Trajectory Reconstruction with NURBS Curves for Robot Programming by Demonstration

Jacopo Aleotti, Stefano Caselli, Giuliano Maccherozzi
RIMLab - Robotics and Intelligent Machines Laboratory
Dipartimento di Ingegneria dell’Informazione
University of Parma, Italy
E-mail {aleotti,caselli}@ce.unipr.it giuliano.macchero@libero.it

Abstract—Service robots require simple programming techniques allowing users with little or no technical expertise to integrate new tasks in a robotic platform. A promising solution for automatic acquisition of robot behaviors is the Programming by Demonstration (PbD) paradigm. Its aim is to let robot systems learn new behaviors from a human operator demonstration.

This paper describes a virtual reality based PbD system for pick-and-place and manipulation tasks. The system recovers smooth robot trajectories from single or multiple user demonstrations, thereby overcoming sensor noise and human inconsistency problems. More specifically, we investigate the benefits of the hand trajectory reconstruction with NURBS curves by means of a best-fit data smoothing algorithm. Some experiments involving object transportation while avoiding obstacles in the workspace show the viability and effectiveness of the approach.

Index Terms—Robot Programming by Demonstration, NURBS curves, Virtual Reality.

I. INTRODUCTION

The development of effective personal and home service robots is becoming one of the major goals in the context of human-robot interaction. This challenge requires systems enabling robot programming for users with little or no technical competence. A promising solution that has been proposed for the automatic acquisition of robot behaviors is Programming by Demonstration (PbD) [6]. PbD offers a way to build user-friendly interfaces with natural methods of interaction. For manipulation tasks, a PbD system provides the user with a conceptually simple way to instruct a robotic platform just by showing how to solve a certain task, given the necessary initial knowledge to the system.

PbD systems can be classified into two main categories depending on the way demonstration is carried out. The most general way is performing the demonstration in the real environment [6], [18], [10], [22], [23]. This approach requires object recognition techniques and routines for human gesture segmentation. At the current state of the art, however, this approach can deal only with highly structured environments including a small set of known objects. An alternative approach involves performing the demonstration in a simulated environment [16], [8], [9], [1]. The use of a simulated virtual environment to directly demonstrate the task can help to overcome some difficulties. First, tracking user’s actions within a simulated environment is easier than in a real environment. Second, there is no need for object recognition, since the state of the manipulated objects is known in advance. Third, while performing the demonstration the user can always control the position of virtual cameras through which the operator views the virtual world. Finally, the virtual environment can be augmented with operator aids such as graphical or other synthetic fixtures [15], and force feedback. A major difficulty of this approach is that in cluttered virtual environments, such as for robotic assemblies, it requires that fine manipulation tasks can be tracked with high fidelity.

Learning is one of the main topics of interest in PbD and can be pursued either at the task level or at the trajectory level. The present work focuses on trajectory learning. Previous research [1] has focused on the development of a PbD platform which handles basic assembly tasks where an operator, wearing a dataglove with a 3D tracker, demonstrates the tasks in a virtual environment. The imitation strategy was aimed at recognizing a sequence of hand-object actions such as pick-and-place, stacking, and peg-in-hole, without reproducing the exact movements of the hand. The task recognized was then performed in a simulated environment for validation and, possibly, in a real workspace exploiting basic straight line movements of the end effector. This approach, however, is not feasible in presence of obstacles in the workspace. The present work aims at generating a smooth path that approximates the actual motion of the grasped object. The method allows the system to fit data sets containing one or multiple example trajectories. As stated in [3], data smoothing yields several advantages for characterizing human motion, since it removes noise from tracking sensors and it reduces jitter and unwanted movements.

NURBS (Non-Uniform Rational B-Spline) curves are used for trajectory reconstruction since they have been proven to be the best parametric curves for path planning both for 2D mobile robots [17] and 3D curve approximation [12]. NURBS curves provide a flexible way to represent both standard analytic and free-form curves, their evaluation is fast, and they can be manipulated either by directly modifying the position of the points lying on the curve or by indirectly changing the configuration of the control points. Moreover, NURBS curves can exploit powerful geometric tool kits such as knot insertion and removal.

