Visualization and Simulation of Sensory Events as a Representation of States for State-based Teaching by Demonstration in VR

Hiromu Onda, Kosei Kitagaki, and Takashi Suehiro

Intelligent Systems Research Institute
National Institute of Advanced Industrial Science and Technology (AIST)
AIST Tsukuba Central 2, 1-1-1 Umezono, Tsukuba-City, Ibaraki, 305-8568, Japan
onda@ni.aist.go.jp

Abstract - "Teaching by demonstration" is a method to generate a robot program that makes a robot do the same task as the task that a human operator demonstrates. We have developed a "teaching by demonstration in VR" system (TbDinVR) which automatically generates a robot program to work in the real world after a task is demonstrated by an operator in the virtual world. In this paper, we have explained necessary and sufficient condition for states in a task description, advantages of using state-based TbDinVR system, and a simulator for determining parameters of skill primitives by using sensor simulation in VR and a random search technique in detail. We have shown an experimental result and visualization of that result.

Index Terms – Teaching by Demonstration, Visualization, Skill, and VR.

I. INTRODUCTION

Robotics research is one of the most dynamic research areas. Various motions and tasks are being achieved by various robots. The programs for those robots are developed by human programmers, which are developed by trial and error using their experience and knowledge. Parameters in such programs are also determined by the programmers in an ad hoc way or using their own experience and knowledge. These programs pose a “reusability problem.” When other people apply such a program to a very different robot or a very different environment, it may cause difficulty. Since such a program has an unknown part for others than its programmer in order to adjust the program, it is difficult for others to modify the program. In order to deal with this problem, it is important to extract a part of programs, which can be adjusted as parameters, and to determine methods of their adjustment. It is also important to visualize applicability of such programs in order to illustrate limits of the applicability of the programs for programmers.

We have been studying a method of development of robot programs using Teaching by Demonstration in Virtual Reality (TbDinVR) system (Fig. 1). We have developed this system by using contact state transitions, that occur in a fine manipulation task like assembly, as a representation of the task [1][2][3][4]. In this framework, the above problem may occur in development process of skill primitives, which enable our system to make contact state transitions robustly. We develop basic and systematic techniques for development of skill primitives using parameterization from skill primitives based on visualization of sensory data, its analysis, and optimization of the parameters using a random search technique.

In this paper, a simulator for determining a parameter of a skill primitive (the most appropriate reference point) by using sensor simulation in VR and a random search technique. Concretely, we describe a simulator of state transitions using the pseudo contact point method which is used for detection of contact states transitions in fine motion, and an example of determining a parameter of a program for state transition detection systematically using random search technique.

II. SUPPORT FOR DESIGN OF SKILL PRIMITIVES

We mentioned an important problem in the reuse of generated programs by such TbDinVR systems, which is how to generate the programs that can be applied to different environments. There are two aspects to this problem: (1) how the system generates flexible programs that can deal with unknown parts of tasks in unknown and different environments, and (2) how the system generates the (approximately) most proper programs that can deal with different parts of tasks in known and different environments. Approaches that use a learning technique can to some extent deal with the first part of this problem. In fact, some approaches have been proposed where (good) abstraction of examples of tasks is used and adaptation to the different environments that are unknown in advance is achieved by using a learning technique. Though a learning approach is
powerful, it is difficult for us to understand how it works and to evaluate its applicability. From a system designer’s point of view, in order to accumulate and reuse generated programs for fine manipulation, we need to analyze these reasons why these generated programs work and the limitations of applicability of the programs to different environments. In another words, our approach is to try to analyze and solve the second aspect of this problem first. We believe that this analysis makes it possible to solve the first aspect of this problem eventually.

In this paper, we propose a new visualization and simulation system to make skill primitives for such a TbDinVR system. We assume a state-based and “skill based system” [5] (This is explained later in section III). In order to define states that are used in such a task model that abstracts and describes a manipulation task using state transitions which occur in the task we need to be mindful of the following three conditions: (a) condition as top-down type definition, (b) condition as bottom-up type definition, and (c) condition as communicable-type definition of the states. Condition (a) means that necessary and sufficient states for description of the task should be defined. Condition (b) means that states that can be realized by a robot in an environment should be defined. Condition (c) means that objective and detectable states in different environments should be defined. It is important to define states that satisfy these three conditions because these conditions make it possible to make the task description complete, realizable, and communicable (reusable).

