J Intell Robot Syst (2011) 62:467–500
DOI 10.1007/s10846-010-9465-0

Planning-Space Shift Motion Generation: Variable-space Motion Planning Toward Flexible Extension of Body Schema

Yuichi Kobayashi · Shigeyuki Hosoe

Received: 22 October 2009 / Accepted: 26 August 2010 / Published online: 15 September 2010

Abstract To improve the flexibility of robotic learning, it is important to realize an ability to generate a hierarchical structure. This paper proposes a learning framework which can dynamically change the planning space depending on the structure of tasks. Synchronous motion information is utilized to generate 'modes' and hierarchical structure of the controller is constructed based on the modes. This enables efficient planning and control in low-dimensional planning space, though the dimension of the total state space is in general very high. Three types of object manipulation tasks are tested as applications, where an object is found and used as a tool (or as a part of the body) to extend the ability of the robot. The proposed framework is expected to be a basic learning model to account for body schema acquisition including tool affordances.

Keywords Hierarchy generation · Motion planning · Tool affordance

1 Introduction

Among approaches for improving learning ability of robots, hierarchical learning is an important issue. In some researches of hierarchical learning, navigation in large building like complex mazes [6, 11] or controlling a robotic arm with multiple joints [20] are investigated as examples of large and complex problems. Apart from those high-dimensionalities, one aspect of the complexity in robot tasks is that it involves *objects* whose shapes can vary depending on situations and the objects can

Y. Kobayashi (✉)
Tokyo University of Agriculture and Technology, 2-24-16 Naka-cho,
Koganei, Tokyo, Japan
e-mail: yu-koba@cc.tuat.ac.jp

S. Hosoe
RIKEN-TRI Collaboration Center for Human-Interactive Robot Research (RTC),
Nagoya University, Shimoshidami,
Anagahora, Nagoya, Japan

play various roles; a target to be carried, an obstacle to avoid collision, or a tool which can be used to achieve different objectives. The existence of such objects makes tasks diverse (various objectives and situations), complex (different dynamics depending on contact or non-contact) and high-dimensional (configuration space). But the tasks that involve objects have not been explicitly discussed in the researches of hierarchical learning.

As an approach to the complex control problems, it is known that some behaviors of the robotic system can be realized by combination of multiple modules, each of which has relatively small dimension, even though the DOF of the total system is very large. For example, subsumption architecture [5] realized flexible and adaptive behaviors of locomotion robots with many DOFs, while individual modules in the architecture played rather simple roles such as collision avoidance or simple maneuver. From the viewpoint of building learning robots, such architectures realize flexibility by focusing on just a part of the total system, where the part in focus flexibly varies depending on the situations and objectives.

In this paper, a learning architecture that can account for such *variable focus* in robotic motion learning *with objects* is proposed. A 'part in focus' is interpreted as a space for motion planning in this research. By changing or 'shifting' the planning space, the architecture can be applied to diversity of tasks. In the proposed architecture, variables to be controlled can be variant, differing from the multiple-module learning model [30] where the state variables and control variables have to be invariant.

In the researches on hierarchical motion planning and generation introduced above [6, 11, 20], 'abstraction' of state space is a common idea, which means that planning is first done in an abstract state space, and later the total motion generation is considered with the total state space. In this paper, we consider a chain of objects and motion of tip object of the chain. The proposed planning framework also uses a kind of abstraction, and thus can be regarded as hierarchical, in the sense that first trajectory of the tip object is calculated and later planning problem (generating trajectory) is attributed to planning of shorter chains successively, where considering the tip of current chain for planning can be regarded as an abstraction.

One possible application of planning-space shift learning is a problem of adaptive tool-use. An object can be utilized to extend reachable region of a robot when the robot can move it. That is, the robot can use the object as a tool. The ability of tool-use has gathered attention partly because it is deeply related to the issue of affordance [10] and body schema, that are frequently discussed in the field of developmental robotics [1, 15], as well as neuropsychology and cognitive science [17]. Stoytchev proposed a behavior-based approach to realize representation of tool use [27]. They proposed to let the robot find the relation between actions while the robot is grasping a tool and motions of an object. The tool affordances were expressed by distance and direction of object motions for several direction of tool motions and various shapes of tools. Based on those experiences, the robot realized carrying the object to desired goals. Nabeshima et al. also proposed a learning framework for tool-use [21]. They used synchronousness of visual and tactile information to extend the body schema. A neural network architecture was implemented for integrating tactile information and visual information related to a tool and an object by an associative memory. By learning extended body schema, the robot realized reaching motion to the object by grasping the tool.

This paper proposes a general and bottom-up description of hierarchy generation. In [27], actions of the robot were restricted to discrete ones, such as 'slide arm left'. This paper deals with continuous observation and action variables, which enables more varied planning of motions. In [21], visual information that is useful for motion generation (such as direction from the tip of the tool to the object and distance between the hand and the object) is extracted by the designer. In this paper, no specific feature extraction is introduced except observation of configuration variables of the robot body and the objects. Moreover, in both [27] and [21], distinction of 'tool' and 'object' is given in advance. On the other hand, distinction of object and tool is not given in this paper. The robot only observes information on objects without any prior knowledge and it learns functional relationship among object motions. Since there is no pre-defined distinction of object and tool, an object can play a role of object to be moved toward a destination or a tool to move another object in this paper's framework. A disadvantage for this more bottom-up framework is that acquisition of relations among objects requires large amount of trial motions and increase of computational amount is also inevitable.

Another related research was conducted by Drescher [9]. He proposed a Piagetian developmental approach for robot learning, where concept invention and planning of goal-oriented actions were designed based on schema mechanism. With a simulated body in a microworld, an agent gradually obtained persistent external objects. In comparison with this, our paper deals with more simple and fundamental problem from the perspective of kinematics learning, while Drescher's schema is based on tactile sensation, moving the view range, distinction of peripheral and foveal vision. While the basic idea of finding relation between sensing and action is close to [9], this paper proposes to find structure of problem from less information, that is, only with motion synchronousness and distinction by change of observation variables. Besides, this paper deals with continuous observation variables, which is much closer to the real world and brings much richer and finer structure of object behaviors.

In the research domain of manipulation planning, motion generation of robots with objects has been discussed. For example, Siméon et al. discussed application of probabilistic roadmap method [14] to planning of manipulation [26]. Compared with those planning methods, the planning strategy presented in this paper is rather primitive and difficult to be applied to complex cases. On the other hand, these conventional researches never deal with 'finding object' or 'finding to use an object as a tool', which can be discussed only by a learning framework from much lower level, as discussed in this paper.

From the perspective of combination of continuous control and mode switching, the framework of hybrid dynamical system control [25] is closely related to the problem discussed in this paper. Though several frameworks have been proposed for obtaining general solutions for this problem (e.g., [3]), unknown factors of control problems have been rarely considered. The planning strategy proposed in this paper can be regarded as a learning-based approach to a class of the hybrid system control problems, where system dynamics including discrete modes and conditions for mode switchings are unknown to the robot.

The remainder of the paper is constructed as followings. In Section 2, problem settings for the proposed learning architecture is described. The learning architecture is proposed in Section 3, followed by evaluations by simulation in Section 4.

2 Problem Description

For simplicity, it is assumed that all motions of the robot and the objects are quasi-static.[1] Let N be the number of joints of the robot arm and M the number of the objects, which move only through contacting with the robot hand or other objects, i.e., they do not move by themselves.[2] The state variables of the system are the followings:

- Variables which express the configuration of the robot; $\boldsymbol{\theta} \in \Theta \subset \mathbb{R}^N$
- Variables which express the configuration of the objects; $\boldsymbol{q}_1 \in \mathbb{R}^{N_1}, \cdots, \boldsymbol{q}_M \in \mathbb{R}^{N_M}$

The objective of the robot task is to move an object to a certain desired configuration. Which object should be moved and its desired configuration are given on-line. The agent can observe the following variables in addition to the configuration of the robot $\boldsymbol{\theta}$:

- Position of the robot hand (end effector) $\boldsymbol{h}(\boldsymbol{\theta})$
- Configuration variables of the objects measured from image information

These variables are presented to the agent as the observation vector $\boldsymbol{y} \in \mathcal{Y} \subset \mathbb{R}^{L_o}$. Initially, the number of the objects or DOFs of motions of them are unknown. The i-th component of observation variable $\boldsymbol{y}$ is denoted by $y_i, i = 1, \cdots, L_o$. The agent does not know the correspondence between the elements of $\boldsymbol{y}$ and the observed position of the robot hand, nor the functional relationships among the elements of $\boldsymbol{y}$ (including the kinematics of the arm).

An example of a robot task is depicted in Fig. 1. The configuration of the robot is expressed by its two joint angles, $\boldsymbol{\theta} = [\theta_1, \theta_2]^T$. Configuration variables of two objects are given by $\boldsymbol{q}_1 = [q_1, q_2, q_3]^T$ for the square object and $\boldsymbol{q}_2 = [q_4, q_5]^T$ for the circular object, where posture is omitted for the circular object. The position of the robot hand is denoted by $[y_1, y_2]^T$, which corresponds to $\boldsymbol{h}(\theta_1, \theta_2)$. The other observation variables are given by $[y_3, y_4, y_5]^T = \boldsymbol{q}_1$ and $[y_6, y_7]^T = \boldsymbol{q}_2$, respectively.

At each time step, the robot can change its configuration variables. That is, $\boldsymbol{u} = \Delta\boldsymbol{\theta}$ is the control input to the system. The following scenario is assumed about the task:

- At the initial state, the robot hand and all of the objects are still and separately located.[3]
- The robot hand begins moving when input $\boldsymbol{u}$ is given.
- A still object begins moving when it comes in contact with the robot hand or other objects in motion. It is assumed that the coupled system of the robot hand and the objects makes a chain form structure by supposing that contact between objects happens only at the endmost moving object in the chain.
- The movement of an object contained in the chain is assumed to be affected only by the one of the neighboring object that is positioned to the robot hand side.

[1] By this, velocity components can be omitted from state variables.

[2] It is assumed that the robot contacts with an object only at its hand.

[3] This assumption is needed so that the robot can distinguish objects only by the information of motion synchronousness.

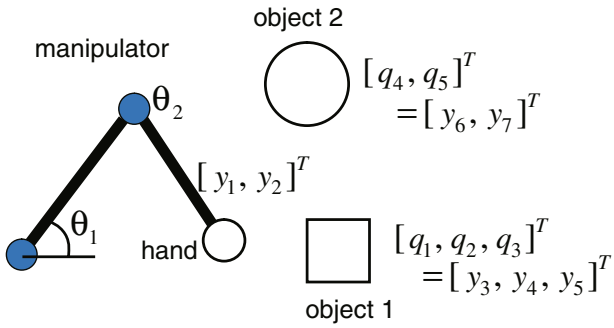


Fig. 1 An exemplar problem setting with two objects

Therefore the effect of the object motions is transmitted from the robot hand to the endmost object in unidirectional way.

The third assumption of forming a chain might restrict the generality of the proposed framework, because generally there can be various types of contacts, such as loops and trees. From the perspective of tool use, however, we believe that the case that the body and tools form a chain will be very common.

3 Learning Structure

The learning structure proposed in this paper is constituted by the four basic functional components. An overview will first be presented in the following subsection and following to this the detail of the algorithm will be described.

3.1 Basic Idea of Planning-Space Shift

The agent repeats trial motions while resetting the configurations of the robot arm and the objects (the reset is done after a certain numbers of motions). The agent initially yields random action controls (random joint velocities), observes $\boldsymbol{y}$ and collects them together with control input $\boldsymbol{u}$. An example with $L_o = 5$ is shown in Figs. 2 and 3. Figure 2 shows that by detecting synchronous movements in the observed variables of $\boldsymbol{y}$ initially, variables y_3, y_4 and y_5 are observed to change in accordance with the change of $\boldsymbol{\theta}$. Also it indicates that y_3, y_4, y_5 take values in set $\mathcal{Y}^1 \subset \mathbb{R}^3$. This corresponds to a situation where the end effector of the robot can be expressed by three dimensional vector with two dimensional vector for the position and a scalar for the posture for planar motions. In this situation, only robot hand moves while all objects are remaining still. And Fig. 3 shows that after the movement of Fig. 2 variables y_1 and y_2 are also observed moving (, where $\mathcal{Y}^2 \subset \mathbb{R}^2$ denotes the value set taken by y_1 and y_2). Suppose there is a circular object on the plane whose configuration can be expressed by two dimensional vector, $[y_1, y_2]$. This case corresponds, from the scenario stated above, to the situation where the robot hand contacted with the circular object and both are moving by keeping the contact. In the following, to designate the set of varying observation variables including the occurrence relationships we will use the terminology ‘mode’. For instance, the case

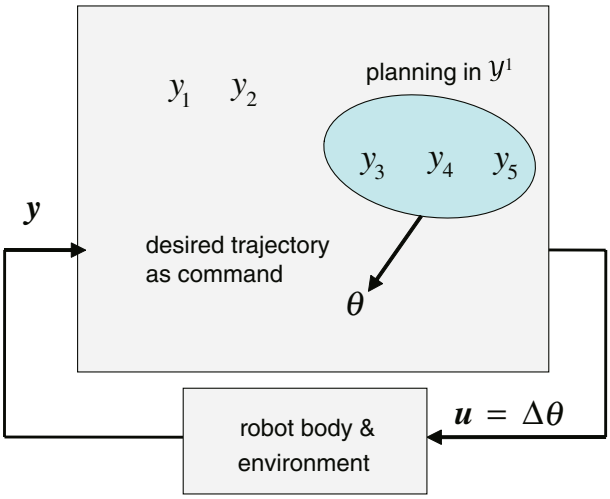


Fig. 2 Mode 1: generation of command $\boldsymbol{u}$ from $\mathcal{Y}^1$

where only variables y_1, y_2, y_3 change while y_4, y_5 are still is called mode 1 and the case where y_4, y_5 also change following y_1, y_2, y_3 is called mode 2. If y_1, y_2, y_3 begin changing first and later y_4, y_5 follow, then a different mode number will be assigned.

