Learning Object-Manipulation Verbs for Human-Robot Communication

Komei Sugiura
[1]Kyoto University
[2]ATR Spoken Language Communication Research Laboratories
2-2-2 Hikaridai, Seika, Soraku
Kyoto 619-0288, Japan
komei.sugiura@atr.jp

Naoto Iwahashi
[1]National Institute of Information and Communications Technology
[2]ATR Spoken Language Communication Research Laboratories
2-2-2 Hikaridai, Seika, Soraku
Kyoto 619-0288, Japan
naoto.iwahashi@atr.jp

ABSTRACT
This paper proposes a machine learning method for mapping object-manipulation verbs with sensory inputs and motor outputs that are grounded in the real world. The method learns motion concepts demonstrated by a user and generates a sequence of motions, using reference-point-dependent probability models. Four components, needed to learn object-manipulation verbs, are estimated from camera images; (1) a trajector and landmark, which are the objects of transitive verbs; (2) a reference point; (3) an intrinsic coordinate system; and (4) parameters of the motion's probabilistic model. The motion concepts are learned using hidden Markov models (HMMs). In the motion generation phase, our method then combines HMMs to generate trajectories to accomplish goal-oriented tasks. Results from simulation experiments in which our method generates motion by combining learned motion primitives are shown.

Categories and Subject Descriptors
I.2.6 [**Artificial Intelligence**]: Learning—*Language Acquisition, Concept Learning*; I.2.9 [**Artificial Intelligence**]: Robotics—*Operator interfaces*

General Terms
Algorithms, Languages

Keywords
human-robot interaction, motion generation, HMM

1. INTRODUCTION
To communicate with humans through natural language in a typical environment, machines need to use mechanism mapping symbols with related information in both linguistic and non-linguistic contexts. While it is sometimes believed that such mappings are fixed within a certain natural language system, however the referent of a symbol may differ from person to person. Let us take an example of conversation at home, "Put the dishes in the cupboard". To make a machine accomplish this task, first the recognition of "the cupboard", which is specific to the home, must be successful. The desired motion also depends on the size and shape of the dishes, as well as those of the cupboard and whether it has a door. Therefore, adaptation to both the user and the environment is necessary. Studies have begun to focus on machine learning methods that map symbols to motion and/or objects[7, 4].

The main advantage of mapping symbols to motion is as follows: The machine can break down the given task into learned motion primitives and then present the planned motions, in a way easy for the user to understand. This is a significant safety feature because if the machine can inform the user of the planned motions before executing them, the user can decide in advance whether they are safe.

In this paper, we propose a method that learns verbs and generates motion. With our method, probabilistic models are learned from visual inputs, and motion is generated using the learned probabilistic models. Figure 1 shows an example of the method's application. The robot has multimodal interfaces such as a camera and a microphone.

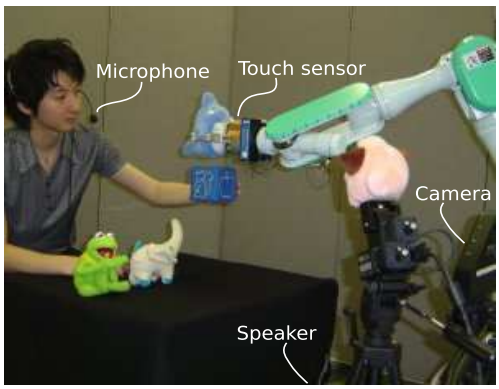

Figure 1: Human-robot interaction

The rest of this paper is organized as follows. Section 2 first

states the problem we try to solve, briefly reviews related work, and introduces out method. It then describes the motion learning/generation method in detail. Section 3 shows the results of simulation experiments in which our method generated motion by combining learned motion primitives. Section 4 discusses problems and possible applications with our method. Then, Section 5 concludes the paper.

2. METHOD

2.1 Reference-Point-Dependent Motion Primitives

How can we teach motion concepts to a robot? To make the question clear, we consider a situation in which the user teaches a verb by uttering it and showing the motion to a robot equipped with multimodal input devices (camera, microphone, etc). The question, therefore, is how the robot can learn and generate motion, and also map it to speech.

Clustering manipulation trajectories and mapping them to a verb are not sufficient for verb learning if the trajectories are considered only in the camera coordinate system. For simplicity, we assume that the mapping between the camera coordinate system and the world coordinate system is given, and that the user's utterance is accurately recognized. Let us take the example shown in Figure 2. The figure depicts a camera image, in which the user is placing the green puppet on the box. In the figure, the dotted line represents the trajectory. The trajectory itself is meaningless, though, if the position of the box has changed.