Our problem is how to find the necessary states to describe a task and how to find (or make) state-to-state correspondence that can be detectable and realized by a specified robot in at least two specified environments. Sensor simulation and visualization of sensing data are key issues in finding these states. For example, the “Pseudo contact point” method [6] is a powerful tool to detect contact state transition in a manipulation task that deals with motion in contact. This method is used to detect contact state transition using data from a force/torque sensor. An appropriate reference point, that is one of important parameters of this method, should be chosen in order to robustly detect contact state transition in a task. Though this kind of parameter is usually chosen by hand ad hoc or using expert knowledge, in order to deal with this systematically we have developed a simulator to determine the appropriate reference point by using sensor simulation and visualization in VR and a random search technique. We will show the result of experiment and visualization of the result. Visualization of state description in sensor space makes it possible to develop skill primitives that are appropriately abstracted and reusable in different environments.

III. TbDinVR SYSTEM AND RELATED WORKS

Teaching by demonstration (or Programming by demonstration (PbD)) is one of the most promising approaches for making robot programs [7-13].

TbDinVR uses demonstration of a task in VR in the teaching stage [1][2][12][13]. We developed one of

TbDinVR systems using a state-based and skill-based approach for fine motion such as an assembly task [1-2]. Our TbDinVR approach has some advantages in fine manipulation: (1) a real object and a real environment are not needed in the teaching stage, (2) a real sensor is not used in the teaching stage, and (3) reusable data for manipulation in different environments are accumulated in VR itself. The first advantage means that we only prepare a computer and some device to manipulate objects in VR in the teaching stage. The second advantage means that detection of such a contact state is achieved more easily in VR than in the real world. The third advantage means that necessary abstraction of a task for generating reusable programs in different environments is needed to make VR. This abstraction is realized by using only information of manipulated objects except information of manipulators, and this is utilized for generating reusable program from data of demonstration in different VR environments in the teaching stage. Some PbD system that assumes the same real object, manipulator and real environment in both teaching stage and execution stage, don’t have these attributes.

State-based and skill-based systems need to be defined here. Execution of a task by a state-based and skill-based system needs a sequence of states (for example, contact states) of the objects from the initial state to the target state and manipulation skill primitives of the manipulator system to achieve these desired states. These states and primitives correspond to the tasks to be performed by a manipulator and how the manipulator achieves them, respectively. Two different types of information about the execution of fine motion are often mixed and mixed data have been used by some previous researchers of PbD systems. In these systems there are both the (internal) events of the manipulator and the events of the objects in the teaching data. Our system explicitly separates the data because the second event mainly depends on the relation of the (manipulated) objects and the first event mainly depends on a specific manipulator and its skill primitive library (if it has one).

There are also tele-operational works [14][15] in which we concentrate simulation of a task before execution of it and the state transition graphs are given in advance. Our system automatically extracts contact state transition sequences of an object from demonstrations in a simulator and uses them for monitoring a task and generating a program for the task.
The process of teaching of our previous TbDinVR system is as follows: (1) human operator demonstrates a fine manipulation task in VR in the teaching stage, (2) contact state transition is automatically detected by the system, (3) a sequence of skill primitives that realize each detected contact state transition in (1) is generated by the system, and (4) the system achieves the task in the execution stage (Fig. 2).

The state-based and skill-based system that we assume and developed for the execution system in our TbDinVR system needs a skill primitive library. At the present time, we have developed a skill primitive for contact state transition by hand and using a trial and error approach. There are some challenging works [16-19] that attempt to generate this kind of low-level skill itself by using a PbD technique. Though PbD is a powerful and promising approach, we don’t think programs generated by PbD can be used directly as skill primitives in our TbDinVR. These generated programs are needed to be tuned up by hand in order to reuse them in environments different from the environment in the teaching stage. We have developed a support system for development of a state-based TbDinVR system. This support system has the capability of simulation of state transition in sensor space. It enables us to simulate and visualize sensory events in a task for development of skill primitives for state-based TbDinVR. The simulator is implemented in EusLisp [20], which is an integrated programming system for robotics based on Common Lisp and object-oriented programming. Euslisp has a solid modeling system and Xwindows interface facilities.