By collecting the data of $\boldsymbol{\theta}$ and y_3, y_4, y_5 for some time period in the situation of Fig. 2, one can obtain a functional relationship between $\boldsymbol{\theta}$ and y_3, y_4, y_5 represented as a mapping from Θ to $\mathcal{Y}^1$ and the inverse mapping from $\mathcal{Y}^1$ to Θ (expressed by an arrow in the figure). Similarly to the case of mode 1, the mapping from $\mathcal{Y}^1$ to $\mathcal{Y}^2$ is regarded as a forward mapping and the inverse mapping from $\mathcal{Y}^2$ to $\mathcal{Y}^1$ is acquired. Note that it requires fair amount of trial motions to explore mode 2 with simple random motions, because the system easily goes back to mode 1. One idea to reduce

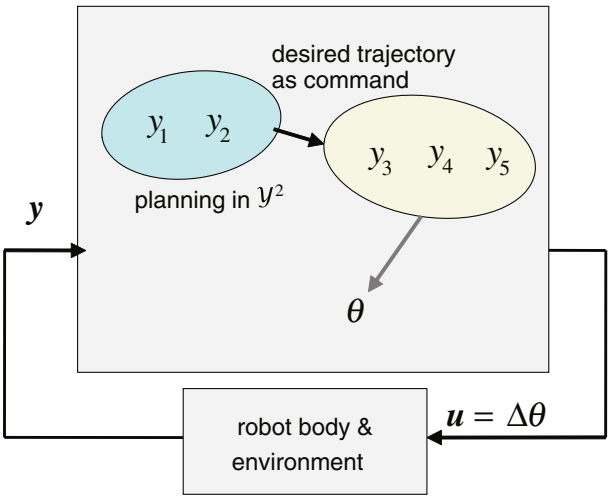


Fig. 3 Mode 2: generation of trajectory in $\mathcal{Y}^1$ from one in $\mathcal{Y}^2$

the amount is to improve controller to avoid mode transitions while the exploration, which is described later in Section 3.4.

The constructed mappings can be used when a task is given to the robot system. For instance, let a desired configuration for $\{y_1, y_2\}$ is given as a cross depicted in Fig. 4. First, a total trajectory with mode transitions is planned as shown as a succession of arrows in the right part of the figure. Up to point p_{12} indicated in Fig. 4, the trajectory has to be retained in mode 1, i.e., only variables y_3, y_4, y_5 can change but y_1, y_2 must be constant. For this, we can use the constructed mapping from $\mathcal{Y}^1$ to $\mathcal{U}$. After transition to mode 2, the mapping from $\mathcal{Y}^2$ to $\mathcal{Y}^1$ is used while maintaining mode 2. Note that in the case of mode 2, the trajectory is first planned in space $\mathcal{Y}^2$. Then a trajectory in $\mathcal{Y}^1$ that realizes the trajectory in $\mathcal{Y}^2$ is generated. In other words, the planning space 'shifts' to $\mathcal{Y}^1$. This is the basic idea of 'shifting' the planning space.

From the perspective of hierarchical structure, control variables in $\mathcal{U}$ 'obeys' the trajectory defined by observation variables in $\mathcal{Y}^1$ in mode 1 (Fig. 2). Next in mode 2, the trajectory in $\mathcal{Y}^1$ is dominated by the trajectory defined in $\mathcal{Y}^2$ (Fig. 3). Thus, different relations of dependence among groups of variables can be seen in each mode, where one group of variables in the upper layer dominates the behavior of another group of variables in the lower layer. In this sense, the proposed architecture can be regarded to generate hierarchy autonomously.

This architecture brings two advantages:

- The motion-planning space is not pre-defined. The planning framework can be flexibly applied to various tasks with different objects and different number of objects.
- The exploration space is divided into lower dimensional spaces ($\mathcal{Y}^1$ and $\mathcal{Y}^2$ in the present example) and this makes the exploration more efficient.

The proposed learning structure has the following functional components:

1. Mode generation by motion synchronousness:
   Modes are generated by grouping observation variables based on motion synchronousness. Observation variables are stored for estimating the boundary between modes.
2. Acquisition of motion mappings with mode keeping and mode transition controls:
   Mapping representing a functional relationship between the observed variables corresponding to objects moving in contact is estimated. This mapping is used to

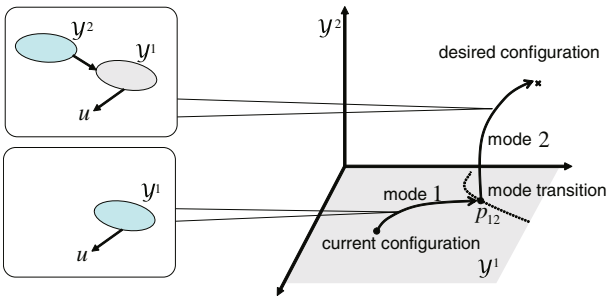


Fig. 4 Planning-space shift among different modes

plan motion of the 'causal' object when a desired motion of the 'resultant' object is given. Within each mode, there can be two strategies for control;

(a) to keep the same mode without any occurrence of mode transition, and
(b) to make mode transition to a certain target mode.

3. Exploration and estimation of reachable region:
Using the motion mappings mentioned above, each mode is explored. During exploration, observation variables are stored to estimate reachable regions.
4. Planning and control via multiple modes:
Once a desired configuration of a certain object is given to the robot, the total trajectory is planned by finding via points on the boundaries among modes (p_{12} in Fig. 4). Parameterization of boundaries is used for this.

These processes can be conducted simultaneously, but we execute and evaluate each process one by one, for simplicity. The following subsections give the details of the components.

3.2 Mode Generation Using Motion Synchronousness

Recall that vector $\boldsymbol{y} = [y_1, \cdots, y_{L_o}]^T \in \mathcal{Y} \subset \mathbb{R}^{L_o}$ is the observation vector. $\boldsymbol{y}(t)$ denotes the observed value of $\boldsymbol{y}$ at time t. In order to detect synchronousness among observation variables, time differences of observation variables are defined as

$$\Delta y_a(t) = y_a(t) - y_a(t-1), \quad a = 1, \cdots, L_o. \tag{3.1}$$

With $\Delta y_a(t)$, its indicator set I_C is defined as[4]

$$I_C(t) = \{a | \Delta y_a(t) \neq 0\}. \tag{3.2}$$

Based on this, the notion of 'mode' is introduced as follows. 'Mode 1' specifies the initial stage of each trial where only the robot hand moves and all the objects are still. This is characterized by the set value taken by $I_C(t)$, which is constant up to the moment when the robot hand touches an object (let it be t_1). Let us represent this set as

$$I_C^{(1)} = \{i_1^1, i_2^1, \cdots, i_{n_1}^1\} = I_C(t),\ 0 \leq t < t_1. \tag{3.3}$$

The corresponding observed variables (changing components of $\boldsymbol{y}$ describing the movement of the robot hand) are represented by the vector

$$\boldsymbol{y}^{(1)} = [y_{i_1^1}, \cdots, y_{i_{n_1}^1}]^T,\ \{i_1^1, \cdots, i_{n_1}^1\} = I_C^{(1)}. \tag{3.4}$$

Continuing the trial, we come to a new stage denoted as mode 2, which represents that the robot hand contacts with an object and both the robot hand and the object move together. This new mode is characterized by the indicator set

$$I_C^{(2)} = \{i_1^1, i_2^1, \cdots, i_{n_1}^1, i_1^2, i_2^2, \cdots, i_{n_2}^2\} = I_C(t),\ t_1 \leq t < t_2, \tag{3.5}$$

[4]This condition of nonzero change is not applicable to problems with noise. An approach to this problem is to set a range of values to judge whether they are still. This point is discussed later in Section 5.

where t_2 denotes the time moment the object in mode 1 meets another object and n_2 the dimension of the vector composed of the observed variables

$$\boldsymbol{y}^{(2)} = [y_{i_1^2}, \cdots, y_{i_{n_2}^2}]^T, \ \{i_1^2, \cdots, i_{n_2}^2\} = I_C^{(2)} - I_C^{(1)}, \tag{3.6}$$

corresponding to the new object. Continuing further in this way, at every moment when a mode that has never been encountered is experienced, we define a new mode by assigning it a new numbering, its corresponding indicator set and the vector expressing the new variation components of $\boldsymbol{y}$.[5] The index of the tip object of the chain corresponding to mode i is denoted by $T(i)$ as a function of i. Furthermore let us denote the vector space spanned by observed vector $\boldsymbol{y}^{(k)}$ for object O^k[6] as $\mathcal{Y}^k$. Using these notations, the set of all the observed variables for O^k throughout trials is defined as

$$\mathcal{O}_{(k)} = \{\boldsymbol{y}^{(k)}(t) \in \mathcal{Y}^k\}. \tag{3.7}$$

Note that the increase of the total number of objects causes (potentially) exponential growth of modes. In the application, however, we do not suppose modes with large number of objects, because standard applications for tool use do not require large number of objects.

3.3 Estimation of Mode Boundaries Using SVM

The boundaries among different modes can be estimated by collecting observed values on $\mathcal{Y}$ at the moment of the mode changes. Specifically, let us consider the boundary between mode i and j, where it is assumed that $I_C^{(i)} \subset I_C^{(j)}$ and let $k = T(i)$ and $\ell = T(j)$. We collect all the observed data $\boldsymbol{y}_1(t)$, $\boldsymbol{y}_2(t)$ just before and $\boldsymbol{y}_1(t+1)$, $\boldsymbol{y}_2(t+1)$ just after the contact between objects O^k and O^ℓ, respectively. Let us arrange all such data in the following way.

$$\begin{aligned}
B_{i,j}^0 &= \left\{\left[\boldsymbol{y}_1(t)^T, \boldsymbol{y}_2(t)^T\right]^T, |\, \boldsymbol{y}_1(t), \in \mathcal{O}_{(k)}, \boldsymbol{y}_2(t), \in \mathcal{O}_{(\ell)}\right\}, \\
B_{i,j}^1 &= \left\{\left[\boldsymbol{y}_1(t+1)^T, \boldsymbol{y}_2(t+1)^T\right]^T | \, \boldsymbol{y}_1(t+1) \in \mathcal{O}_{(k)}, \boldsymbol{y}_2(t+1) \in \mathcal{O}_{(\ell)}\right\}, \\
B_{i,j} &= B_{i,j}^0 \cup B_{i,j}^1 = \{\boldsymbol{a}_1, \boldsymbol{a}_2, \cdots, \boldsymbol{a}_{m_s}\},
\end{aligned} \tag{3.8}$$

where m_s denotes the number of the observed data. To get the boundary equations between modes i and j, non-linear Support Vector Machine (SVM), which is known as a non-linear classifier with kernel functions [29], is applied.[7] For this, let us denote by $\boldsymbol{x} = [\boldsymbol{y}_1^T, \boldsymbol{y}_2^T]^T$ an arbitrary variable vector. In non-linear SVM with Gaussian kernel, by introducing kernel function K as

$$K(\boldsymbol{x}, \boldsymbol{a}_c) = \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{a}_c\|^2}{\sigma_s^2}\right), \tag{3.9}$$

[5]Suppose there are two objects A and B. Then generally there can be five modes corresponding to the chains of the robot and the objects, hand, hand-A, hand-B, hand-A-B and hand-B-A.

[6]Hereafter, we suppose that the term 'object' includes the robot hand. And we let observation variables $\boldsymbol{y}^{(k)}$ correspond to object k, denoted by O^k.

[7]Among various classification methods, we selected SVM because of its usability for parameterizing mode boundaries in the later section using discrimination function $F(\boldsymbol{x})$ given by SVM.

where σ_s denotes a width parameter for the Gaussian kernel, the separation surface between two classes is expressed as

$$F(\boldsymbol{x}) \equiv \sum_{c=1}^{m_s} d_c w_c K(\boldsymbol{x}, \boldsymbol{a}_c) = 0, \tag{3.10}$$

where $d_c = 1$ (respectively -1) if $\boldsymbol{a}_c$ corresponds to mode i (respectively mode j) and $\boldsymbol{w}$ is the solution of the following optimization problem:

$$\min_{\boldsymbol{w}} \left\{ \frac{1}{2} \boldsymbol{w}^T Q \boldsymbol{w} - \boldsymbol{e}^T \boldsymbol{w} \right\}, \ \boldsymbol{e} = [1, \cdots, 1]^T \in \mathbb{R}^{n_s}. \tag{3.11}$$

Q is given by

$$Q = \frac{1}{\nu} + HH^T, \quad H = D[A, -\boldsymbol{e}], \quad \nu > 0, \tag{3.12}$$

where $D = \mathrm{diag}[d_1, \cdots, d_{m_s}]$, $A = [\boldsymbol{a}_1, \cdots, \boldsymbol{a}_{m_s}]^T$ and ν is a parameter for the optimization problem. For implementation of optimization in Eq. 3.11, Lagrangian SVM [16] is applied.

The boundary information obtained by SVM is used for (1) generation of mode keeping (and mode transition) motion controller and (2) trajectory planning using via points on the boundaries with parameterization. In the following, the discrimination function between mode i and j is denoted by $F_{i,j}(\boldsymbol{x})$.

3.4 Mode-Transition and Mode-Keeping Motion Mapping

In this section we consider contact-keeping motions of objects. For this, it is supposed that object O^k is contacting with object O^ℓ in a chain of objects (including the robot hand) and O^k is located at the nearer side to the robot hand. Therefore motion of O^ℓ is affected by O^k. In the following, a mapping will be described to determine necessary displacement of O_k when the desired displacement of O^ℓ is specified. Note here that in general there can be redundancy in the mapping between $\mathcal{Y}^k$ and $\mathcal{Y}^\ell$.[8] To derive the mapping, all the motion data expressing the contacted motion of objects O^k and O^ℓ are to be collected and stored throughout all random explorations. Let such data set be obtained as

$$\{\boldsymbol{y}_a^{(k)}, \boldsymbol{y}_a^{(\ell)}\}, \ a = 1, \cdots, n_g, \tag{3.13}$$

where n_g denotes the number of the samples. Now assume that at time t we have observed $\boldsymbol{y}^{(k)}(t)$ and $\boldsymbol{y}^{(\ell)}(t)$ on the observation variables for O^k and O^ℓ respectively and that desired displacement $\Delta \boldsymbol{y}^{(\ell)}$ for O^ℓ is specified. With these data set we wish to find $\Delta \boldsymbol{y}^{(k)}$ which realizes the desired displacement. This situation is depicted in Fig. 5 as motion transformation $G_{k,\ell}$ from $\mathcal{Y}^\ell$ to $\mathcal{Y}^k$. Motion transformation $G_{k,\ell}$ receives

[8] One example is the redundancy of the kinematics of the manipulator itself. When the manipulator has 3 DOF in planar motion and observation variables for the hand are only x-y configuration variables (excluding posture), there can be infinite motions of the joint angles that realize a desired 2D displacement of the hand. Another example is shown in Fig. 6 with O^k (circle) and O^ℓ (ellipsoid). In the figure, three different displacements of $\boldsymbol{y}^{(k)}$ can cause the same motion of $\boldsymbol{y}^{(\ell)}$. To determine $\Delta \boldsymbol{y}^{(k)}$ uniquely in such cases, we will pose a constraint to choose the smallest displacement of $\boldsymbol{y}^{(k)}$.