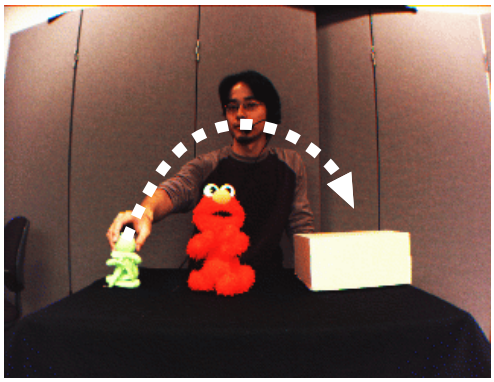

Figure 2: An example of camera images. The user is manipulating the green puppet. The dotted line represents the trajectory of the manipulation.

On the other hand, if we consider a coordinate system with its center at the box, we are able to cluster the trajectories in the coordinate system and map them to a verb. We call such a coordinate system an *intrinsic coordinate system*. The origin of an intrinsic coordinate system is called the reference point. Note that the origin of the intrinsic coordinate system changes as the position of the box changes.

Regier investigated a model describing the spatial relationship between two objects[6]. He proposed to model verbs as the time evolution of the spatial relationship between a trajector and a landmark. Here, a trajector is defined as a participant (object) that is focused on. A landmark has a secondary focus and a trajector is characterized with respect to a landmark. In cognitive linguistics, words representing spatial relationships such as "away" and "left of" are described as the relationship between a trajector and a landmark[5]. A method based on recurrent neural networks is proposed in [8], but this method is ineffective when the position of objects has changed. Inamura *et al.* developed a motion learning/generation method based on hidden Markov models (HMMs)[3]. However, their method deals with only intransitive verbs, such as "run" and "walk", and is thus not applicable to object-manipulation verbs.

In contrast, our method estimates four components, which are necessary for learning object-manipulation verbs, from camera images. The components are (1) the trajector and landmark, (2) the reference point, (3) the intrinsic coordinate system, and (4) the parameters of the motion's probabilistic model. Let us consider two examples "raise" and "move-closer" (Figure 3). We can reasonably assume that the reference point of "raise" is the trajector's center of mass. The intrinsic coordinate system can be a Cartesian coordinate system, as shown in the left-hand figure. In the case of "move-closer", another type of intrinsic coordinate system is necessary. In this case, the x-axis of the coordinate system passes through the centers of mass of the trajector and the landmark. As explained in the following subsection, we think there are several types of intrinsic coordinate systems.

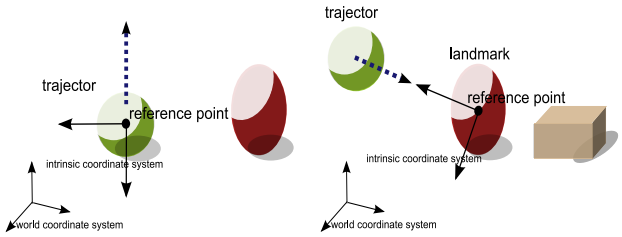

Figure 3: Relationship between trajector/landmark, a reference point, and an intrinsic coordinate system. The spheres, ellipsoids, and box represent objects, and the arrows represent the axes of the intrinsic coordinate systems. Left: "raise". The small sphere is the trajector, and the reference point is its center. The x-axis of the intrinsic coordinate system is horizontal. Right: "move-closer". The direction of the x-axis is toward the trajector from the landmark.

2.2 Motion Learning by Reference-Point-Dependent Probabilistic Models

Consider that N kinds of learning data are given for a verb. Let $\mathcal{V}_i$ denote the ith learning data. $\mathcal{V}_i$ consists of the motion information of the trajector, $\mathcal{Y}_i$, and the set of positions of static objects, $\boldsymbol{O}_i$, as follows:

$$\mathcal{V}_i = (\mathcal{Y}_i, \boldsymbol{O}_i), \tag{1}$$

$$\mathcal{Y}_i = \{\boldsymbol{y}_i(0), \boldsymbol{y}_i(1), ..., \boldsymbol{y}_i(T_i)\}, \quad \boldsymbol{y}_i(t) = \begin{bmatrix} \boldsymbol{x}_i(t) \\ \dot{\boldsymbol{x}}_i(t) \\ \ddot{\boldsymbol{x}}_i(t) \end{bmatrix}, \tag{2}$$

where $\mathcal{Y}_i$ is a time series of vectors $\boldsymbol{y_i}(t)$. Here, $\boldsymbol{y_i}(t)$ is composed of position $\boldsymbol{x}_i(t)$, velocity $\dot{\boldsymbol{x}}_i(t)$, and acceleration

$\ddot{\boldsymbol{x}}_i(t)$. For simplicity, we omit index i when the context is clear.