In the next section we explain simulation motion in contact. Simulation of motion in contact needs simulate states transition in sensor space.

V. STATE TRANSITION IN SENSOR SPACE

The three conditions stated in Section II for definition of states in TbDinVR demand that state transitions in the teaching stage correspond to state transitions in sensor space in the execution stage of state-based TbDinVR. In this section, simulation tools for sensory events and simulation of state transition in sensor space are explained. As an example, we explain the simulation tool of the “pseudo contact point” method. First, we explain the internal representation of pseudo contact point, simulation of movement in a vertex-face contact transition, and experiment in the real world.

A. Pseudo Contact Point Method

The “Pseudo contact point” method is a powerful tool to monitor contact state transition in a task. A pseudo contact point is defined as the point obtained from a set of observed force/momentum data existing on the line L containing the contact point and external force vector and closest to a given reference point (Fig. 3). We assume a point contact state where the end-effector and an external environment contact each other at a point. In order to simplify the equations, the gravity term has been compensated for. The points, Os, Rr and Re indicate the center of the Force/Torque sensor, the reference point and the contact point.

Tensor analysis [21] is used in this paper in order to identify invariant for coordinate transformations and in order to identify the way of transformation of an object according to coordinate transformations (i.e. scalar, contravariant vector, covariant vector, pseudo-vector, density and so on). We choose this representation because there is the advantage that we can find invariants and appropriate coordinates from which we can robustly identify an event that we give an attention to in sensor space. In addition to this advantage, a human (skill primitive programmer) can estimate candidates of objects that should be given an attention to in sensor space.

Tensor representation of the “pseudo contact point” is shown here. The equation for calculation of “pseudo contact point” in [6] is as follows (Equation 4.1 is a line on which a contact point exists (line L in Fig. 3)),

\[
  r_c = k_0 \frac{f_c}{\|f_c\|} + \frac{f_c \otimes m_s}{\|f_c\|},
\]

(Eq. 4.1)

where \( r_c \) denotes the position vector of a contact point, \( f_c \) denotes the vector of external force, \( m_s \) denotes a
momentum, $k_0$ is a scalar. This equation of a line $L$ can be represented by using tensor analysis as follows,
\[
\tilde{\mathbf{M}}_g = \tilde{e}_{\mathbf{lm}} f_{\mathbf{g}}^l (g_{\mathbf{lm}} f_{\mathbf{g}}^l)\]
\[
f^l_g = t(g_{\mathbf{lm}} f_{\mathbf{g}}^l) + \frac{1}{g_{\mathbf{lm}} f_{\mathbf{g}}^l} \tilde{E}_{\mathbf{lm}} f_{\mathbf{g}}^l \tilde{\mathbf{M}}_g.
\]

where $r$ is a contravariant vector that means “position,” $f$ is a covariant vector that means “force,” $t$ is scalar, $M$-tilde means “momentum,” $g$ is a contravariant fundamental tensor (metric tensor), $E$-tilde is a pseudo-tensor density of weight -1 and contravariant valence 3, and $e$-tilde is a pseudo-tensor density of weight -1 covariant valence 3. Then the “pseudo contact point” is represented by using tensor analysis as follows,
\[
f^l_g = \frac{r_{\mathbf{g}} f_{\mathbf{g}}^l (g_{\mathbf{lm}} f_{\mathbf{g}}^l)}{g_{\mathbf{lm}} f_{\mathbf{g}}^l} + \frac{1}{g_{\mathbf{lm}} f_{\mathbf{g}}^l} \tilde{E}_{\mathbf{lm}} f_{\mathbf{g}}^l \tilde{\mathbf{M}}_g.
\]

For example, if the position of the F/T sensor is chosen as a reference point (that is, every component of the right side of $r$ is set to zero), the “pseudo contact point” is represented by the following equation,
\[
f^l_g = \frac{1}{g_{\mathbf{lm}} f_{\mathbf{g}}^l} \tilde{E}_{\mathbf{lm}} f_{\mathbf{g}}^l \tilde{\mathbf{M}}_g.
\]

