extent:

$$K(d) = \exp(-d^2)$$

(c) Enough data: Sufficient data are needed to satisfy the statistical requirements.

(d) Labeled data: Each training data point should be linked to a specific output.

(e) Representations: Fixed length vectors are produced for a list of specified features.

5. Results

The developed algorithm was applied to an experimental rig consisting of a hole with two degrees of freedom (the pitch and yaw angles) controlled by two stepper motors, and a peg with one degree of freedom (the translation along the axis of the peg) controlled by a DC speed motor. These 3DOF are sufficient to study the insertion phase. The radius of the peg is l0mm and that of the hole is 10.05mm. This gives a clearance is 0.05mm between the peg and the hole. The hole has 0.18° degree turning angle in each step.

LWR algorithm was applied to estimate the required corrective actions in the form of rotation angles of the hole according to the contact states. A typical performance of the approach is illustrated in Fig. 5. The variation of 6 normalized series of *fx, fy, fz, Mx, My* and *z* are illustrated in this diagram.

Fig. 5 shows the forces: *fx*, *fy*, *fz*, the torque around *X*, *Y*-axis: *Mx*, *My* and the position translation along *Z*-axis.

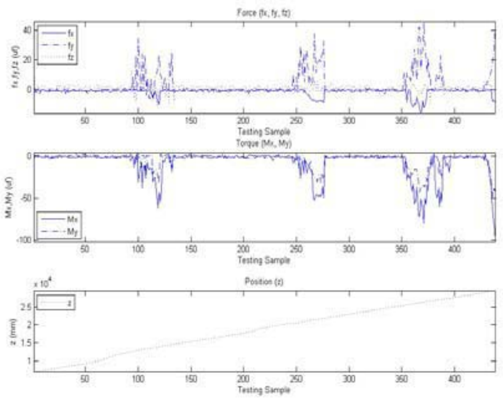


Fig. 5. Experimental results

6. Conclusion

The development of a new paradigm for programming of robotics manipulators performing complex constrained motion tasks was reported in the paper. The manipulation skills are acquired from the demonstration of the task performed by a human operator in a haptic rendered virtual manipulation environment. The peg-in-hole assembly process is used as a platform to study the concept. The work explores how human psychomotor manipulation skills can be replicated by a machine.

The work employed Competitive Agglomeration (CA) algorithm as well as Fuzzy Maximum Likelihood Estimation (FMLE) algorithm to identify the state of the peg and the hole and performing the peg-in-hole insertion process. The concepts and methodologies developed for this application will be expanded in the next stage of the project towards a more generic algorithms and models.

7. References

[1] Frigui, H. and Krishnapuram, R. "Clustering by Competitive Agglomeration," *Pattern Recognition*, vol. 30, no. 7, 1997, pp. 1,223-1,232.

[2] Gath, I. and Geva, A., "Unsupervised Optimal Fuzzy Clustering", *Pattern Analysis and Machine Intelligence (PAMI)*, IEEE Transaction, Vol. 11, 1989, pp. 773-781.

[3] SensAble Technologies, *GHOST SDK Programmer's Guide,* Cambridge MA, 1999

[4] Reachin Technologies AB, *Programmer's Guide Reachin API 3.2,* 2003

[5] Diego C. Ruspini, Krasimir Kolarov, Oussama Khatib, "The Haptic Display of Complex Graphical Environments", *Proceedings of the 24th Annual Conference on Computer Graphics and Tnteractive Techniques,* 1997, pp. 345-352.

[6] Renz, M., Preusche, C., Potke, M. Kriegel, H. P., and Hirzinger, G., "Stable Haptic Interaction with Virtual Environments Using an Adapted Voxmap-PointShell Algorithm", *Proceedings of the Eurohaptics Conference, Birmingham,* UK, 2001, pp. 149-154.

[7] McNeely, W. A., Puterbaugh, K. D., and Troy, J. J., "Six Degree-of-Freedom Haptic Rendering Using Voxel Sampling", *Proceeding of the 26 the Annual Conference on Computer Graphics,* ACM, 1999, pp. 401-408.

[8] Kaiser, M. and Dillman, R., "Building Elementary Robot Skills from Human Demonstration," *Proceedings of the 1996 IEEE International Conference on Robotics and Automation,* vol. 3, 1996, pp. 2700-2705.

[9] Šuc, D. and Bratko, I., "Modelling of Control Skill by Qualitative Constraints", *Thirteenth International Workshop on Qualitative Reasoning,* Loch Awe, Scotland, 1999, pp. 212-220.

[10] Bratko, I., Urbancic, T., and Sammut, C., "Behavioural Cloning: Phenomena, Results and Problems", *5th IFAC Symposium on Automated Systems Based on Human Skill,* Berlin. 1995.

[11] Atkeson, C. G., Moore, A.W., and Schaal, S., "Locally Weighted Learning", *Artificial Intelligence Review,* vol. 11, 1-5, 1997, pp. 11-73.