We assume that there are several types of intrinsic coordinate system, and these are given by the designer. We denote the type of coordinate system by C. From the estimation of C and the reference point $\boldsymbol{r}$, we obtain the intrinsic coordinate system in the ith data. The candidate set of reference points is denoted by $\boldsymbol{R}$. The set of positions of static objects, $\boldsymbol{O}$, has to be included in $\boldsymbol{R}$, since those objects are the candidates of the landmark. We also include the first position of the trajector, $\boldsymbol{x}(0)$, in $\boldsymbol{R}$ so that we can describe a motion concept which is dependent only on the object's trajectory. Additionaly, the center of the camera image, $\boldsymbol{x}_{\text{center}}$, is put in $\boldsymbol{R}$ to describe motion concepts that are independent on the positions of objects. Therefore,

$$\boldsymbol{R} = \{\boldsymbol{O}, \boldsymbol{x}(0), \boldsymbol{x}_{\text{center}}\} \triangleq \{\boldsymbol{r}_1, \boldsymbol{r}_2, ..., \boldsymbol{r}_M\}. \tag{3}$$

Let ${}^{C_k(m_i)}\mathcal{Y}_i$ denote the trajectory in the intrinsic coordinate system $C_k(m_i)$, where k and m_i are the type of the coordinate system and the index of the reference point in $C_k(m_i)$, respectively. Henceforth parameters in a coordinate system are written in a similar manner. Now, the optimal k, the optimal set of reference points, $\boldsymbol{m}$, and the optimal parameters of a probabilistic model, λ, are searched for using the following maximum likelihood criterion:

$$(\hat{\lambda}, \hat{k}, \hat{\boldsymbol{m}}) = \underset{\lambda,k,\boldsymbol{m}}{\operatorname{argmax}} \sum_{i=1}^{N} \log P(\mathcal{Y}_i | m_i, \boldsymbol{R}_i, k, \lambda), \tag{4}$$

$$= \underset{\lambda,k,\boldsymbol{m}}{\operatorname{argmax}} \sum_{i=1}^{N} \log P({}^{C_k(m_i)}\mathcal{Y}_i; \lambda), \tag{5}$$

where $\hat{\cdot}$ represents estimation.

The number of candidate solution of k is generally smaller than that of $\boldsymbol{m}$, which is $\prod_{i=1}^{N} m_i$. Therefore, we first fix k and obtain the following equation from Equation (5).

$$({}^{C_k}\hat{\lambda}, {}^{C_k}\hat{\boldsymbol{m}}) = \underset{\lambda,\boldsymbol{m}}{\operatorname{argmax}} \sum_{i=1}^{N} \log P({}^{C_k(m_i)}\mathcal{Y}_i | \lambda, k) \tag{6}$$

It is not practical to solve this equation because of its computational complexity. Therefore, for constraint relaxation, we approximate the above discrete optimization problem by a continuous optimization problem:

$$({}^{C_k}\hat{\lambda}, {}^{C_k}\hat{\boldsymbol{w}}) = \underset{\lambda,\boldsymbol{w}}{\operatorname{argmax}} \sum_{i=1}^{N} \log \left[\sum_{m=1}^{M_i} w_{i,m} P({}^{C_k(m_i)}\mathcal{Y}_i | \lambda, k) \right] \tag{7}$$

where

$$\sum_{m=1}^{M_i} w_{i,m} = 1 \qquad (i = 1, 2, \ldots, N),$$

$$\boldsymbol{w}_i = \{w_{i,1}, w_{i,2}, \ldots, w_{i,M_i}\}$$

$$\boldsymbol{w} = \{\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_N\}$$

Here, $w_{i,m}$ denotes the weight to select the reference point m in the learning data $\mathcal{V}_i$. Thus we can think of ${}^{C_k}m$ as hidden parameters. We can solve Equation (7) efficiently by applying an EM algorithm[1]. [2] gives the solution to Equation (7) for the case where an HMM is used as the probabilistic model. The optimal k is obtained by solving the following equation.

$$\hat{k} = \underset{k}{\operatorname{argmax}} \sum_{i=1}^{N} \log \left[\sum_{m=1}^{M_i} {}^{C_k}\hat{w}_{i,m} P({}^{C_k(m_i)}\mathcal{Y}_i | {}^{C_k}\hat{\lambda}) \right] \tag{8}$$

These learned HMMs are used for motion generation. Tokuda *et al.* has proposed a method that generates the maximum likelihood trajectory from HMMs[9].

2.3 Motion Generation by Combining Transformed HMMs

In motion learning, a verb is learned from multiple scene data. On the other hand, the scene is fixed in motion generation. Our proposed method deals with the following two types of motion generation:

(a) Target Instruction
The user requests that the robot move an object to a goal. Inputs from the user are the object ID and goal position x_G. The proposed method then outputs a sequence of verb-landmark pairs, $(\boldsymbol{v}, \boldsymbol{m}')$. The method also outputs a trajectory, $\mathcal{Y}$, to the goal from the initial position of the object. The method searches for both the optimal sequence of verb-landmark pairs and the optimal trajectory according to likelihood criteria. In other words, it decomposes this goal-oriented task into the verb-landmark sequence with the maximum likelihood. For example, if the user wants to put object A above object B, the estimated sequence would be composed of two verbs: "put-on A B" and "raise A".