B. Simulation and Experiment

In this section, we illustrate the results of visualization and simulation. An example of a task is when a manipulated object in contact state transition goes from one vertex-face contact to another vertex-face contact (Fig. 4). Figure 5 shows the data of force and momentum from the F/T sensor in this task. The contact state transition from a vertex-face-contact to another vertex-face contact occurs around at 40. Figure 6 shows one result of visualization of contact states transition in sensor space that is accomplished by our support system of development of skill primitives for ThDinVR. (Though it seems difficult to understand the shape of a locus of a pseudo contact point from these three static images, it is easy to understand the shape and structure by using animation in which our system shows rotation of this locus.) Two clusters, which corresponds to different vertex-face contact states, are observed in this figure. Force and momentum are measured by a F/T sensor that is set to the wrist of a PA10 manipulator. Displacement and force coordinates are described with respect to this local coordinate frame that is set to the wrist position. In order to obtain an appropriate reference point for “pseudo contact point” monitoring, this system searches the reference point that minimizes the distance between the contact points and “pseudo contact” points (ref. h(p) in Equation 5.1 in this section). Search space is a cube with diagonal that is the line segment between (-200, -200, -200) and (200, 200, 200). The system searches for a reference point $P_{\text{min}}$ (151, 36, -68) that gives the (probabilistically approximated) minimum ($h(P_{\text{min}})=732$) by generating a hundred thousand points randomly in this search space (Monte Carlo algorithm [22]). Pseudo-random numbers of this random search used in this paper are generated by erand48 ([23]). We checked uniformness and randomness of pseudo-random numbers by checking points of scatter plots, k-th moment of uniform distribution, and correlations between neighbors of pseudo-random numbers. The data from the F/T sensor in this task is used to produce the loci of pseudo contact points $X(p)$ in this search. The distance $f(x; p)$ is calculated by using the following function,
Fig. 7 illustrates a locus that is found in a task. You can use another criterion. For example, to determine whether contact state transition occurs or not, you don't need to estimate a position of contact point and you need to detect whether contact state transition occurs or not, you can use another criterion. For example, the middle of Fig. 7 illustrates a locus that is found in a search process. It is sufficient for this detection, though these centers of points are far from real contact points in a task.

A method of integrating multiple sensors and actuators using online estimation of the Jacobian matrix [24] (Fig. 8) is used to implement sensory feedback control of the state-based system. We developed the sub-system and simulator for free motion, states of which is represented by artificial constraints in free space by using this method. For example, a set of features on a manipulated object in real images from a visual sensor is transferred to a goal that is also represented by a set of features on an environment in real images. We apply this method in order to integrate our subsystem for free motion and that for motion in contact in next step of this research.

VI. CONCLUSION AND FUTURE WORKS

In this paper, we have described the necessary and sufficient condition for states in a task description, the advantages of using a state-based TbDinVR system, and a simulator for determining a parameter of a skill primitive (the most appropriate reference point) by using sensor simulation in VR and a random search technique. We have shown an experimental result and visualization of that result. Our method gives a better parameter (the probabilistically best parameter) than the parameter that are determined by a human (skill primitive programmer). Visualization of state description in sensor spaces makes it possible to develop skill primitives for a state-based TbDinVR system.

In this paper, we have found a reference point in order to minimize the distance between a contact point and a pseudo contact point. But this is only one of criteria. If you don't need to estimate a position of contact point and you need to detect whether contact state transition occurs or not, you can use another criterion. For example, the middle of Fig. 7 illustrates a locus that is found in a search process. It is sufficient for this detection, though these centers of points are far from real contact points in a task.

A method of integrating multiple sensors and actuators using online estimation of the Jacobian matrix [24] (Fig. 8) is used to implement sensory feedback control of the state-based system. We developed the sub-system and simulator for free motion, states of which is represented by artificial constraints in free space by using this method. For example, a set of features on a manipulated object in real images from a visual sensor is transferred to a goal that is also represented by a set of features on an environment in real images. We apply this method in order to integrate our sub-system for free motion and that for motion in contact in next step of this research.

ACKNOWLEDGMENT

A part of this study was financially supported by the Budget for Nuclear Research of the Ministry of Education, Culture, Sports, Science and Technology, based on the screening and counselling by the Atomic Energy Commission.

REFERENCES