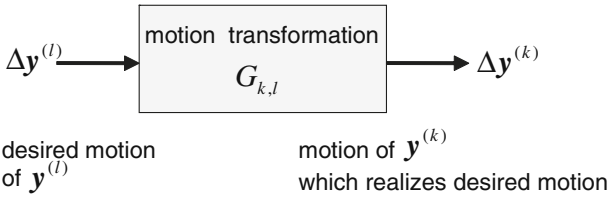


Fig. 5 Motion transformation between $\mathcal{Y}^k$ and $\mathcal{Y}^\ell$

$\Delta y^{(\ell)}$ as an input and gives $\Delta y^{(k)}$ as an output. As a result, it can generate a small motion of $y^{(k)}$ so as to realize given (desired) small motions of $y^{(\ell)}$.

Such a transformation can be constructed by an extension of sample-based function approximation techniques (e.g. [2, 28]). Displacement $\Delta y^{(k)}$ that realizes $\Delta y^{(\ell)}$ can be expressed as

$$\Delta y^{(k)} = \sum_{a=1}^{n_g} \tilde{w}_a y_a^{(k)} - y^{(k)}(t), \tag{3.14}$$

where $\tilde{w}_a, a = 1, \cdots, n_g$ denotes a weighting coefficient. $\tilde{w}_a$ is normalized as $\tilde{w}_a = w_a / \sum_b w_b$, where w_a is given as

$$w_a = \omega_a g\left(s_\omega, \exp\left(-\frac{\| y(t) - y_a^{(k)} \|^2}{2\sigma_1^2}\right)\right). \tag{3.15}$$

ω_a and s_ω are given as

$$\omega_a = \exp\left(-\frac{\| y^{(\ell)}(t) + \Delta y^{(\ell)} - y_a^{(\ell)} \|^2}{2\sigma_2^2}\right), \quad s_\omega = \sum_{a=1}^{n_g} \omega_a \tag{3.16}$$

and function $g(s, \beta)$ is defined as

$$g(s, \beta) = \beta + (1 - \beta)\left(1 - \frac{1}{\exp(-\zeta(s - \xi)) + 1}\right), \tag{3.17}$$

where $\zeta, \xi > 0$ are parameters for shaping the function. The shape of g is shown in Fig. 7. Observe from Eq. 3.16 that s_ω becomes large when there are sufficient number of samples that are close to $y^{(\ell)}(t) + \Delta y^{(\ell)}$. Therefore when s_ω is large we can

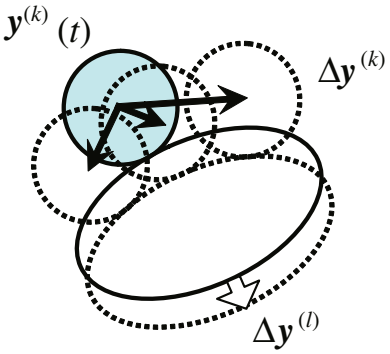


Fig. 6 Redundant mapping between groups of observation variables

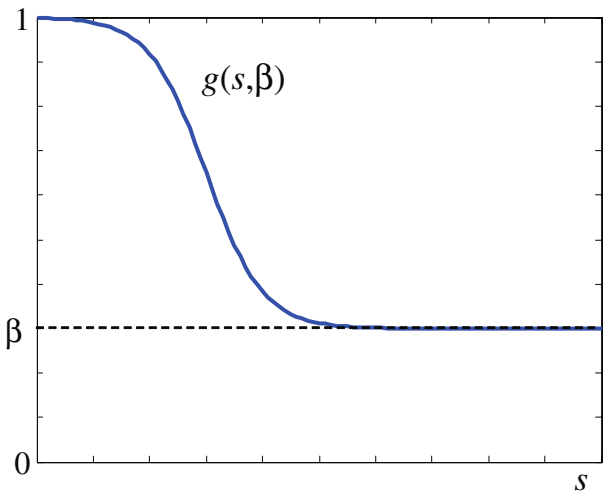


Fig. 7 Outline of $g(s, \beta)$

utilize generally more sufficient (possibly redundant) samples to determine $\Delta \boldsymbol{y}^{(\ell)}$. In such case, weighting coefficient w_a is calculated using both information of $\boldsymbol{y}_a^{(k)}$ and $\boldsymbol{y}_a^{(\ell)}$, and finally $\Delta \boldsymbol{y}^{(k)}$ is decided based on both $\boldsymbol{y}_a^{(k)}$ and $\boldsymbol{y}_a^{(\ell)}$. (note that $g(s, \beta) \simeq \beta$ in Eq. 3.15 because s_ω is large. Then w_a is decided by the product of the term including $\boldsymbol{y}_a^{(\ell)}$ and the term including $\boldsymbol{y}_a^{(k)}$). If s_ω is small, contrarily, there are not sufficient samples. In such case, w_a is decided only by considering the weight of $\boldsymbol{y}_a^{(\ell)}$ (note that $g(s, \beta) \simeq 1$ in Eq. 3.15 because s_ω is small. Then w_a reflects only the term including $\boldsymbol{y}_a^{(\ell)}$). Function $g(s, \beta)$ is defined in Eq. 3.17 as a continuous curve in order to smoothly connect the both cases.

Here we have to address a scalability problem of the proposed method, i.e., the number of samples needed for constructing mappings between motions become in general very large according to the increase of dimensions of input and output spaces, since the whole spaces have to be covered with samples. We will face the same problem also for estimation and parameterization of boundaries. A key idea to reduce this problem is to effectively use already obtained knowledge, which is also related to generalization of approximation, as can be seen in an example of ellipsoidal estimation of reachable region described in Section 3.6.2. The issue of scalability will be also discussed in Section 5.

To accumulate motion data to construct the transformation mapping through random explorations and also to apply the mapping for motion generation, the mode has to be kept unchanged, i.e., it has to stay in the same mode. In the following, by assuming that the current mode is i and $k = T(i)$, an algorithm will be presented to avoid mode changes from mode i to any other modes. For this, consider an arbitrary mode h ($\neq i$) and let $m = T(h)$. Recall that the boundary between modes i and h is given by $F_{i,h}(\boldsymbol{x}) = 0$ (see Eq. 3.10), where $F_{i,h}(x) > 0$ (respectively $F_{i,h}(x) < 0$) for mode i (respectively h). Define a potential function by $\Phi_{i,h}(\boldsymbol{x}) = \{F_{i,h}(\boldsymbol{x})\}^2$. The following algorithm can be used to generate random displacement of $\Delta \boldsymbol{y}(k)$ for the exploration in mode i. In case of the motion generation based on Eq. 3.14, rand() in step 2 should simply be replaced by $\Delta \boldsymbol{y}(k)$ by Eq. 3.14.

Algorithm 1 Motion generation with mode keeping

1. Set $found \leftarrow false$
2. While $found = false$ repeat:
   (a) Set $\Delta \boldsymbol{y}^{(k)}_{\rm tmp} \leftarrow$ rand()
   (a) If $F_{i,h}([\boldsymbol{y}^{(k)}(t)^T + \Delta \boldsymbol{y}^{(k)\,T}_{\rm tmp}, \boldsymbol{y}^{(m)}(t)]^T) > \varepsilon_b$, then $found \leftarrow true$
   (c) Else repeat the followings for n_o times:
      i Set $\Delta \boldsymbol{y}^{(k)}_{\rm tmp} \leftarrow \Delta \boldsymbol{y}^{(k)}_{\rm tmp} - \eta \dfrac{\partial \Phi_{i,h}}{\partial \boldsymbol{y}^{(k)}}$
      ii If $F_{i,h}([\boldsymbol{y}^{(k)}(t)^T + \Delta \boldsymbol{y}^{(k)\,T}_{\rm tmp}, \boldsymbol{y}^{(m)}(t)^T]^T) > \varepsilon_b$, then $found \leftarrow true$ and break
3. Output $\Delta \boldsymbol{y}^{(k)}$ with $\Delta \boldsymbol{y}^{(k)} = \Delta \boldsymbol{y}^{(k)}_{\rm tmp}$

Here, 'rand()' denotes a random vector generator, η a parameter to adjust the modification of the displacement of $\boldsymbol{y}^{(k)}$ and ε_b a threshold value to judge the closeness to the boundary of modes i and h.

3.5 Parameterization of Mode Boundary

Parameterization of boundaries between modes will be needed to determine a point for transition on the boundary for the total planning accompanying mode transitions. It can be also used to estimate reachable set based on the parametric approximation of the reachable set. In this section, a boundary parameterization method is presented based on the expression of mode boundary presented in Section 3.3. The potential function $\Phi(\boldsymbol{x})(= F(\boldsymbol{x})^2)$ which is defined by the SVM discrimination function is used to utilize the boundary information.

Let us consider parameterization of boundary between mode i and mode j, where $k = T(i)$ and $\ell = T(j)$. The parameterization is done for each discretized value $\boldsymbol{y}^{(\ell)}_{d(1)}, \cdots, \boldsymbol{y}^{(\ell)}_{d(n_\ell)}$ of $\boldsymbol{y}^{(\ell)}$ as shown in Fig. 8. In this paper, it is supposed for simplicity that boundaries are one-dimensional manifolds in two-dimensional space.[9] Thus, boundary is estimated by a curve on a plane of $\mathcal{Y}^k \subset \mathbb{R}^2$. The idea employed here for curve fitting is similar to the active contour models (e.g. [19, 31]) that were developed for edge detection in image processing. The curve model requires an easy computation for the parameterization and the model presented here meets the requirement because it applies fully segmented representation of the curve.

The curve model consists of multiple segments that are specified by nodes. Now let $\boldsymbol{q}_c \in \mathbb{R}^2$ denote a node on the boundary between mode i and mode j which we call hereafter c-th node and L denote the number of nodes.[10] A curve from c-th node

[9] One dimensional boundary corresponds to the case with objects and circular hand expressed in 2D space.

[10] Here we discuss the case with fixed number of nodes and the number of nodes is appropriately specified. It is also possible to adaptively change L based on the resultant shape of the estimated curve.

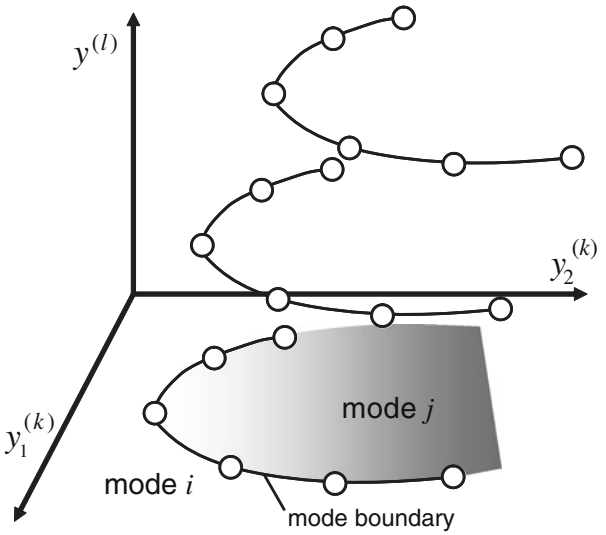


Fig. 8 Parameterization of boundary by nodes

to $(c+1)$-th node, which is defined as c-th segment of the curve, is defined using following equation as shown in Fig. 9.

$$\boldsymbol{z}^{(c)}(p') = \boldsymbol{q}_c + \boldsymbol{v}_{1c}p' + \boldsymbol{v}_{2c}p'^2, \tag{3.18}$$

where $\boldsymbol{v}_{1c}, \boldsymbol{v}_{2c} \in \mathbb{R}^2$ are coefficient vectors that give the shape of c-th segment of the curve. This curve is designed so that changing parameter p' from zero to one corresponds to the change on the curve from $\boldsymbol{q}_c$ to $\boldsymbol{q}_{c+1}$. By differentiating Eq. 3.18 by p', $\frac{\partial \boldsymbol{z}^{(c)}}{\partial p'}(p') = \boldsymbol{v}_{1c} + 2\boldsymbol{v}_{2c}p'$ is obtained. This gives a tangential vector of the curve. By considering the continuity of positions of the curve at the both edge of the curve segment, that is, by letting $\boldsymbol{z}^{(c)}(1) = \boldsymbol{z}^{(c+1)}(0)$,

$$\boldsymbol{q}_c + \boldsymbol{v}_{1c} + \boldsymbol{v}_{2c} = \boldsymbol{q}_{c+1}, \quad c = 0, \cdots, L-1 \tag{3.19}$$

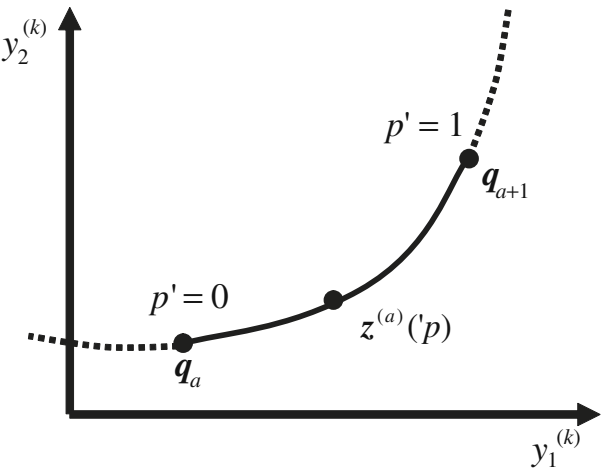


Fig. 9 A curve model constructed by nodes

is obtained as the condition for position continuity. Similarly, the condition for continuity of tangential vector is given by $\frac{\partial \boldsymbol{z}^{(c)}}{\partial p'}(1) = \frac{\partial \boldsymbol{z}^{(c+1)}}{\partial p'}(0)$, that is,

$$\boldsymbol{v}_{1c} + 2\boldsymbol{v}_{2c} = \boldsymbol{v}_{1c+1}, \quad c = 0, \cdots, L-2. \tag{3.20}$$