(b) Explicit Instruction
The user requests the robot to move an object according to his instruction. The instruction consists of a sequence of motions. Inputs from the user are the object ID and a sequence of verb-landmark pairs. The proposed method estimates the optimal trajectory that accomplishes the sequence. The output trajectory is optimal in terms of likelihood.

In each case, the generated motion is composed of one or more motion primitives taught by the user. A novel method that combines the probabilistic models for generating a composite trajectory is described below.

The main point of our method is the transformation of an intrinsic coordinate system of reference-point-dependent HMMs. Although trajectory generation based on HMMs has been widely investigated in the field of speech synthesis, those HMMs do not have to be transformed since they share the same coordinate system (Figure 4). It is therefore possible to combine two HMMs by simply aligning them temporally. However, it is impossible to combine two reference-point-dependent HMMs in such a way. Figure 5 illustrates an example of combining two reference-point-dependent HMMs. To combine HMMs corresponding to "raise" and "move-closer", each of the HMM output probability distributions needs to be transformed since it represents a distribution on a coordinate system intrinsic to each motion primitive.

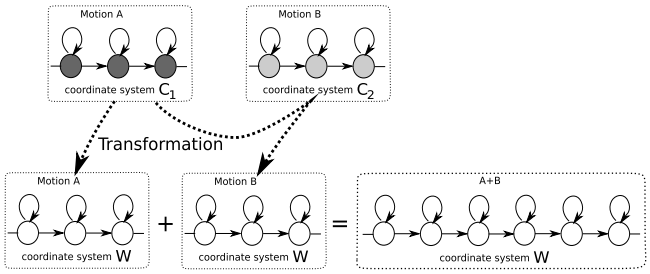

Figure 4: Schematic of the combination of two motion primitives.

An advantage of transforming intrinsic coordinate systems is the smoothness in composite trajectories. In our method, velocity and acceleration data are used for learning as well as position data. For safety reasons, changes in the velocity and acceleration data should be continuous. It is therefore important to obtain smooth trajectories of $\dot{\boldsymbol{x}}$ and $\ddot{\boldsymbol{x}}$ when combining two HMMs. Let us consider a case in which verbs dependent only on velocity information such as "throw" are to be combined. If two HMMs were simply aligned to generate the composite trajectory, the velocity changes might be discontinuous in this case. On the other hand, our method, which is described in detail below, generates a smooth trajectory.

Let Λ denote a sequence of HMM parameters that represents a sequence of verb-landmark pairs. Λ is a D-tuple HMM parameters that are transformed from learned HMM parameters:

$$\Lambda = ({}^{W}\lambda_1, {}^{W}\lambda_2, \ldots, {}^{W}\lambda_D) \tag{9}$$

$${}^{W}\lambda_j = F_{WC_j}(\lambda_j) \tag{10}$$

where F_{WC_j} denotes a transformation from the jth intrinsic coordinate system C_j to the world coordinate system, and C_j is defined as $C_j \triangleq C_{k_j}(m_j)$. We use a single Gaussian for each dimension of the output probability distributions of the HMMs. The transformation F_{WC_j} is performed in two stages. First, the mean vectors in λ_j is transformed, and then the variance vectors is trandformed.

The mean and variance vectors at state s of the jth HMM are represented as

$$\boldsymbol{\mu_y}(j,s) = \begin{bmatrix} \boldsymbol{\mu_x}(j,s) \\ \boldsymbol{\mu_{\dot{x}}}(j,s) \\ \boldsymbol{\mu_{\ddot{x}}}(j,s) \end{bmatrix}, \qquad \boldsymbol{\sigma_y}^2(j,s) = \begin{bmatrix} \boldsymbol{\sigma_x}^2(j,s) \\ \boldsymbol{\sigma_{\dot{x}}}^2(j,s) \\ \boldsymbol{\sigma_{\ddot{x}}}^2(j,s) \end{bmatrix}, \tag{11}$$
$$(j = 1, 2, ..., D, \quad s = 1, 2, ..., S_j)$$

That is, the mean vector $\boldsymbol{\mu_y}(j,s)$ consists of mean position vector $\boldsymbol{\mu_x}(j,s)$, mean velocity vector $\boldsymbol{\mu_{\dot{x}}}(j,s)$, and mean acceleration vector $\boldsymbol{\mu_{\ddot{x}}}(j,s)$. Similarly, variation vector $\boldsymbol{\sigma_y}^2(j,s)$ consists of position, velocity, and acceleration vectors, $\boldsymbol{\sigma_x}^2(j,s)$, $\boldsymbol{\sigma_{\dot{x}}}^2(j,s)$ and $\boldsymbol{\sigma_{\ddot{x}}}^2(j,s)$.