The rest of the paper is organized as follows. Section 2
reviews the state of the art regarding a sample of related PbD and motion learning research. Section 3 introduces the structure and the main components of the system. Section 4 shows some experimental results that demonstrate the potential of the proposed approach. The paper closes in section 5 summarizing the work and discussing our future investigation.

II. RELATED WORK

Ude and Dillmann [20] proposed a PbD system where a stereo vision system is used to measure the trajectory of the objects being manipulated by the user. Smoothing vector splines are utilized to reconstruct trajectories either in Cartesian or in joint coordinates.

Riley et al. [14] focused on motion synthesis of dance movements for a humanoid robot. The approach includes collection of example human movements, handling of marker occlusions, extraction of motion parameters, and trajectory generation. In [19] the same authors presented a method for the formulation of joint trajectories based on B-spline basis functions.

Lee [7] described a spline smoother for finding a best-fit trajectory from multiple examples of a given physical motion that can simultaneously fit position and velocity information. The system was tested for handwriting motions.

Yang et al. [21] described a system for skill learning with application to telerobotics. Human skills are represented as a parametric model using HMM and the best action sequence can be selected from all previously measured actions. Experiments were carried out involving position trajectory learning in Cartesian space.

Delson and West [3] investigated a method for generating a robot trajectory from multiple human demonstrations for both 2D and 3D tasks. The range of human inconsistency is used to define an obstacle free region. The generated robot trajectory is guaranteed to avoid obstacles and is shorter than any of the demonstrations.


Billard and Schaal [2] proposed a biologically inspired model of human imitation. A dynamic simulation of a humanoid avatar was implemented and the system could learn the principal features of a 3 DOF arm trajectory.

Drumwright and Mataric [5] [4] introduced a method for performing variations on a given demonstrated behavior with the aid of an interpolation mechanism. The method allows recognition of free-space movements for humanoid robots. This approach can achieve a more accurate interpolation of the examples compared to the one proposed in this paper, which is based on a global approximation algorithm. However, our method has fewer constraints, since the approach in [5] [4] requires an appropriate manual segmentation of the example trajectories into time-frames.

The main difference between the previously cited works and our approach lies in the choice of the mathematical tool used for curve fitting. Our solution appears to be the only one that is based on NURBS curves. Moreover our method can be applied for fitting both single and multiple trajectories, as it will be shown in the experiments. The resulting curve can be also easily manipulated in presence of collisions.

III. OVERVIEW OF THE SYSTEM

The PbD system described in this paper handles basic manipulation operations in a 3D virtual environment. The system targets task-level program acquisition in a “block world”. An operator, wearing a dataglove with a 3D tracker, performs a manipulation task in the virtual environment. The virtual scene simulates the actual workspace and displays the relevant assembly components.

The system recognizes, from the user’s actions, a sequence of high level hand-object operations. In details, during the demonstration phase the user can provide multiple examples for each single operation. The movements of the grasped object are tracked; then, the system transforms each group of recorded trajectories, that correspond to the same operation, into a smooth curve, which is finally translated into a set of commands for a robot manipulator. After the demonstration phase the recognized task is performed in a simulated environment for validation. Then, if the operator agrees with the simulation, the task is executed in the real environment referring to actual object location in the workspace. The end effector is programmed to follow the generated curve by applying an inverse kinematics algorithm.

It must be pointed out that the method for trajectory approximation proposed in this paper is demonstrator-dependent and is not guaranteed to generate collision-free trajectories. However, the operator can take advantage of the preventive simulation feature of the PbD environment to avoid unsafe path reconstructions. Moreover, the user interface allows editing of the computed trajectories by manipulating the parameters of the parametric curve, as will be shown in section IV.

Figure 1 shows the main components of the PbD testbed. The actual robot controlled by the PbD application is a 6 dof Puma 560 manipulator. A vision system (currently operating in 2D) is exploited to recognize the objects in the real workspace and detect their initial configurations. The whole application is built on top of a CORBA-based framework which interconnects clients and servers while providing transparent access to the various heterogeneous subsystems.