The above-mentioned conditions can be expressed in matrix form as

$$AV = Q, \quad A \equiv \begin{bmatrix} A_1 \\ A_2 \end{bmatrix}, Q \equiv \begin{bmatrix} Q_1 \\ O_{2(L-1)\times 1} \end{bmatrix}, \tag{3.21}$$

where matrices are defined as

$$A_1 = \begin{bmatrix} I_2 & I_2 & & & O \\ & & I_2 & I_2 & \\ & & & \ddots & \\ O & & & & I_2 \ I_2 \end{bmatrix} \in \mathbb{R}^{2L\times 4L}, \tag{3.22}$$

$$A_2 = \begin{bmatrix} I_2 & 2I_2 & -I_2 & & & O \\ & I_2 & 2I_2 & -I_2 & & \\ & & & \ddots & & \\ O & & & I_2 & 2I_2 & -I_2 \ O_2 \end{bmatrix} \in \mathbb{R}^{2(L-1)\times 4L} \tag{3.23}$$

and

$$V = \begin{bmatrix} \boldsymbol{v}_{10} \\ \boldsymbol{v}_{20} \\ \vdots \\ \boldsymbol{v}_{1L-1} \\ \boldsymbol{v}_{2L-1} \end{bmatrix} \in \mathbb{R}^{4L}, \ Q_1 = \begin{bmatrix} \boldsymbol{q}_1 - \boldsymbol{q}_0 \\ \vdots \\ \boldsymbol{q}_L - \boldsymbol{q}_{L-1} \end{bmatrix} \in \mathbb{R}^{2L}. \tag{3.24}$$

When the positions of nodes are given, the minimum-norm solution V for Eq. 3.21 can be calculated using the pseudo-inverse as $V = A^T(AA^T)^{-1}Q \equiv A^{\dagger}Q$. By connecting all segments, a one-dimensional submanifold $\boldsymbol{z}(s)$ can be defined as follows:

$$\boldsymbol{z}(s) \equiv \boldsymbol{z}^{(c)}(p'), \quad c \le s < c+1, \ p' = s - c, c \in \mathbb{Z}. \tag{3.25}$$

The positions of the nodes need to be modified if curves $\boldsymbol{z}(s)$ lie close to the boundary. By assuming that the boundary is smooth, this is realized by an incremental procedure to reduce energy defined on the curve, where energy function of the curve is defined with coefficient α_{ext} and α_{int} as

$$E = \alpha_{\text{ext}} E_{\text{ext}} + \alpha_{\text{int}} E_{\text{int}}. \tag{3.26}$$

An internal energy for the curve is defined as

$$E_{\text{int}} = \int_0^L \left\| \frac{\partial^2 \boldsymbol{z}(s)}{\partial s^2} \right\|^2 ds. \tag{3.27}$$

Using Eq. 3.18, E_{int} can be expressed as

$$E_{\text{int}} = \sum_{c=1}^{L} \int_0^1 \left\| \frac{\partial^2 \boldsymbol{z}^{(c)}}{\partial p'^2} \right\|^2 dp' = 4 \sum_{c=1}^{L} \|\boldsymbol{v}_{2c}\|^2. \tag{3.28}$$

E_{ext} is defined so that the curve coincides with the boundary when E_{ext} is minimized. By using the potential function $\Phi_{i,j}$, it can be defined as

$$E_{\text{ext}} = \sum_{c=1}^{L} \Phi_{i,j}([\boldsymbol{q}_c^T, \boldsymbol{y}_{(d)}^{(\ell)\,T}]^T), \tag{3.29}$$

where $\boldsymbol{y}_{(d)}^{(\ell)}$ denotes discretized value of $\boldsymbol{y}^{(\ell)}$. By iterating the following gradient descent update

$$\boldsymbol{q}_c \leftarrow \boldsymbol{q}_c - \eta_n \frac{\partial E}{\partial \boldsymbol{q}_c}, \quad c = 1, \cdots, L, \tag{3.30}$$

the positions of nodes $1, \cdots, L$ are modified so that the curve lies more closely to the boundary while keeping the smoothness of the curve. The positions of the end nodes 1 and L have to be updated so that they expand up to the furthest (in the direction of $\frac{\partial \boldsymbol{z}^{(L-1)}}{\partial p'}(1)$ and $-\frac{\partial \boldsymbol{z}^{(1)}}{\partial p'}(0)$ from the positions of 1st and L-th nodes, respectively) observed data in $B_{i,j}$. After convergence of the iteration of Eq. 3.30, any point on the boundary can be expressed by parameter s.

In the following sections, parameterization of boundary between mode i and j will be presented as $\boldsymbol{z}_{i,j}(s)$ to identify modes.

3.6 Expression for Reachable Region

For planning trajectories to realize the desired configurations of objects, it is required to identify where the object can reach (reachable region). In this section, a describing method is given for the reachable region for observation variable $\boldsymbol{y}^{(\ell)}$ (corresponding to O^ℓ). Before this, let us consider the pushing task of a triangular object as an example (see Fig. 10). When the hand is touching the object at the left (, respectively right) leg, the reachable region of the object is restricted to the right-hand (, respectively left-hand) space of the current object position. This shows that the reachable region of an object changes according to at which part of object- (hand-) surface the contact takes place. In the general case, the reachable region (, say of object O^ℓ) is characterized by the following factors:

(i) Set of objects forming a chain whose tip object is O^ℓ and their arranging order
(ii) Initial locations of the the objects in the chain
(iii) Contact positions of any two mutually connected objects in the chain

(i) is reflected in the observation on mode transitions since each mode corresponds to a chain. (ii) can be expressed by the observation variables that start moving at the instant of mode changes. (iii) can be specified by the boundary parameters recorded when each contact between two objects happened. Since positions of objects and their contact positions take continuous values, the reachable regions of objects depend on these continuous variables and therefore are characterized in general by infinitely many parameters. Here to get an explicit description of the reachable regions, we will introduce a finite number approximation for them by discretizing continuous variables. The left hand of Fig. 11 shows the conceptual diagram for the discretization, where the black small circles on the vertical axis of $\boldsymbol{y}^{(\ell)}$ indicates the discretization of observation variable $\boldsymbol{y}^{(\ell)}$ and the black small circles on the curves

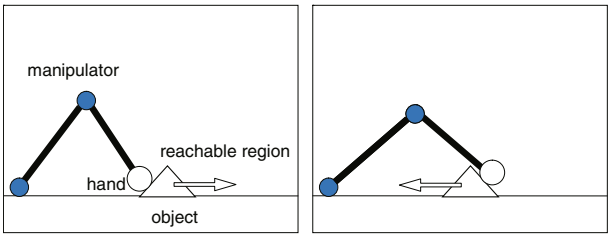


Fig. 10 Different reachable region depending on contact point and object configuration

(that designates a boundary between two modes) indicates the discretization of a parameter for the mode boundary.[11]

Now, let us consider the reachable region of object O^{ℓ_c}. We assume that O^{ℓ_c} is the tip object of mode i_c ($\ell_c = T(i_c)$) and mode transitions have occurred as $i_1 \to i_2 \to \cdots \to i_c$. This sequence of mode transitions is expressed by $\boldsymbol{\tau} = [i_1, \cdots, i_c]^T \in \mathbb{N}^c$ and it contains the information corresponding to (i). Let i and j be any two adjacent mode transition in $\boldsymbol{\tau}$ and let O^k and O^ℓ be the contacting objects corresponding to them, i.e., $k = T(i), \ell = T(j)$. It is supposed that O^k is located to the nearer position to the robot hand. Discretized values of observation variables for O^ℓ, which corresponds to (ii), is defined as $\mathcal{Y}_D^\ell = \{\boldsymbol{y}_{(1)}^{(\ell)}, \cdots, \boldsymbol{y}_{(n(\ell))}^{(\ell)}\}$, where $n(\ell)$ denotes the number of discretization of the observation variable. This is shown in the left hand of Fig. 11, where the horizontal plane indicates $\mathcal{Y}^k$ and the curves in the planes denote the boundary sets between modes i and j. In the figure, space of $\boldsymbol{y}^{(\ell)}$ (vertical axis) is discretized with equal distances as shown by the small circles on the axis. Recall that s is the parameter on the boundary defined in Eq. 3.25 and corresponding to the condition (iii) about contact, the discretization of boundary parameters (between mode i and j) is introduced as $\mathcal{S}_{D(b)}^{(i,j)} = \{s_{(b)}^{(i,j),1}, \cdots, s_{(b)}^{(i,j),n(i,j)}\}, b = 1, \cdots, n(\ell)$, where $s_{(b)}^{(i,j),a}$ denotes a discretized parameter on the boundary of mode transition from i to j when the discretized observation variable takes value $\boldsymbol{y}_{(b)}^{(\ell)}$ and $n(i, j)$ denotes the number of discretization of the boundary. The right hand of Fig. 11 indicates the conceptional picture of reachable regions. Each ellipse represents the reachable region of object O^ℓ caused by the contact between objects O^k and O^ℓ where the contact initiated at $\boldsymbol{y}^{(\ell)} = \boldsymbol{y}_{(b)}^{(\ell)}$ and the boundary parameter was $s_{(b)}^{(i,j),a}$. The horizontal plane indicates the space $\mathcal{Y}^\ell$ (, here for drawing purpose it is assumed to be two-dimensional whereas it was represented by one-dimensional space in the left hand figure).

Based on the above-mentioned notation of $\mathcal{Y}_D^\ell$ and $\mathcal{S}_{D(b)}^{(i,j)}$, estimation of reachable region is given. Let $\mathcal{T}$ denote the set of experienced mode transition sequences. $\mathcal{T}$ is updated whenever a new mode transition $\boldsymbol{\tau}$ is experienced, i.e.,

$$\mathcal{T} \leftarrow \mathcal{T} \cup \{\boldsymbol{\tau}\} \quad \text{if} \quad \boldsymbol{\tau} \notin \mathcal{T} \wedge I_C^{(i_1)} \subset I_C^{(i_2)} \subset \cdots \subset I_C^{(i_c)}. \tag{3.31}$$

The second condition stands for that new mode sequence is memorized only when the total number of objects in the chain increases. By this, mode transitions with separation between objects are omitted. Identification of mode transition for mode

[11] Note that the idea of discretization is same as shown in Fig. 8, though the objective of discretization in Section 3.5 was to parameterize the boundary by multiple segments of curves.

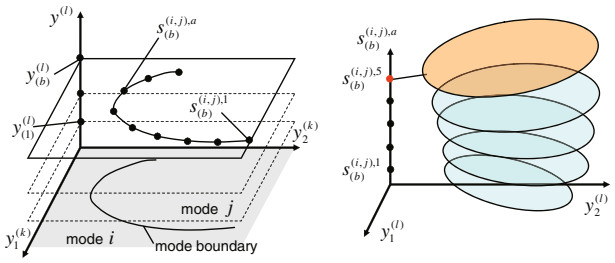


Fig. 11 Discretization of mode boundary and observation variable and extrapolation with parameter on boundary

transition parameter $s^{(i,j)}$ and observation variable $\boldsymbol{y}^{(\ell)}$ can be given as $\boldsymbol{g}_{\boldsymbol{\tau}} = \{\bar{\boldsymbol{s}}_{\boldsymbol{\tau}}, \bar{\boldsymbol{y}}_{\boldsymbol{\tau}}\}$, where

$$\bar{\boldsymbol{y}}_{\boldsymbol{\tau}} = [\boldsymbol{y}^{(\ell_2)}_{(b_2)}, \cdots, \boldsymbol{y}^{(\ell_c)}_{(b_c)}],\ b_e = \arg\min_b \|\boldsymbol{y}^{(\ell_e)}_{(b)} - \boldsymbol{y}^{(\ell_e)}\|,\ \ell_e = T(i_e),\ e = 2, \cdots, c \quad (3.32)$$

and

$$\bar{\boldsymbol{s}}_{\boldsymbol{\tau}} = [s^{(i_1,i_2),a_{1,2}}_{(b)}, \cdots, s^{(i_{c-1},i_c),a_{c-1,c}}_{(b)}]^T, b = b_{e+1},$$
$$a_{e,e+1} = \arg\min_a \|s^{(i_e,i_{e+1}),a}_{(b)} - s^{(i_e,i_{e+1}),a}\|,\ e = 1, \cdots, c-1, . \quad (3.33)$$

Note here that $\boldsymbol{g}_{\boldsymbol{\tau}}$ contains discrete expression of (i), (ii) and (iii). Let $\mathcal{D}^{\ell}_{\boldsymbol{\tau}}(\boldsymbol{g}_{\boldsymbol{\tau}})$ denote the set of all observed $\boldsymbol{y}^{(\ell)}$ where mode transitions specified by $\boldsymbol{\tau}$ and $\boldsymbol{g}_{\boldsymbol{\tau}}$ have taken place (here we let $\ell = \ell_c$ for simplicity of notation), which will be used as a data base for estimating the reachable regions. When $\boldsymbol{y}^{(\ell)}(t)$ is observed during exploration, $\mathcal{D}^{\ell}_{\boldsymbol{\tau}}(\boldsymbol{g}_{\boldsymbol{\tau}})$ is updated as

$$\mathcal{D}^{\ell}_{\boldsymbol{\tau}}(\boldsymbol{g}_{\boldsymbol{\tau}}) \leftarrow \mathcal{D}^{\ell}_{\boldsymbol{\tau}}(\boldsymbol{g}_{\boldsymbol{\tau}}) \cup \{\boldsymbol{y}^{(\ell)}(t)\} \ \text{ if } \ \|\boldsymbol{y}^{(\ell)}(t) - \boldsymbol{y}^{(\ell)}_a\| > R_r, {}^{\forall}\boldsymbol{y}^{(\ell)}_a \in \mathcal{D}^{(\ell)}_{\boldsymbol{\tau}}(\boldsymbol{g}_{\boldsymbol{\tau}}) \quad (3.34)$$

where $R_r > 0$ denotes a threshold value and $\boldsymbol{y}^{(\ell)}_a$ denotes a-th element in $\mathcal{D}^{\ell}_{\boldsymbol{\tau}}(\boldsymbol{g}_{\boldsymbol{\tau}})$.