The mean vector of the jth HMM is transformed by the following equation:

$$\begin{aligned} &{}^{W}\boldsymbol{\mu_y}(j,s) \\ &= G_{WC_j}\left(\begin{bmatrix} {}^{C_j}\boldsymbol{\mu_x}(j,s) - {}^{C_j}\boldsymbol{\mu_x}(j,1) \\ {}^{C_j}\boldsymbol{\mu_{\dot{x}}}(j,s) \\ {}^{C_j}\boldsymbol{\mu_{\ddot{x}}}(j,s) \end{bmatrix}\right) + \begin{bmatrix} {}^{W}\boldsymbol{\mu_x}(j-1, S_{j-1}) \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}, \end{aligned} \tag{12}$$
$$(j = 1, 2, ..., D)$$

$${}^{W}\boldsymbol{\mu_x}(0, S_0) = \boldsymbol{x}_S \tag{13}$$

Here, G_{WC_j} denotes an affine transformation from the intrinsic coordinate system of the jth motion primitive, C_j, to the world coordinate system, W. Also, $\boldsymbol{x}_S$ denotes the initial position of the trajector.

Next, the variance vectors are transformed:

$$\begin{aligned} &{}^{W}\boldsymbol{\sigma_y}^2(j,s) \\ &= \begin{cases} {}^{C_j}\boldsymbol{\sigma_y}^2(j,s) & \text{if } \boldsymbol{r}_j \in \boldsymbol{O}, \\ {}^{C_j}\boldsymbol{\sigma_y}^2(j,s) + \begin{bmatrix} {}^{W}\boldsymbol{\sigma_x}^2(j-1, S_{j-1}) \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix} & \text{otherwise} \end{cases} \end{aligned} \tag{14}$$
$$(j = 1, 2, ..., D)$$

where

$${}^{W}\boldsymbol{\sigma_x}^2(0, S_0) = \mathbf{0} \tag{15}$$

$${}^{W}\boldsymbol{\sigma_x}^2(1,1) = \mathbf{0} \tag{16}$$

Here, $\boldsymbol{O}$ denotes the set of static objects' positions. Equation (14) adds the position variance of the $(j-1)$th HMM to that of the $(j-1)$th HMM if the type of coordinate system intrinsic to the jth motion primitive does not depend on any landmark. Hence, our method combines two HMMs smoothly.

From Equations (12), (13), and (16), we obtain ${}^{W}\boldsymbol{\mu_x}(1,1) = \boldsymbol{x}_S$ and ${}^{W}\boldsymbol{\sigma_x}^2(1,1) = \mathbf{0}$; consequently, the composite trajectory starts exactly at $\boldsymbol{x}_S$ where its likelihood is at a maximum.

In the target instruction mode, all candidate motion sequences (sequences of verb-landmark pairs) are generated. Here, the number of combined HMMs, D, is a constant. D works as a search depth parameter. The maximum likelihood trajectory and motion sequence are then estimated under the constraint on start and goal positions.

$$(\hat{\mathcal{Y}}, \hat{\boldsymbol{v}}, \hat{\boldsymbol{m}}') = \underset{\mathcal{Y}, \boldsymbol{v}, \boldsymbol{m}'}{\operatorname{argmax}} \log P(\mathcal{Y} | \boldsymbol{x}_S, \boldsymbol{x}_G, \boldsymbol{v}, \boldsymbol{m}'), \tag{17}$$

where $\hat{\mathcal{Y}}$ denotes the estimated maximum likelihood trajectory, $\hat{\Lambda}$ denotes the estimated maximum likelihood motion sequence, and $\boldsymbol{m}'$ represents the sequence of m_j. The method explained in [9] gives the maximum likelihood trajectory from Λ. Therefore, we obtain the solution to Equation (17) by applying this method to all candidate sequences.

On the other hand, in the direct instruction mode, the user inputs a motion sequence, therefore only the maximum likelihood trajectory is searched for under the constraint on the

goal position.

$$\hat{\mathcal{Y}} = \underset{\mathcal{Y}}{\operatorname{argmax}} \log P(\mathcal{Y}|\boldsymbol{x}_S, \boldsymbol{v}, \boldsymbol{m}') \quad (18)$$

We obtain the solution by following [9] as well.