A. Demonstration interface

A CyberTouch VR glove, from Immersion Corporation Inc., has been used throughout the experiments to track hand motions and gestures. This device is a tactile feedback instrumented glove with 18 resistive sensors for joint-angle measurements and 6 vibrotactile actuators. The glove comprises two bend sensors on each finger, four abduction sensors, plus sensors measuring thumb crossover, palm arch, wrist flexion and wrist abduction. One vibrotactile actuator is located on the back of each finger, and one
additional actuator is located in the palm. Each actuator can be individually programmed to vary the strength of vibrations.

The system comprises also a FasTrack 3D motion tracking device (from Polhemus, Inc.). The FasTrack is an electro-magnetic sensor that computes the position and orientation of a small receiver mounted on the wrist of the CyberTouch (i.e., of an operator performing the task) as it moves through space.

The human operator uses the glove and tracker combination as input devices. For demonstration purposes, operator’s gestures are directly mapped to an anthropomorphic 3D model of the hand which is shown in the simulated workspace along with the objects.

In the demonstration setup, the virtual environment is built upon the Virtual Hand Toolkit (VHT) provided by Immersion Corporation. VHT exploits a Haptic Scene Graph (HSG) to deal with geometrical information in an effective manner. Scene graphs are data structures providing high-level descriptions of environment geometries. Each element of the HSG can be easily imported in VHT through a VRML parser included in the library.

To achieve dynamic interaction between the virtual hand and the objects in the scene, VHT allows objects to be grasped. A collision detection algorithm (V-Clip) generates collision information between the hand and the objects, including the surface normal at the collision point. A grasp state is computed by VHT based on the contact normals; when the condition for a grasped object is no longer satisfied, the object is released.

The system can also incorporate different kinds of virtual fixtures that help the user in grasping and releasing objects, such as vibrotactile, auditory, and visual feedbacks.

B. Trajectory reconstruction

The system analyzes the demonstration provided by the human operator through the task planner module (Figure 1). Segmentation of human actions is performed by means of a simple algorithm based on changes in the grasping state: a new operation is generated whenever a grasped object is released. In general, the specific trajectory followed by the user for a given object motion action should be learned, as it carries useful information about navigation and obstacle avoidance strategies, preferential approach directions, and so on. On the other end, user motions are noisy, often inconsistent, and may include irrelevant deviations and pauses.

NURBS curves are the mathematical tool exploited in this work for trajectory approximation. NURBS have become popular in the CAD/CAM community as standard primitives for high-end 3D modelling applications and simulation environments. They provide the following advantages compared to other parametric curve formulations: their manipulation is easy, their evaluation is fast and computationally stable, and they are invariant under geometric transformations such as translation, rotation and affine transformations.

A NURBS [12] is a vector-valued piecewise rational
polynomial function of the form
\[ C(u) = \frac{\sum_{i=0}^{n} w_i P_i N_{i,p}(u)}{\sum_{i=0}^{n} w_i N_{i,p}(u)} \quad a \leq u \leq b \] (1)
where the \( w_i \) are scalars called the weights, the \( P_i \) are the control points and the \( N_{i,p}(u) \) are the \( p \)th degree B-spline basis functions defined recursively as
\[
N_{i,0}(u) = \begin{cases} 
1 & \text{if } u_i \leq u \leq u_{i+1} \\
0 & \text{otherwise} 
\end{cases}
\]
\[
N_{i,p}(u) = \frac{u - u_i}{u_{i+p} - u_i} N_{i,p-1}(u) + \frac{u_{i+p+1} - u}{u_{i+p+1} - u_{i+1}} N_{i+1,p-1}(u) \] (2)
where \( u_i \) are real numbers called knots that act as breakpoints, forming a knot vector \( U = u_0, \ldots, u_m \) with \( u_i \leq u_{i+1} \) for \( i = 0, \ldots, m \).

The PbD system exploits the NURBS++ library (http://libnurbs.sourceforge.net/) that provides classes, data structures and algorithms to manipulate NURBS curves.

As stated before, the complete task consists of several operations; for every operation the system tracks each demonstration provided by the user, sampling at constant rate the \((x, y, z)\) coordinates of the grasped object. After a number of trials, the resulting data set can be written as \((x_t, y_t, z_t, s)\), where \( t \) is the trial index and \( s \) indicates the index of the sample within the trial. The data points are then transformed into the reference frame of the simulation environment and sorted into a single list of points \( p_i = (x_i, y_i, z_i) \) according to their euclidean distance to the starting point \( p \) of the current movement \((||p_0 - p|| \leq ||p_1 - p|| \leq \cdots \leq ||p_n - p||)\). The starting point is known, given the configuration of the objects in the environment. Finally, approximation algorithms are applied to the sorted data set to find the best-fit NURBS curve. Two approximation algorithms have been investigated [13]: the first is based on a least squares method, while the second is a global approximation algorithm with a specified error bound.