3.6.1 Estimation of Reachable Region with Raw Samples

A straightforward approach to estimate reachable region is to sum up hyperspheres whose centers are observed samples as shown in the left hand of Fig. 12. Based on this idea, reachable region of O^{ℓ} via mode transition $\boldsymbol{\tau}$ and $\boldsymbol{g}_{\boldsymbol{\tau}}$ can be expressed as

$$\mathcal{R}^{\ell}_{\boldsymbol{\tau}}(\boldsymbol{g}_{\boldsymbol{\tau}}) = \{\boldsymbol{y}^{(\ell)} \in \mathcal{Y}^{\ell} | {}^{\exists}\boldsymbol{y}^{(\ell)}_m \in \mathcal{D}^{\ell}_{\boldsymbol{\tau}}(\boldsymbol{g}_{\boldsymbol{\tau}}), \|\boldsymbol{y}^{(\ell)} - \boldsymbol{y}^{(\ell)}_m\| \leq R_r\}, \quad (3.35)$$

which will be used in motion planning described in Section 3.7.

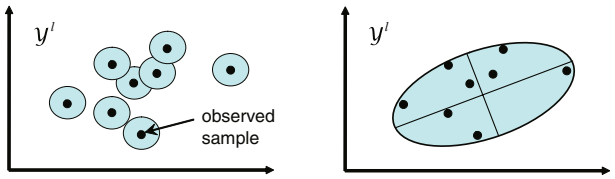


Fig. 12 Estimation of reachable region by two ways: directly approximating by observed samples and approximating with parameterized geometrical model

3.6.2 Estimation of Reachable Region with Parameterized Smooth Function

Another approach to estimate reachable region is to apply parameterized geometrical models as indicated in the right hand of Fig. 12. This model is suitable to be used when the number of available observation samples are not enough. Let us consider a case in the right hand of Fig. 11, where we do not have sufficient number of observed samples for $s_{(5)}^{(i,j),a}$. In this case, it is possible to estimate the reachable region for $s_{(b)}^{(i,j),5}$ by extrapolating parameters of reachable regions for $s_{(b)}^{(i,j),a}$, $a = 1, \cdots, 4$ as can be seen in the figure (a yellow ellipsoid).

Now we consider again the case where mode i_c has been reached by mode transition $\boldsymbol{\tau} = [i_1, \cdots, i_c]^T$, where $\ell = T(i_c)$. Assume that the mode transition is specified by $\boldsymbol{g}_{\boldsymbol{\tau}} = \{\bar{\boldsymbol{s}}_{\boldsymbol{\tau}}, \bar{\boldsymbol{y}}_{\boldsymbol{\tau}}\}$, where

$$\bar{\boldsymbol{y}}_{\boldsymbol{\tau}} = [\boldsymbol{y}_{(b_2)}, \cdots, \boldsymbol{y}_{(b_c)}],\ \bar{\boldsymbol{s}}_{\boldsymbol{\tau}} = [s_{(b)}^{(i_1,i_2),a_{1,2}}, \cdots, s_{(b)}^{(i_{c-1},i_c),a_{c-1,c}}]^T. \tag{3.36}$$

Among these parameters for the mode transitions, we are interested in the boundary parameter of the latest mode transition of $i_{c-1} \to i_c$, that is, where $O^k, k = T(i_{c-1})$ touched with O^ℓ. So we rewrite $\boldsymbol{g}_{\boldsymbol{\tau}}$ as

$$\boldsymbol{g}_{\boldsymbol{\tau}} = \left\{ \left[\bar{\boldsymbol{s}}_{\boldsymbol{\tau}}^{0\,T}, s_{(b)}^{(i_{c-1},i_c),a_{c-1,c}} \right]^T, \bar{\boldsymbol{y}}_{\boldsymbol{\tau}} \right\}. \tag{3.37}$$

That is, $\bar{\boldsymbol{s}}_{\boldsymbol{\tau}}^0$ denotes discretized parameters on the mode transition boundaries except for the latest transition $i_{c-1} \to i_c$. Then we wish to get an approximation for the reachable region for O^ℓ (denoted by $\mathcal{R}^\ell_{\boldsymbol{\tau},\bar{\boldsymbol{s}}_{\boldsymbol{\tau}}^0,\bar{\boldsymbol{y}}_{\boldsymbol{\tau}}}(s_{(b)}^{(i_{c-1},i_c),a})$) realized via mode transition $\left[\bar{\boldsymbol{s}}_{\boldsymbol{\tau}}^{0\,T}, s_{(b)}^{(i_{c-1},i_c),a}\right]$ and $\bar{\boldsymbol{y}}_{\boldsymbol{\tau}}$. $\mathcal{R}^\ell_{\boldsymbol{\tau},\bar{\boldsymbol{s}}_{\boldsymbol{\tau}}^0,\bar{\boldsymbol{y}}_{\boldsymbol{\tau}}}(s_{(b)}^{(i_{c-1},i_c),a})$ is in the form

$$\mathcal{R}^\ell_{\boldsymbol{\tau},\bar{\boldsymbol{s}}_{\boldsymbol{\tau}}^0,\bar{\boldsymbol{y}}_{\boldsymbol{\tau}}}(s_{(b)}^{(i_{c-1},i_c),a}) = \{\boldsymbol{y} \in \mathbb{R}^{n_\ell} | \mathcal{F}_{\boldsymbol{\phi}}(\boldsymbol{y}) \le 0\}, \tag{3.38}$$

where $\mathcal{F}_{\boldsymbol{\phi}}(\boldsymbol{y})$ denotes a quadratic form with parameters $\boldsymbol{\phi} = [\lambda_1, \cdots \lambda_{n_\ell}, \boldsymbol{\psi}^T, \boldsymbol{d}^T]^T$ described by

$$\mathcal{F}_{\boldsymbol{\phi}}(\boldsymbol{y}) = (\boldsymbol{y} - \boldsymbol{d})^T R(\boldsymbol{\psi})^T \begin{bmatrix} \lambda_1^2 & & \\ & \ddots & \\ & & \lambda_{n_\ell}^2 \end{bmatrix} R(\boldsymbol{\psi})(\boldsymbol{y} - \boldsymbol{d}) - 1, \quad \lambda_1, \cdots, \lambda_{n_\ell} > 0, \tag{3.39}$$

where $R(\boldsymbol{\psi}) \in \mathbb{R}^{n_\ell \times n_\ell}$ denotes the rotation matrix with parameter $\boldsymbol{\psi} \in \mathbb{R}^{n_\ell(n_\ell-1)/2}$.

Parameter $\boldsymbol{\phi}$ is determined so that $\mathcal{R}^\ell_{\boldsymbol{\tau},\bar{\boldsymbol{s}}_{\boldsymbol{\tau}}^0,\bar{\boldsymbol{y}}_{\boldsymbol{\tau}}}(s_{(b)}^{(i_{c-1},i_c),a})$ contains all of the observation variables in $\mathcal{D}^\ell_{\boldsymbol{\tau}}(\boldsymbol{g}_{\boldsymbol{\tau}})$ while making the volume of $\mathcal{R}^\ell_{\boldsymbol{\tau},\bar{\boldsymbol{s}}_{\boldsymbol{\tau}}^0,\bar{\boldsymbol{y}}_{\boldsymbol{\tau}}}(s_{(b)}^{(i_{c-1},i_c),a})$ minimal. Therefore, parameter vector $\boldsymbol{\phi}$ is decided by minimizing the following cost function:

$$J(\boldsymbol{\phi}) = \int_{\mathcal{R}^\ell_{\boldsymbol{\tau},\bar{\boldsymbol{s}}_{\boldsymbol{\tau}}^0,\bar{\boldsymbol{y}}_{\boldsymbol{\tau}}}(s_{(b)}^{(i_{c-1},i_c),a})} d\boldsymbol{y}, \quad \text{subject to} \quad \mathcal{F}_{\boldsymbol{\phi}}(\boldsymbol{y}_m^{(\ell)}) \le 0, {}^\forall \boldsymbol{y}_m^{(\ell)} \in \mathcal{D}^\ell_{\boldsymbol{\tau}}(\boldsymbol{g}_{\boldsymbol{\tau}}). \tag{3.40}$$

This estimation is processed for each discretized boundary of $s_{(b)}^{(i_{c-1},i_c),a}$, $a = 1, \cdots, n(i_{c-1}, i_c)$ which satisfies $\left|\mathcal{D}^\ell_{\boldsymbol{\tau}}(\boldsymbol{g}_{\boldsymbol{\tau}})\right| \ge N_d$,[12] where N_d denotes a threshold for

[12] $|S|$ denotes the number of elements in set S.

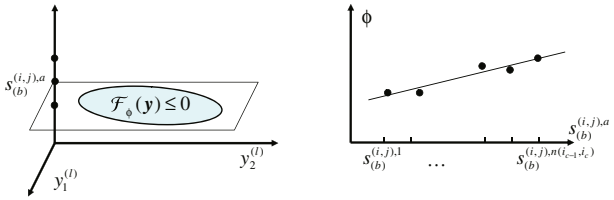


Fig. 13 Interpolation using parameter vectors

the number of observed samples. Let $a_1, \cdots, a_{n_d}$ denote the discretization indices of the boundary that meet the above condition for the number of samples, where $n_d \le n(i_{c-1}, i_c)$ denotes the number of discretized boundaries that have sufficient numbers of samples.

Next step is to find an approximation of the parameter vector $\boldsymbol{\phi}(s)$ as a function of boundary parameter s based on $\boldsymbol{\phi}_b$, as shown in the right hand of Fig. 13. A linear polynomial approximation model can be expressed by

$$\boldsymbol{\phi}(s) = \boldsymbol{\phi}_1 s + \boldsymbol{\phi}_0, \quad s \in \mathbb{R},\ \boldsymbol{\phi}_0, \boldsymbol{\phi}_1 \in \mathbb{R}^{n_p}, \tag{3.41}$$

where $\boldsymbol{\phi}_0$ and $\boldsymbol{\phi}_1$ are coefficient vectors. The least square approximation of coefficient vectors $\{\boldsymbol{\phi}_0, \boldsymbol{\phi}\}$ when $s_{(b)}^{(i_{\mathrm{pre}},i),a_1}, \cdots, s_{(b)}^{(i_{\mathrm{pre}},i),a_{n_d}}$ and $\hat{\boldsymbol{\phi}}_{a_1}, \cdots, \hat{\boldsymbol{\phi}}_{a_{n_d}}$ are given are calculated by

$$[\bar{\boldsymbol{\phi}}_1\ \bar{\boldsymbol{\phi}}_0]^T = S^\dagger A_\phi, \quad S = \begin{bmatrix} s_{(b)}^{(i_{\mathrm{pre}},i),a_1} & 1 \\ & \vdots \\ s_{(b)}^{(i_{\mathrm{pre}},i),a_{n_d}} & 1 \end{bmatrix}, A_\phi = \begin{bmatrix} \hat{\boldsymbol{\phi}}_{a_1}^T \\ \vdots \\ \hat{\boldsymbol{\phi}}_{a_{n_d}}^T, \end{bmatrix}, \tag{3.42}$$

where $\hat{\boldsymbol{\phi}}_a$ denotes the solution for the optimization problem of Eq. 3.40. Using the obtained approximation of $\bar{\boldsymbol{\phi}}(s) = \bar{\boldsymbol{\phi}}_1 s + \bar{\boldsymbol{\phi}}_0$, estimation of the reachable region is given as

$$\bar{\mathcal{R}}^{\ell}_{\boldsymbol{\tau}, \bar{\boldsymbol{s}}^0_{\boldsymbol{\tau}}, \bar{\boldsymbol{y}}_{\boldsymbol{\tau}}}(s) = \{\boldsymbol{y} \in \mathbb{R}^{n_\ell} | \mathcal{F}_{\bar{\boldsymbol{\phi}}(s)}(\boldsymbol{y}) \le 0\}, \tag{3.43}$$

which gives the interpolation and the extrapolation mentioned in the example.

3.7 Trajectory Generation and Control with Mode Transitions

Planning with mode transitions described in this section corresponds to the dynamical shift of planning space, because different observation variables are used in planning within each mode. The total algorithm of trajectory generation and control is given in Algorithm 2. A task is given to the robot system as a target configuration of a certain object.[13] Let the target object be O^{k_*} and assume its target configuration $\boldsymbol{y}_*^{(k_*)} \in \mathcal{Y}^{k_*}$ is given. The robot interprets the given information as a target mode (step 2. in Algorithm 2). Using mode transition and reachable region

[13]Here it is assumed that configurations of other objects and the robot arm are not specified as a task.

database, the robot finds an appropriate transition sequence of modes and pair of mode transition parameters and observation variable at mode transition (steps 3. and 4.). At step 5., the robot starts control of its body to realize mode transition sequence of $\boldsymbol{\tau}^\dagger$ with transition parameter $\boldsymbol{g}^\dagger_{\boldsymbol{\tau}^\dagger}$. At each mode, the robot aims at a target (subgoal) point on the mode transition boundary which is specified by discretized parameter of boundary curve (b). Once a subgoal is determined, the robot repeats small movements toward the subgoal while trying to keep current mode. For mode keeping, Algorithm 1 can be used by replacing $\Delta \boldsymbol{y}^{(i)}_{\rm tmp}$ by $\Delta \boldsymbol{y}^{(i)}$ given in (c)-i. The desired motion $\Delta \boldsymbol{y}^{(i)}$ is realized by transformations of motions as shown in Fig. 14.