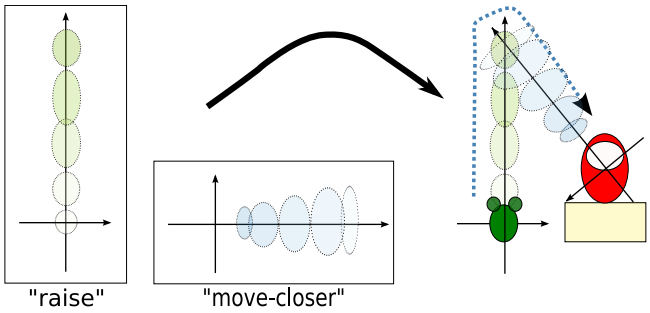

Figure 5: Example of transformation in the combination of two HMMs, "raise" and "move-closer". Each dotted circle represents the variation of output probability distributions at each state of a left-to-right HMM. The direction of state transition is indicated by the color darkness. The intrinsic coordinate system of "move-closer" is transformed so that the x-axis of it passes through both the landmark (the reference point of "move-closer") and the last position of the HMM regarding "raise". The dotted line represents the composite trajectory.

3. EXPERIMENTS

3.1 Experimental Settings

First, motion primitives to be used in the motion were prepared. All motion primitives were taught by the user in a learning phase beforehand, and they were fixed throughout the motion generation experiments. During the learning phase, the user taught verbs to the robot by uttering them and demonstrating their trajectories.

In the learning phase, the robot learned the following verbs.

raise, move-closer, move-away, rotate, place-on, put-down, jump-over

For each verb, the number of learning data was 15.

The verbs were successfully learned in the experiment. Figure 6 shows examples of the recognition results for the learning data. In Figure 6, the thick arrows represent the trajectories of an object manipulated by the user. The thin arrows show the x- and y-axes of the estimated type of intrinsic coordinate system. The type name is shown at the lower right of each illustration. The types of intrinsic coordinate system are defined as follows:

C_1 a coordinate system with its origin at the landmark position. C_1 is a transformed camera coordinate system. The x-axis is inverted in case the x-coordinate of the original position of the trajector is negative after transformation.

C_2 an orthogonal coordinate system with its origin at the landmark position. The direction of the x-axis is from the landmark towards the trajector.

C_3 a transformed camera coordinate system with its origin at the original position of the trajector.

C_4 a transformed camera coordinate system with its origin at the center of the image.

After motion primitives were prepared, we carried out motion generation simulation experiments. Two types of motion experiments were done: for target instruction and for explicit instruction.

In the target instruction experiment, the user requested that the robot move an object in a camera image. The object ID and a goal point were input, and then the robot output both a sequence of verb-landmark pairs and a trajectory to the goal. We used 64 grid points as goal points. The search depth parameter D was set as $D = 3$. Therefore, the estimated motion sequence was composed of up to three motion primitives.

3.2 Result (A): Target Instruction

The simulation environment is shown in Figure 7. There were five objects in the environment (depicted as numbered boxes and circles). Object 1 was used as the trajector. Figure 7 shows examples of maximum likelihood trajectories output by our method. In the figure, bracketed pairs represent the estimated sequences of verb-landmark pairs. For example, <place-on, 2><move-closer, 5> means "place object 1 on object 2, and move object 1 closer to object 5".

From Figure 7, we can see that our method combined two motion primitives smoothly rather than independently. Specifically, although <move-away, 2> and <move away, 2><jump-over, 4> share <move-away, 2>, the trajectories do not overlap each other. This is probably due to the large variation for the last position in the learned probabilistic model of "move-away". In other words, part of the trajectory for <move-away, 2> was curved to smoothly combine <move away, 2> and <jump-over, 4>, but the likelihood of the resultant trajectory was still high because of its large variance.

3.3 Result (B): Direct Instruction

The direct instruction experiment was carried out in the same simulation environment. Figure 8 shows two examples of motion generation: "jump object 1 over object 2, then put object 1 down, and move object 1 closer to object 4" and "jump object 2 over object 1, jump object 2 over object 1 again, and then place object 2 on object 5". The inputs for the two cases were as follows:

(a) trajector = object 1, motion sequence = <jump-over, 2><put-down, no landmark><move-closer, 4>

(b) trajector = object 2, motion sequence = <jump-over, 1><jump-over, 1> <place-on, 4>

Figure 8 shows that the three motion primitives were combined smoothly. To examine this result quantitatively, we