IV. EXPERIMENTS

The capabilities of the PbD system have been evaluated in assembly experiments consisting of pick-and-place operations on a small cubic box (shown in yellow or in clear shading in the following pictures) in a workspace comprising a working plane and two static obstacles (a larger rectangular box and a cylinder). We describe three of the experiments in the following.

The first experiment (Figure 3) consists of a single demonstration. The user is required to grasp the small cubic box and to move it to its final configuration on the working plane, located on the other side of the obstacles, while avoiding the obstacles. In figure 3 the image on the left is a snapshot of the virtual environment for the demonstration phase; the image in the center shows the simulation environment with the approximating NURBS drawn in blue or dark line (the line was rendered using the NURBS interface provided by OpenGL); the center of the grasped object follows the curve. The image on the right shows the real workspace. Figure 4 shows a diagram with the resulting NURBS and the points sampled during the demonstration. The tight fitting of the sampled points highlights the effectiveness of the approximation algorithm.

Fig. 3. Demonstration interface, task simulation and real workspace for experiment 1.

Fig. 4. Generated NURBS and sampled points for experiment 1.
the trajectories has strong "wiggles". Figure 5 shows the resulting NURBS and the points sampled from each trial (drawn with different symbols). The spatial sampling of the curve was kept low resulting in an average of 7 points for each demonstration. These results show the effectiveness of the least squares algorithm with sparse data, since the best-fit trajectory is smooth and correctly removes the jitter of individual demonstrations.

The same experiment has been exploited to assess the robustness of this algorithm with a dense data set, as shown in Figure 7. In this case an average of 26 samples were collected for each demonstration. The resulting NURBS is not satisfactory since it has wobbles, in particular in the central part of the trajectory, where the differences between the four user paths are remarkable.

To deal with dense data set, a modified NURBS-based algorithm has been tested. The algorithm is based on a global approximation method that accepts an error bound as input parameter. Figure 9 shows the results obtained
with the same data set of experiment 2 and an error bound of 5cm. From the above and other similar experiments, we derive that if the error bound is higher than a certain threshold then the generated NURBS is smooth. User motion variability indicates that the robot can actually choose a path within a range of free space. The NURBS approximation with error bound takes advantage of this freedom to compute a smooth trajectory. Therefore, the global approximation algorithm should be preferred when the input data set is dense and there are many points close to each other.

A final experiment consisted of a single demonstration, as in the first one. The user, while performing the task, (deliberately, in this case) moved the grasped object into an obstacle, as shown in figure 6 (left). At the end of the simulation, the user decided to manually modify the resulting trajectory to amend the colliding path segment. The PbD system provides an interactive editor that can be used to directly manipulate the generated NURBS by changing the position of some selected points on the curve. The new curve is redrawn in real time. The final NURBS can be saved and used as input for the execution phase of the task in the real environment.

V. CONCLUSIONS

In this paper we have described a PbD system based on a virtual reality teaching interface. We have investigated the effectiveness of a NURBS-based trajectory reconstruction algorithm that approximates both single and multiple demonstrations. This method allows noise reduction and filtering of unwanted movements. Although the generated paths are not guaranteed to be collision free, the user can take advantage of tools such as a preventive simulation for testing the task and of an interactive editor to manipulate the trajectories. Future work will include the possibility of mixing trajectory reconstruction and trajectory clustering in qualitatively different paths with a geometric approach. A Hidden Markov Model based algorithm will also be investigated to evaluate the consistency of the trajectories among each cluster.

ACKNOWLEDGMENT

This research is partially supported by MIUR (Italian Ministry of Education, University and Research) under project RoboCare (A Multi-Agent System with Intelligent Fixed and Mobile Robotic Components). We thank Monica Reggiani for her contribution to the initial part of this research.

REFERENCES