Algorithm 2 Trajectory generation and control

1. Given: target object k_* and its configuration $\boldsymbol{y}^{(k_*)}_*$
2. Find target mode i_* that satisfies $k_* = T(i_*)$
3. Find $\boldsymbol{\tau}^\dagger \in \mathcal{T}$ s.t. $(^\exists a \in \mathbb{N}$ s.t. $i_* = \tau^\dagger_a)$
4. Find $\boldsymbol{g}^\dagger_{\boldsymbol{\tau}^\dagger} = \{\bar{\boldsymbol{s}}^\dagger_{\boldsymbol{\tau}}, \bar{\boldsymbol{y}}^\dagger_{\boldsymbol{\tau}}\}$ s.t. $\boldsymbol{y}^{(k_*)}_* \in \mathcal{R}^{k_*}_{\boldsymbol{\tau}^\dagger}(\boldsymbol{g}^\dagger_{\boldsymbol{\tau}^\dagger}) \wedge \bar{\boldsymbol{y}}^\dagger_{\boldsymbol{\tau}^\dagger} = [\boldsymbol{y}_{(b_2)}, \cdots, \boldsymbol{y}_{(b_c)}]^T$, where $b_e = \arg\min_b \| \boldsymbol{y}^{(\ell_e)}_{(b)} - \boldsymbol{y}^{(\ell_e)}(0) \|$, $\ell_e = T(i_e), e = 2, \cdots, c$
5. Set current mode as $i = 1$ and current tip of the chain as $k = 1$. Repeat following trajectory generation and control procedures until target configuration is achieved, i.e. $\| \boldsymbol{y}^{(k_*)}(t) - \boldsymbol{y}^{(k_*)}_* \| < \varepsilon_d$ holds:
   (a) If $i \neq i_*$, find next mode as $j = \tau^\dagger_\iota$ where $\iota = \kappa + 1, \tau^\dagger_\kappa = i$. Let $\ell = T(j)$ and set subgoal on mode boundary by $\boldsymbol{y}^{(k)}_\dagger = \boldsymbol{z}_{i,j}\left(s^{(i,j),a^\dagger}_{(b_0)}\right)$, where $\bar{\boldsymbol{s}}^\dagger_{\boldsymbol{\tau}^\dagger} = [\cdots, s^{(i,j),a^\dagger}_{(b_0)}, \cdots]^T, b_0 = \arg\min_b \| \boldsymbol{y}^{(\ell)}_{(b)} - \boldsymbol{y}^{(\ell)}(t) \|$
   (b) If $i = i_*$, set target observation variable as $\boldsymbol{y}^{(k)}_\dagger = \boldsymbol{y}^{(k_*)}_*$
   (c) Repeat small movement of $\Delta \boldsymbol{y}^{(k)}$ until target point is reached (i.e. $\| \boldsymbol{y}^{(k)}(t) - \boldsymbol{y}^{(k)}_\dagger \| \leq \varepsilon_d$)
      i Calculate displacement $\Delta \boldsymbol{y}^{(k)}$ by modifying $\Delta \boldsymbol{y}^{(k)}_{\rm tmp} = \gamma \frac{(\boldsymbol{y}^{(k)}_\dagger - \boldsymbol{y}^{(k)}(t))}{\| \boldsymbol{y}^{(k)}_\dagger - \boldsymbol{y}^{(k)}(t)) \|}$ with Algorithm 1
      ii Calculate $\Delta \boldsymbol{y}^{(k_m)}$ using $\Delta \boldsymbol{y}^{(k)}$, $\Delta \boldsymbol{y}^{(k_{m-1})}$ using $\Delta \boldsymbol{y}^{(k_m)}$, $\cdots$, $\Delta \boldsymbol{y}^{(k_0)}$ using $\Delta \boldsymbol{y}^{(k_1)}$ with Eq. 3.14, where $\{k_0, k_1, \cdots, k_m, k\}$ denotes the sequence of the chain of objects in mode i ($k = T(i), k_0 = 1$).
   (d) Set $i \leftarrow j$

In 5(c)-i and 5(c)-ii, displacements $\Delta \boldsymbol{y}^{(k)}, \Delta \boldsymbol{y}^{(k_m)}, \cdots$ are calculated so that the transition to irrelevant modes and the transition to the desired mode at undesired point on the boundary are avoided. We introduce a threshold value ε_p and replace conditions of 2(a) and 2(b)-ii in Algorithm 1 by

$$F_{i,h}([\boldsymbol{y}^{(k)}(t)^T + \Delta \boldsymbol{y}^{(k)\,T}_{\rm tmp}, \boldsymbol{y}^{(m)}(t)^T]^T) > \varepsilon_b \wedge$$
$$(F_{i,j}([\boldsymbol{y}^{(k)}(t)^T + \Delta \boldsymbol{y}^{(k)\,T}_{\rm tmp}, \boldsymbol{y}^{(\ell)}(t)^T]^T) > \varepsilon_b \vee \| \boldsymbol{y}^{(k)} + \Delta \boldsymbol{y}^{(k)}_{\rm tmp} - \boldsymbol{y}^{(k)}_\dagger \| > \varepsilon_p), \quad (3.44)$$

where h denotes irrelevant mode ($h \neq j$) in this application. By this, mode transition to j at undesired point on the boundary that is not close to $\boldsymbol{y}^{(k)}_\dagger$ can be avoided.

Step 4. in the algorithm is based on the direct memory-based estimation of the reachable region described in Section 3.6.1. An alternative approach is to use

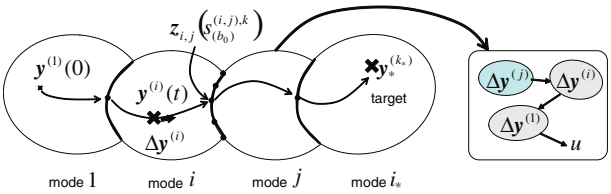


Fig. 14 Planning with mode transition

the estimation by the parameterization-based estimation described in Section 3.6.2, where the procedure will be replaced by the following.

$$4'.\quad \text{Find } \boldsymbol{g}^{\dagger}_{\boldsymbol{\tau}^{\dagger}} = \left\{ \left[\bar{\boldsymbol{s}}^{0\dagger T}_{\boldsymbol{\tau}}, s^{\dagger} \right]^T, \bar{\boldsymbol{y}}^{\dagger}_{\boldsymbol{\tau}} \right\} \quad \text{s.t. } \boldsymbol{y}^{(k_*)}_* \in \mathcal{R}^{k_*}_{\boldsymbol{\tau}^{\dagger}, \bar{\boldsymbol{s}}^{0\dagger}_{\boldsymbol{\tau}^{\dagger}}, \bar{\boldsymbol{y}}_{\boldsymbol{\tau}^{\dagger}}}(s^{\dagger}) \wedge$$

$$\bar{\boldsymbol{y}}^{\dagger}_{\boldsymbol{\tau}^{\dagger}} = [\boldsymbol{y}_{(b_2)}, \cdots, \boldsymbol{y}_{(b_c)}]^T,\ b_e = \arg\min_b \| \boldsymbol{y}^{(\ell_e)}_{(b)} - \boldsymbol{y}^{(\ell_e)}(0) \|,\ \ell_e = T(i_e), e = 2, \cdots, c.$$

4 Simulation

The proposed mode generation and trajectory generation algorithms are implemented and evaluated in three types of settings by simulations. In the first setting, the manipulator moves in a vertical plane and the objects are restricted to move on a line (floor) as indicated in the left hand of Fig. 15. Thus motions of objects are constrained in a one-dimensional line because of the contact with the floor as long as the objects keep the same posture and do not rotate. In the second setting, the objects and the hand move on a plane (2D space) without any constraint as depicted in the right hand of Fig. 15. Planning and motion generation are more complicated in the second case because of the two-dimensional motions of the object. The proposed planning framework is evaluated in the first case with multiple objects including approximation of the boundary with SVM. In the second case, motion generation on a 2D plane with redundant mapping between groups of the observation variables and approximation of reachable region are evaluated. In the third setting, L-shaped tool is utilized to move the object close to the robot itself. It is shown that the proposed framework is applicable to a similar problem setting to learning tool affordance [27].

4.1 Case 1: Multiple Objects with One-dimensional Constraint

The robot manipulator has two joints and there are two objects, rectangular and circular objects as shown in the left hand of Fig. 15. The end effector of the robot

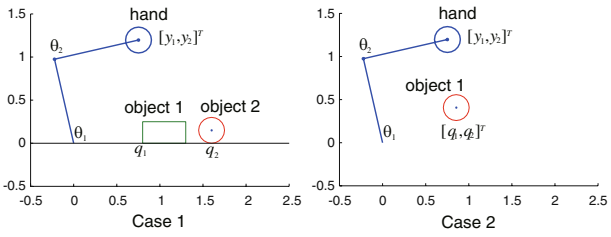


Fig. 15 Simulation settings: robot hand and objects

hand is also circularly shaped. The state variables are the joint angles of the manipulator $\boldsymbol{\theta} \in \mathbb{R}^2$ and positions of two objects $q_1, q_2 \in \mathbb{R}$. The observation variables are the position of the robot hand (center of circle) $[y_1, y_2]^T$, the position of the rectangular object (object 1) $y_3 (= q_1)$ and the position of the circular object (object 2) $y_4 (= q_2)$. Initial position of object 1 is randomly chosen in $0.6 \leq q_1 \leq 0.9$, while position of object 2 is fixed at $q_2 = 1.6$. The manipulator and the objects are initialized every 100 steps.

Exploration and learning are done by the followings:

1. Move the arm randomly for 3,000 steps, including the initializations per 100 steps.
2. Build mapping between $[\theta_1, \theta_2]^T$ and $[y_1, y_2]^T$, estimate boundary between mode 1 and mode 2,[14] and generate parameterization on the boundary.
3. Explore mode 2 using the motion generator which keeps mode 2 for 200 steps.
4. Build mapping between q_1 and q_2, estimate boundary between mode 2 and mode 3 (, where object 2 is moving).
5. Explore modes 2 and 3 using mode keeping motion generator described in Section 3.4 for 200 steps to estimate reachable region.

Explorations in procedure 3 and 5 are initialized when the robot fails to keep mode 2 or 3 in addition to initializations per 100 steps. For the estimation of the reachable region at procedure 5, direct memory-based approximation described in Section 3.6.1 is used. It is expected that while procedure 1, contact between the hand and object 1 happens frequently enough to estimate boundary. Similarly it is expected that sufficient motion information of object 2 is obtainable to finally estimate the reachable region. The mapping and boundary building in 4. is formally done but actually not so important because motions of object 1 and 2 are restricted on a line and the boundary of mode switching is just a point.

After the total exploration, three tasks are given to the robot system:

- Task 1: Move the end effector to $[y_1, y_2]^T = [1.3, 0.6]^T$.
- Task 2: Move object 1 to $q_1 = 0.5$.
- Task 3: Move object 2 to $q_2 = 1.9$.

First, the robot acquired the relation between its joint angles and the hand position by randomly changing the joint angles (procedure 1). The first group of observation variables was built as $[y_1, y_2]^T \in \mathcal{Y}^1$. When the hand contacted object 1, the second group of observation variable was constructed as $y_3 \in \mathcal{Y}^2$. While the hand keeps contact with the object 1, the object moved together with the hand. When moving object 1 contacted with object 2, object 2 began to move and the third group was constructed as $y_4 \in \mathcal{Y}^3$. While collecting mapping data by the random motion, the robot also collected boundary information between mode 1 and mode 2. In this case discrimination function $F_{1,2}(\boldsymbol{x})$ of SVM was defined in three-dimensional space, that is, $\boldsymbol{x} = [y_1, y_2, y_3]^T$. Figure 16a and b show boundaries of two modes where $0.55 \leq q_1 < 0.65$ for (a) and $0.65 \leq q_1 < 0.75$ for (b). Circles in the figure denote variables in mode 1, crosses denote mode 2 and curves denote boundaries of $F_{1,2}(\boldsymbol{x}) = 0$.

[14] They correspond respectively to the movement of the robot hand and the movement of the rectangular object by contacting with the robot hand. Mode 3 corresponds to the movement of object 2 caused by the contact with object 1.

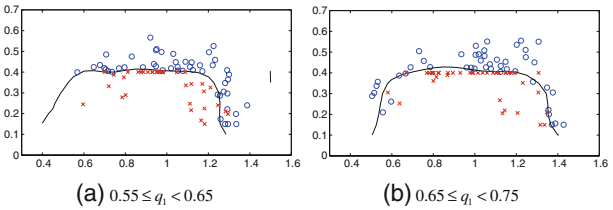


Fig. 16 Estimated boundaries between mode 1 and 2

The discrimination function $F_{1,2}(\boldsymbol{x})$ was utilized for parameterization on the boundary explained in Section 3.5. Figure 17a and b are resultant node distributions with $L = 9$ after 100 iterations of node movements, where nodes were initially located around the center of the boundary. It can be seen that nodes are distributed along the contour of $F_{1,2}(\boldsymbol{x}) = 0$. Both end nodes are constrained to the nearest observed data, which can be understood by comparing Fig. 16 with Fig. 17.

Figure 18 shows the trajectory of task 1, where a cross in the figure denotes the target configuration. This motion was realized by using the mapping between $\mathcal{Y}^1$ and $\mathcal{Y}^2$, which can be regarded as inverse kinematics learning. In the case of task 2, first reachable region of y_3 was checked by Algorithm 2. A cross in Fig. 19 denotes the destination on the boundary between mode 1 and 2. First the robot hand aimed at the destination on the boundary, and then kept contact to realize the desired configuration of y_3. In the case of task 3, first reachable region of y_4 was checked. The lower cross in Fig. 20 indicates the destination on boundary between mode 2 and 3, while the upper cross indicates the destination on boundary between mode 1 and 2. After realizing contact between the hand and object 1, the robot kept contact to move object 1 to the right direction and to realize contact between object 1 and object 2. It can be seen that finally the robot realized the desired configuration of $y_4 = 1.9$ by extending its body by using object 1.

4.2 Case 2: Pushing Manipulation on a Plane

The manipulator used for this setting is the same as case 1. In this case, it is assumed that only one object exist (, the simplest case) but it can move in the 2D space by contacting with the robot hand (Fig. 21). The kinematics of the object motion is approximately given as follows: The position of the hand is updated every discrete time step based on the updated values of the joint angles in the simulation. When interference between the hand and the circular object occurs, the position of the

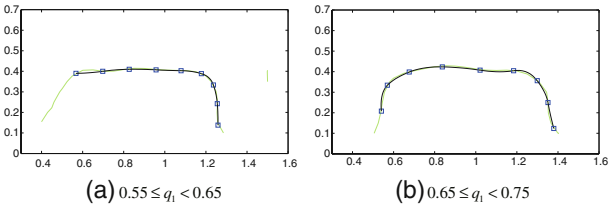
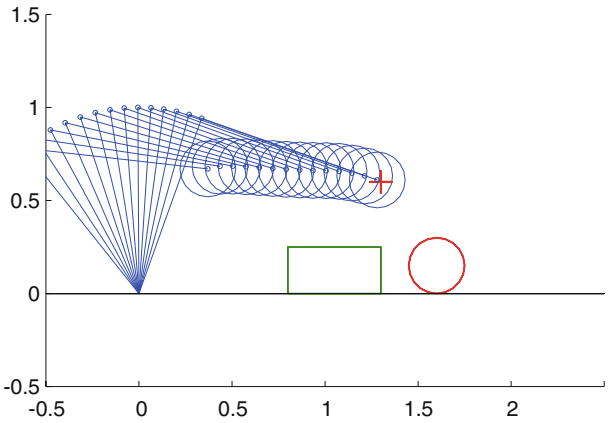


Fig. 17 Parameterization on boundary with nodes

Fig. 18 Trajectory of task 1

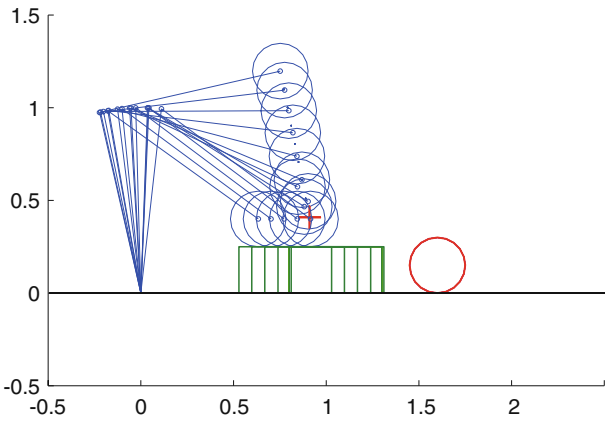


object is modified so as to resolve the interference with minimal displacement from the original position.