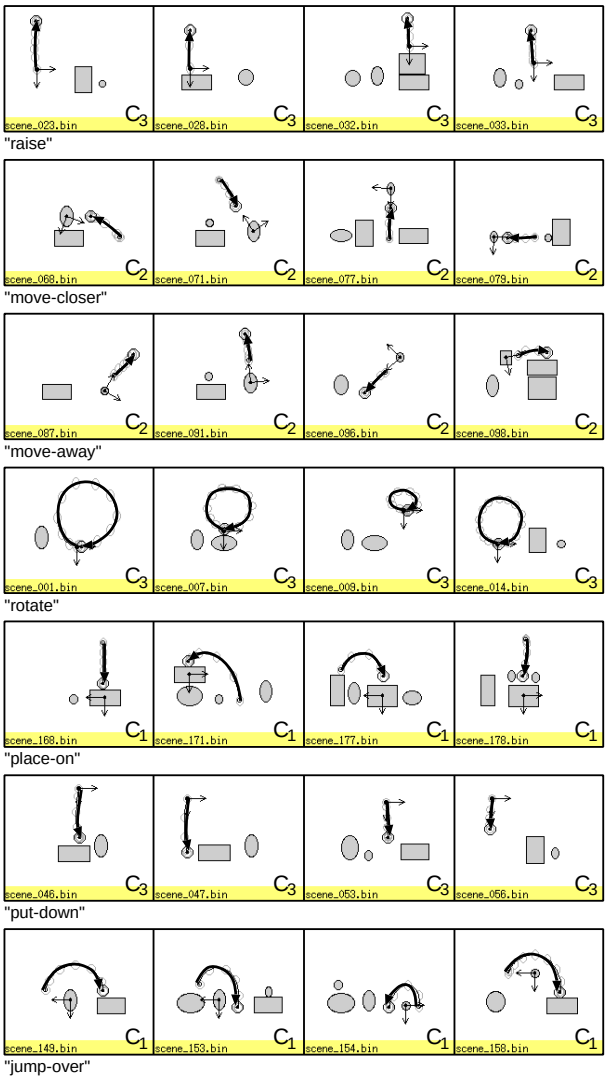

Figure 6: Examples of learning data.

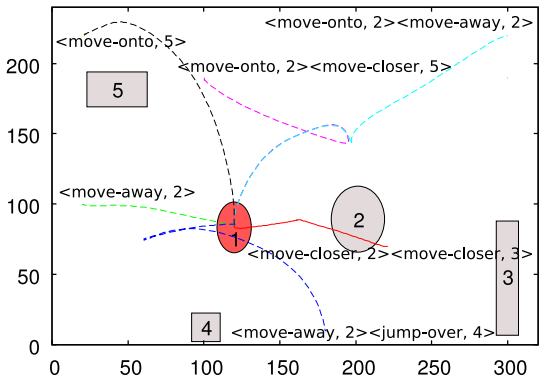

Figure 7: Examples of generated motion for manipulating object 1. Numbered boxes and circles represent objects. Bracketed pairs represent the estimated sequence of verb-landmark pairs. Specifically, <place-on, 2><move-closer, 5> means "place object 1 on object 2, and move object 1 closer to object 5".

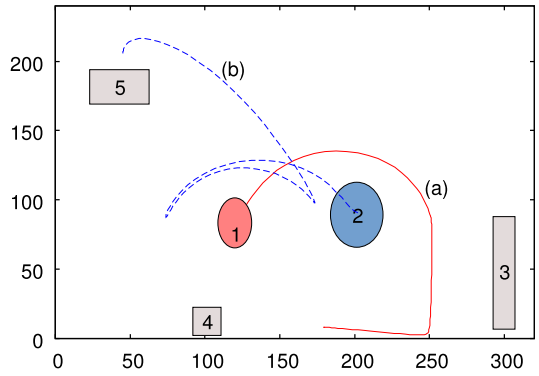

Figure 8: Direct Instruction

plotted the evolution of the position, velocity, and acceleration in case (a) in Figure 9. Figure 9 verifies that the composite trajectory of the velocity and that of the acceleration were continuous.

4. DISCUSSION

4.1 Problems

Our method has a possibility of generating inappropriate trajectories leading to object collision. For instance, Figure 7 shows that the trajectory of <move-closer, 2><move-closer, 3> runs through Object 2.

There are at least three solutions to this problem. The first is to select the maximum likelihood sequence of verb-landmark pairs among which no collision occur. This is the simplest solution since the positions of all static objects are evident in camera images. Another solution is to change the maximum trajectory slightly to avoid collisions. The third one is to modify the output probability density functions of HMMs by setting them to 0 near the position of obstacles. The last method, however, will not work if there are many obstacles in the camera image.

Another problem is that Equation (14) does not consider the full-covariance matrices, so the rotation of coordinate systems is ignored. The transformed variance is therefore thought to be an approximation. One reason these matrices are not considered is that no trajectory generation method has been formulated.

4.2 Possible Applications

The proposed method can be applied to motion planning other than manipulating objects. One example is a mobile robot that generates a path to a goal set by the user by combining learned motion primitives and tells them to the user. Suppose a camera is placed on the ceiling of an office and a camera image like Figure 7 is obtained. In this case, the robot can decompose an instruction like "go to the president's room" into learned motion primitives such as "move-forward" and "move-along".

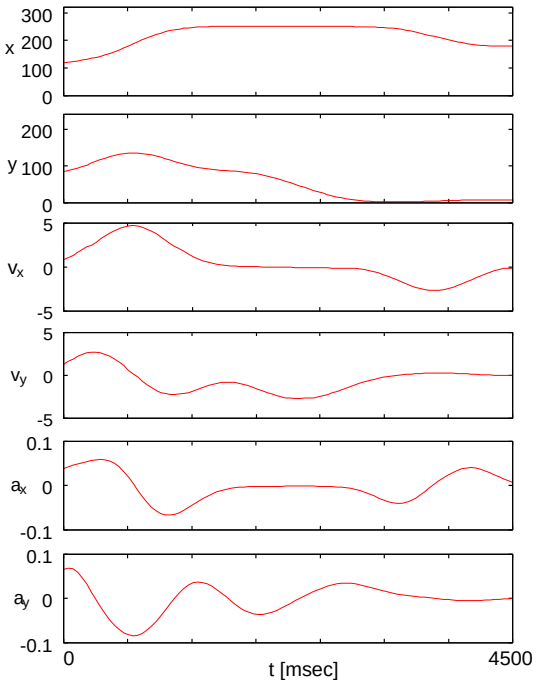

Figure 9: Evolution of position, velocity, acceleration under condition (a) in Figure 8

Another application would be a system to recognize the actions and movements of people within an office. Wren proposed an HMM-based method for understanding human behavior on a building-wide scale[10].

In the target instruction mode, the proposed method searches for trajectories that pass through both the start and goal positions. This is done by computing the likelihood for all combinations of motion primitives, where the search depth is a constant. Consider a detection problem in which the movements of people are recorded. The maximum likelihood verb-landmark sequence is searched for through the following equation:

$$(\hat{\boldsymbol{v}}, \hat{\boldsymbol{m}}') = \underset{\boldsymbol{v},\boldsymbol{m}'}{\operatorname{argmax}} \log P(\boldsymbol{v}, \boldsymbol{m}'|\mathcal{Y}) \tag{19}$$

Thus the user's movement can be described by a combination of learned motion primitives. Furthermore, if the estimation is done with part of the trajectory, the system can predict the landmark of the motion and help the user. For example, this could be used in a technique to unlock a door before the user arrives at it.

5. CONCLUSION

Within environments shared by humans and machines, it is important that machines be able to report their internal states to humans in a comprehensive way. For example, it is critical to the safety of people working around machines, that a robot functioning in the same area be able to tell what it will do next. In this paper, we have described a method that (1) learns motion primitives grounded in real world actions, and (2) plans a motion sequence to accomplish goal-oriented motion by combining learned motions.

Acknowledgments

This research was partially supported by the Ministry of Education, Science, Sports and Culture, Grant-in-Aid for JSPS Fellows, 18-2972, and by a research grant from the National Institute of Informatics.

6. REFERENCES

[1] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society Series B*, 39(1):1–38, 1977.

[2] T. Haoka and N. Iwahashi. Learning of the reference-point-dependent concepts on movement for language acquisition. In *PRMU2000-105*, pages 39–46, 2000.

[3] T. Inamura, I. Toshima, H. Tanie, and Y. Nakamura. Embodied symbol emergence based on mimesis theory. *International Journal of Robotics Research*, 23(4):363–377, 2004.

[4] N. Iwahashi. Robots that learn language: Developmental approach to human-machine conversations. In P. Vogt et al., editors, *Symbol Grounding and Beyond: Proceedings of the Third International Workshop on the Emergence and Evolution of Linguistic Communication*, pages 143–167. Springer, 2006.

[5] R. W. Langacker. *Foundations of Cognitive Grammar: Theoretical Prerequisites*. Stanford Univ Pr, 6 1987.

[6] T. Regier. *The Human Semantic Potential: Spatial Language and Constrained Connectionism*. Bradford Books, 9 1996.

[7] D. Roy. Grounding words in perception and action: computational insights. *Trends in Cognitive Science*, 9(8):389–396, 2005.

[8] Y. Sugita and J. Tani. Learning semantic combinatoriality from the interaction between linguistic and behavioral processes. *Adaptive Behavior*, 13(1):33–52, 2005.

[9] K. Tokuda, T. Kobayashi, and S. Imai. Speech parameter generation from HMM using dynamic features. In *Proceedings of International Conference on Acoustics, Speech, and Signal Processing*, pages 660–663, 1995.

[10] C. R. Wren. Large networks of ultra-low resolution sensors in buildings. In *Proceedings of International Conference on Integration of Knowledge Intensive Multi-Agent Systems*, pages 373–389, 2005.