Since there is only one object, there are only mode 1 and 2, where mode 2 denotes that the object is moving with the hand. Exploration and learning are done by the followings:

1. Move randomly for 3,000 steps (within mode 1), including the initializations per 100 steps.
2. Build mapping between $[\theta_1, \theta_2]^T$ and $[y_1, y_2]^T$, estimate boundary between mode 1 and mode 2, and generate parameterization on the boundary between modes 1 and 2.
3. Choose a target subgoal on the boundary randomly and navigate the hand toward the subgoal in mode 1. When the hand contacts with the object, explore mode 2 randomly while trying to keep mode 2. Totally 1,000 steps are performed for this reachable region exploration while the positions of the hand and the object is reset every 50 steps.

Fig. 19 Trajectory of task 2



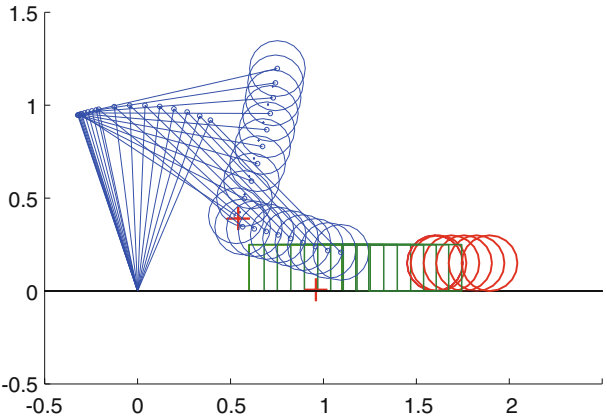


Fig. 20 Trajectory of task 3

In the estimation of the reachable region, approximation with parameterized function described in Section 3.6.2 is applied. A two-dimensional ellipsoid with parameter set of $\boldsymbol{\phi} = [\lambda_1, \lambda_2, \psi, \boldsymbol{d}^T]^T \in \mathbb{R}^5$ is used for the reachable region approximation:

$$\mathcal{F}_{\boldsymbol{\phi}}(\boldsymbol{y}) = (\boldsymbol{y} - \boldsymbol{d})^T R(\psi)^T \begin{bmatrix} \lambda_1^2 & \\ & \lambda_2^2 \end{bmatrix} R(\psi)(\boldsymbol{y} - \boldsymbol{d}) - 1, \quad \lambda_1, \lambda_2 > 0, \tag{4.1}$$

where $R(\psi) \in \mathbb{R}^{2\times 2}$ denotes the rotation matrix with rotation angle ψ. Using this parameter set, the square measure of the ellipsoid can be expressed by

$$J(\boldsymbol{\phi}) = \frac{1}{\lambda_1 \lambda_2}. \tag{4.2}$$

There are two tasks given to the robot:

- Task 1: Move the end effector to $[y_1, y_2]^T = [1.4, 0.4]^T$.
- Task 2: Move object 1 to $[q_1, q_2]^T = [0.86, 0.04]^T$.

Grouping of hand-related observation variables $[y_1, y_2]^T \in \mathcal{Y}^1$ and learning of kinematics of the manipulator are the same as the first setting. After that, the second

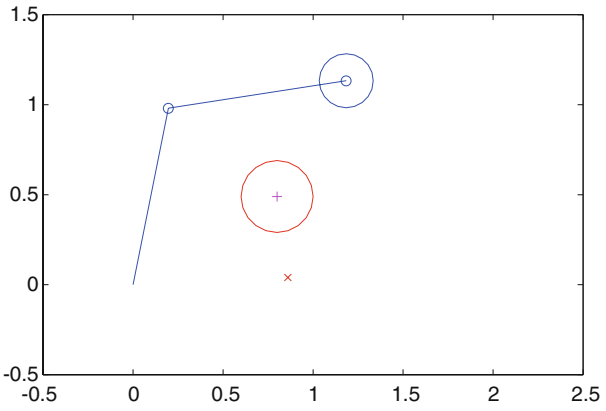


Fig. 21 Problem setting with one object

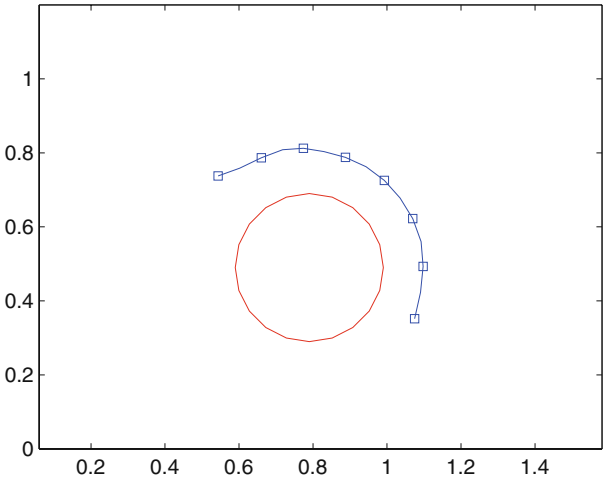


Fig. 22 Parameterized boundary

group of observation variables are grouped as $[y_3, y_4]^T \in \mathcal{Y}^2$ when they change by contact with the object. Thus, modes are created as mode 1 for free motion of the manipulator and mode 2 for motion of the object by the contact.

The parameterized boundary between mode 1 and 2 is shown in Fig. 22. 8 nodes are applied for the parameterization. Estimation of reachable regions is done for each discretized parameter on the boundary. In Fig. 23, three estimated ellipsoids with discretized parameters of $b = 2, 3, 4$ are shown. It can be seen that each ellipsoid contains all of the observed samples while keeping the size of the ellipsoid small. The optimization of Eq. 3.40 was solved with Nelder-Mead method [22]. Figures 24 and 25 show the result of interpolation of ellipsoidal parameters obtained by the least-square approximation of Eq. 3.42 from different views. Cross in Fig. 24 indicates the target position of the object.

Figure 26 shows the trajectory for task 1. In this case, initial configuration of the arm is given as $\boldsymbol{\theta} = [\frac{5}{7}\pi, -\frac{5}{8}\pi]^T$. The hand moved toward the target position while avoiding contact with the object. This was achieved by the mechanism of mode keeping, where contact with the object causes the mode transition from mode 1 to mode 2 and keeping mode 1 corresponds to avoiding contact with the object.

Figure 27 shows the trajectory for task 2, where the initial configuration of the arm is given as $\boldsymbol{\theta} = [\frac{3}{7}\pi, -\frac{7}{16}\pi]^T$. First the robot checked if the target position of

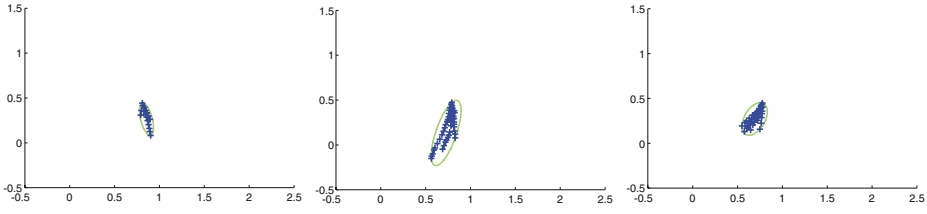


Fig. 23 Ellipsoids for reachable region estimation

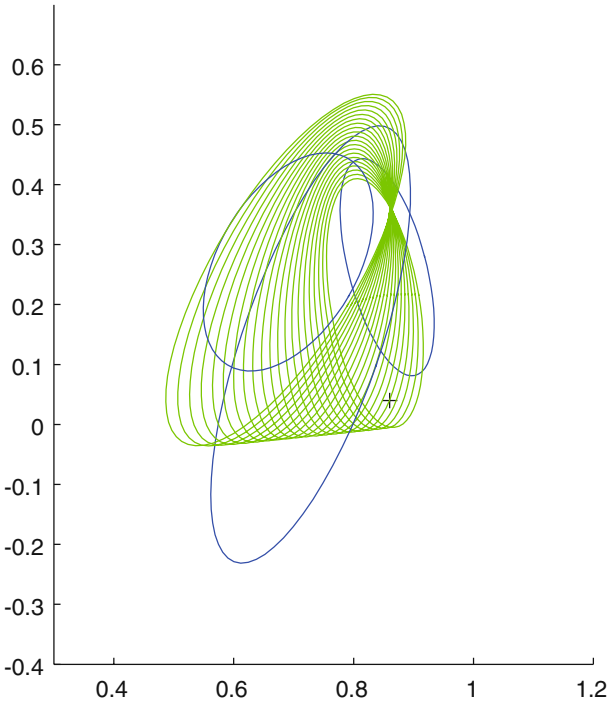


Fig. 24 Ellipsoid parameters interpolation (1)

the object is included in the approximated reachable region in mode 2 (step 4′ in Algorithm 2). The target (indicated as a cross in the figure) is not included in any of the approximated ellipsoids. Interpolation proposed in Section 3.6.2, however, enabled to recognize that the target position is included in the approximated reachable region with parameter $s^\dagger = 0.5$. The destination on the boundary that corresponds to $\boldsymbol{z}_{1,2}(s^\dagger)$ is indicated as a '+' in the figure. First the hand moved toward the destination

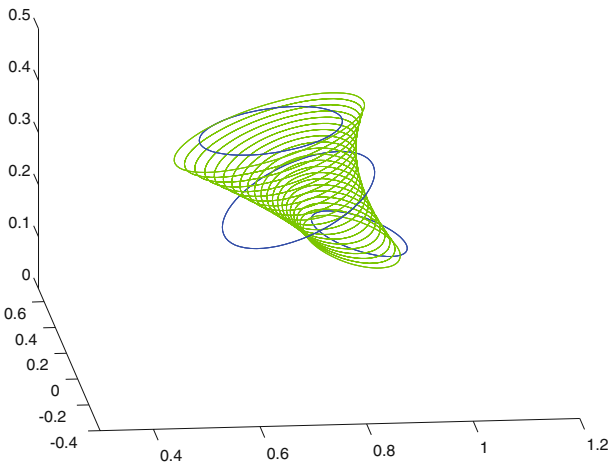


Fig. 25 Ellipsoid parameters interpolation (2)

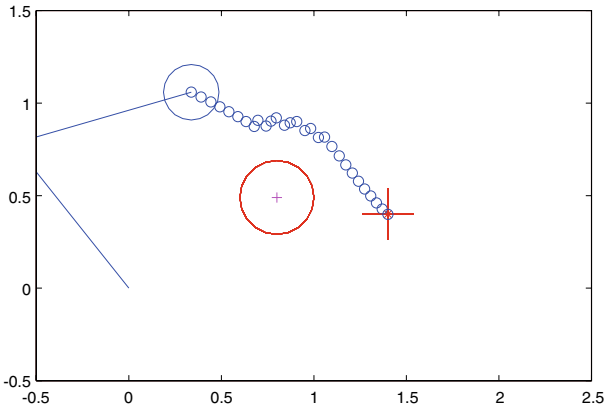


Fig. 26 Trajectory of task 1: reaching of the hand while avoiding contact with the object

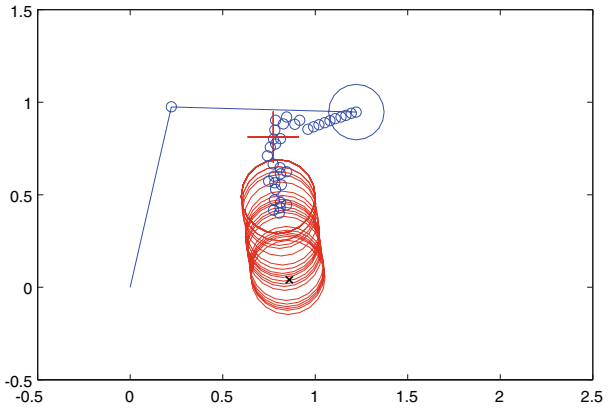


Fig. 27 Trajectory of task 2: pushing of the object toward the target position

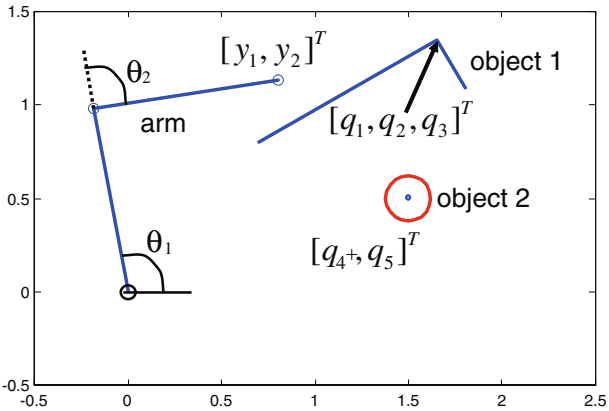


Fig. 28 Simulation setting with L-shaped and circular objects

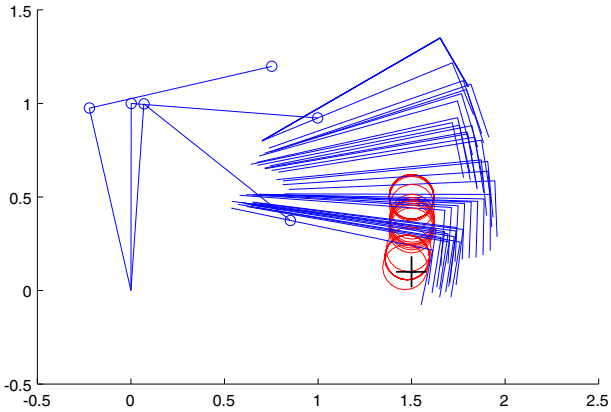


Fig. 29 Trajectory for task 1

on the boundary while avoiding to contact the object at different point from $\boldsymbol{z}_{1,2}(s^{\dagger})$, by the effect of Eq. 3.44. After mode transition from 1 to 2, the hand pushed the object toward the target position using the redundant mapping between $[y_1, y_2]^T$ and $[y_3, y_4]^T$ with Eq. 3.14. It can be seen that the realized trajectory of the robot is smooth, while the trajectory of the hand is zigzag.

4.3 Case 3: Using L-shaped Tool for Manipulation

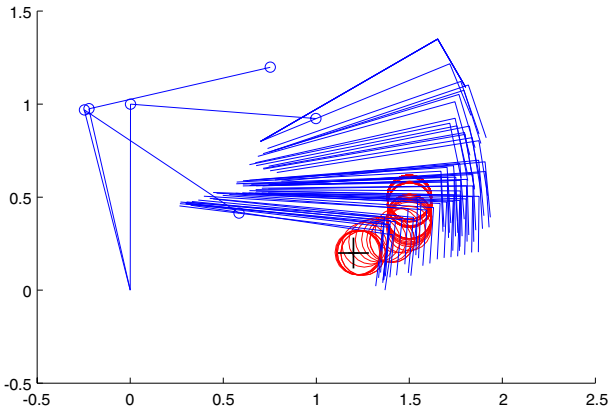
In the third problem setting, the link length of the manipulator is the same as case 1 and 2, whereas the circular end effector is omitted (Fig. 28). There are two objects, L-shaped object (object 1) and a circular object (object 2). When the end of the manipulator contacts with the L-shaped object, the hand sticks to the object and they move together thereafter. The relative posture of the object is fixed when the hand contacts with the object.

There are three modes in this problem. In mode 2, L-shaped object is moving with the hand. In mode 3, the circular object is moving by the contact with the L-shaped object. Exploration and learning are done by the followings:

1. Move randomly for 3,000 steps (within mode 1), including the initializations per 100 steps.
2. Build mapping between $[\theta_1, \theta_2]^T$ and $[y_1, y_2]^T$, estimate boundary between mode 1 and mode 2, and generate parameterization on the boundary between modes 1 and 2. h
3. Choose a target subgoal on the boundary between mode 1 and 2. Go to the boundary and then explore mode 2. Totally 3,000 steps are spent for this exploration including initializations per 100 steps.[15]
4. Build mapping between $[y_1, y_2]^T$ and $[y_3, y_4, y_5]^T$, estimate boundary between mode 2 and mode 3, and generate parameterization on the boundary between modes 1 and 2.

[15]For mode 2, explicit mode keeping controller is not required because once the hand contacts with the L-shaped object, they keep contact until the initialization.

Fig. 30 Trajectory for task 2



5. Choose target subgoals on the boundaries between mode 1 and 2 and between mode 2 and 3. The robot first aims the first subgoal (toward mode 2) and then the second subgoal (toward mode 3). After reaching mode 3, explore mode 3 randomly while trying to keep mode 3. Totally 6,000 steps are performed for this exploration while all of the configurations are initialized per 100 steps.
6. Build mapping between $[y_3, y_4, y_5]^T$ and $[y_6, y_7]^T$.

Since the initial positions of the L-shaped object and the circular object are fixed, parameterization of the mode boundaries are done only for the specific configuration decided by the initial configuration. Target subgoals on the mode boundaries were selected by calculating the mean of the total length of the boundary (for example, if a boundary is parameterized from 0 to 5, subgoal is decided by specifying the parameter as 2.5).

There are two tasks given to the robot:

– Task 1: Move object 2 to $[q_4, q_5]^T = [1.2, 0.2]^T$.
– Task 2: Move object 2 to $[q_4, q_5]^T = [1.5, 0.1]^T$.

Learning of kinematics of the manipulator are the same as the first and the second setting. After that, the second group of observation variables are grouped as $[y_3, y_4, y_5]^T \in \mathcal{Y}^3$ when they change by contact with the object.[16] Thus, modes are created as mode 1 for free motion of the manipulator and mode 2 for motion of the L-shaped object. Later, mode 3 is found for motion of the circular object. In the both cases in Figs. 29 and 30, the L-shaped object was utilized to move the circular object to the desired position.

[16]At the moment of contact, interference between the arm and the L-shaped object is resolved by rotating the object around the tip (closer to the arm). The configuration of the L-shaped object is specified by its position of its corner, thus three configuration variables changes at a time when contact occurs.

5 Discussion

In the simulation, it was shown that the proposed learning architecture enables various task execution (including pushing manipulation and obstacle avoidance) by finding relations among variables and using the relations for planning with various observation variables.

Motor babbling [7] has been discussed as human learning of motor coordination. While the first stage of mapping learning for the arm in the proposed method can be equivalent to inverse-kinematics learning (e.g. [23] for learning with neural networks and [8] for imitation learning for analyzing development of infants), the total framework deals with wider problem of finding relations among various observation variables. The proposal of this paper can be regarded as an construction method of body scheme [32], where 'self' is identified through the criterion of 'what can be controlled' and 'what cannot be controlled'. Speaking more concretely, the distinction between 'what can be controlled' and 'what cannot be controlled' is obtained through the grouping of the observation variables. The indicator sets of $I_C^{(i)}, i = 1, \cdots, j$ give the observation variables that can be controlled and the remaining indicators denote that those observation variables are impossible to control so far. The 'sense of agency' (e.g. [4]) is expressed as potentially controllable variables of $\boldsymbol{y}^{(1)}, \cdots, \boldsymbol{y}^{(k)}$ in this framework.

The proposed learning architecture includes a criterion for autonomous exploration (e.g.[24]), which is based on the structure of the task. Exploration in the proposed learning method is done while keeping a mode or aiming at different modes, which helps building an exploration strategy.

On the other hand, the applications presented in the simulation is rather simple and easy. The proposed architecture is applicable to more complex and high-dimensional problems, but to apply the framework to higher-dimensional problems, it is required to consider parameterization of high-dimensional sub-manifolds (boundaries between modes). Other possible extensions are the followings:

- The influence of noise should be taken into account in order to implement the proposed framework in the real world. One possible way to deal with noise in motion synchronousness detection is to approximate the effect of noise by preceding measurement. It would be more useful if online approximation of stochastic model of the motion and observation could be implemented.
- Observation variables directly correspond to configuration variables in this paper. Extraction of observation variables from various image feature vectors should be taken into consideration.
- The proposed framework does not include integration of multi-modal sensor information. E.g., tactile information is of great importance to generate the representation of the object [18]. Generation of modes can be done more efficiently with integration of other sensor information.
- A control strategy that utilizes a partial information of state variable is known as 'backstepping' in nonlinear control research fields [12].
- Normally we human 'grasp' an object when we intend to use the object as a tool. To fix the relative positions between the hand and the object, mechanical structures such as fingers or grips have to be implemented. We avoided such mechanisms in the robot for simplicity, but grasping behavior of the robot hand should be discussed in the future.

– For the discussion of motor development of human body, control of force through musculo-skeletal system is an important issue [13]. The idea of planning-space shift might be applied also to the different levels of motor control.
– From the viewpoint of widening applicability of the proposed framework, development of generalization ability will be one of the most important issues. The simulation results were realized based on many off-line trials. It will be preferable to decrease the number of trials when the robot learns to manipulate different objects based on the obtained knowledge. The interpolation of the estimated reachable region based on the parameterization of the mode boundary is an attempt to decrease the number of trials by effectively utilizing existent sampled data.

6 Conclusion

In this paper, an approach to hierarchy generation is proposed as planning-space shift learning. The learning framework consists of generation of modes based on motion synchronousness, estimation of boundaries between modes, and planning via multiple modes with estimated reachable regions. In simulation, three examples were shown to evaluate the proposed motion generation scheme. In the first example, an object was utilized as a tool to move another object in a simple one-dimensional motion. In the second example, an obstacle avoidance behavior and a pushing manipulation, normally realized by different control schemes with different state variables, were realized with the unified control architecture of hierarchical learning. In the third example, an L-shaped object was utilized as a tool to move a circular object to a destination with planar motion. In all of these examples, different hierarchies (different dependencies of groups of observation variables) were realized by the proposed framework. The proposed scheme is expected to be a fundamental building block for generation of complex motions of robots. In the future work, we consider executing experiments using real data. For this the influence of noise should be taken into account in order to implement the proposed framework in the real world.

The proposed architecture does not utilize any explicit representation of 'tool' and it uses just the information of motion synchronousness. An advantage of this approach is that the designer does not have to model tools, while in other researches 'objects' and 'tools' must be fixed and modeled in advance. For example, the robot may use an object as a tool that has not been supposed by the designer. On the other hand, this approach requires more trial of the robot to find objects and their motional properties. To extend and improve the proposed architecture, effective exploration and effective approximation of the motional properties will play an important role.

References

1. Asada, M., MacDorman, K., Ishiguro, H., Kuniyoshi, Y.: Cognitive developmental robotics as a new paradigm for the design of humanoid robots. Robot. Auton. Syst. **37**, 185–193 (2001)
2. Atkeson, C.G., Moore, A.W., Schall, S.: Locally weighted learning. Artif. Intell. Rev. **11**, 11–73 (1997)

3. Bemporad, A., Morari, M., Dua, V., Pistikopoulos, E.: The explicit linear quadratic regulator for constrained systems. Automatica **38**(1), 3–20 (2002)
4. Blakemore, S.J., Wolpert, D.M., Frith, C.D.: Abnormalities in the awareness of action. Trends Cogn. Sci. **6**(6), 237–242 (2002)
5. Brooks, R.A.: A robust layered control system for a mobile robot. IEEE J. Robot. Autom. **RA-2**, 253–262 (1986)
6. Dayan, P., Hinton, G.E.: Feudal reinforcement learning. In: Giles, C.L., Hanson, S.J., Cowan, J.D. (eds.) Advances in Neural Information Processing Systems, vol. 5, pp. 271–278 (1993)
7. Demiris, Y., Dearden, A.: From motor babbling to hierarchical learning by imitation: a robot developmental pathway. In: Berthouse, L., et al. (eds.) Proc. of the Fifth Int. Workshop on Epigenetic Robotics, pp. 31–37. Japan (2005)
8. Demiris, Y., Meltzoff, A.: The robot in the crib: a developmental analysis of imitation skills in infants and robots. Infant Child Dev. **17**, 43–53 (2008)
9. Drescher, G.L.: Made-Up Minds: A Constructivist Approach to Artificial Intelligence. MIT (1991)
10. Gibson, J.J.: The theory of affordances. In: Perceiving, Acting, and Knowing: Toward Ecological Psychology, pp. 62–82. Lawrence Erlbaum, NJ (1977)
11. Hauskrecht, M., Meuleau, N., Boutilier, C., Kaelbling, L.P., Dean, T.: Hierarchical solution of markov decision processes using macro-actions. In: Proc. of 14th Annual Conference on Uncertainty in Artificial Intelligence, pp. 220–222 (1998)
12. Khalil, H.K. (ed): Nonlinear Systems. Prentice Hall (2001)
13. Kuniyoshi, Y., Sangawa, S.: Early motor development from partially ordered neural-body dynamics: experiments with a cortico-spinal musculo-skeletal model. Biol. Cybern. **95**, 589–605 (2006)
14. Lavalle, S.M.: Planning Algorithms. Cambridge University Press (2006)
15. Lungarella, M., Metta, G., Pfeifer, R., Sandini, G.: Developmental robotics: a survey. Connect. Sci. **15**(4), 151–190 (2003)
16. Mangasarian, O.L., Musicant, D.R.: Lagrangian support vector machines. J. Mach. Learn. Res. **1**, 161–177 (2001)
17. Maravita, A., Iriki, A.: Tools for the body (schema). Trends Cogn. Sci. **8**(2), 79–86 (2004)
18. Metta, G., Fitzpatrick, P.: Early integration of vision and manipulation. Adapt. Behav. **11**(2), 109–128 (2003)
19. Michael Kass, A.W., Terzopoulos, D.: Snakes: active contour models. Int. J. Comput. Vis. **1**(4), 321–331 (1988)
20. Miyamoto, H., Morimoto, J., Doya, K., Kawato, M.: Reinforcement learning with via-point representation. Neural Netw. **17**, 299–305 (2004)
21. Nabeshima, C., Kuniyoshi, Y., Lungarella, M.: Adaptive body schema for robotic tool use. Adv. Robot. **20**(10), 1105–1126 (2006)
22. Nelder, J.A., Mead, R.: A simplex method for function minimization. Comput. J. **7**, 308–313 (1965)
23. Oyama, E., Agah, A., MacDorman, K.F., Maeda, T., Tachi, S.: A modular neural network architecture for inverse kinematics model learning. Neurocomputing **38–40**, 797–805 (2001)
24. Ratitch B., Precup D.: Using MDP Characteristics to Guide Exploration in Reinforcement Learning. Springer, Berlin (2003)
25. van der Schaft, A., Schumacher, H.: An Introduction to Hybrid Dynamical Systems. Springer (2000)
26. Siméon, T., Laumond, J.P., Cortés, J., Sahbani, A.: Manipulation planning with probabilistic roadmaps. Int. J. Rob. Res. **23**(7–8), 729–746 (2004)
27. Stoytchev, A.: Behavior-grounded representation of tool affordances. In: Proc. of IEEE Int. Conf. on Robotics and Automation, pp. 3071–3076. Barcelona, Spain (2005)
28. Stoytchev, A.: Toward video-guided robot behaviors. In: Proc. of the Seventh International Conference on Epigenetic Robotics, pp. 165–172. Rutgers University, NJ (2007)
29. Vapnik, V.N.: The Nature of Statictical Learning Theory. Springer (1995)
30. Wolpert, D.M., Kawato, M.: Multiple paired forward and inverse models for motor control. Neural Netw. **11**, 1317–1329 (1998)
31. Xu, C., Prince, J.L.: Snakes, shapes, and gradient vector flow. IEEE Trans. Image Process. **7**(3), 359–369 (1998)
32. Yoshikawa, Y., Hosoda, K., Asada, M.: Does the invariance in multi-modalities represent the body scheme?—A case study with vision and proprioception—. In: Proc. of the 2nd Int. Symp. on Adaptive Motion of Animals and Machines, vol. SaP-II-1 (2003)