Compact Models of Motor Primitive Variations for Predictable Reaching and Obstacle Avoidance

Freek Stulp[§,‡,¶], Erhan Oztop[†,‡], Peter Pastor[§], Michael Beetz[¶], Stefan Schaal[§,‡]

***Abstract*— In most activities of daily living, related tasks are encountered over and over again. This regularity allows humans and robots to reuse existing solutions for known recurring tasks. We expect that reusing a set of standard solutions to solve similar tasks will facilitate the design and on-line adaptation of the control systems of robots operating in human environments.**

In this paper, we derive a set of standard solutions for reaching behavior from human motion data. We also derive stereotypical reaching trajectories for variations of the task, in which obstacles are present. These stereotypical trajectories are then compactly represented with Dynamic Movement Primitives. On the humanoid robot Sarcos CB, this approach leads to reproducible, predictable, and human-like reaching motions.

I. Introduction

In almost all activities of daily living, related tasks are encountered over and over again. Therefore, humans "tend to solve similar or even identical instances over and over, so we can keep recycling old solutions with minor modifications" [1]. Motor primitives are an effective way of representing solutions to specific tasks, and have proven to be a successful approach for motor control in animals [2], [3], as well as in robots [4]. Reusing a set of standard solutions to solve similar tasks 1) drastically reduces the search space for control; 2) makes learning control in high dimensional movement systems feasible; 3) facilitates the design and on-line adaptation of the control systems of robots operating in human environments.

In this paper, we focus on the second part of the initial quote: "recycling old solutions with minor modifications" [1], and investigate methods for generalizing motor primitives, so that they not only provide solutions to a specific task, but also to variations of this task. We apply these ideas to a task in which the robot reaches for a target object, where some *standard variations* arise due to the presence of obstacles.

The humanoid robot used in this work learns the appropriate response to task variations from human demonstration, acquired with a motion tracking system. Instead of training the robot with just any reaching trajectory, we first gather human motion under different task conditions, and model it with Point Distribution Models and clustering techniques to acquire a set of stereotypical reaching trajectories for both the default and avoidance behaviors. We then train Dynamic Movement Primitives [4] for these stereotypical trajectories, which enables there execution on the humanoid robot 'Sarcos CB' [5]. This process is summarized in Fig. 1. The result is a small set of reaching motor primitives that deals with many different obstacle positions. We also compute a mapping from task parameters to appropriate reaching motion, to deal with the multi-modality inherent in the task.

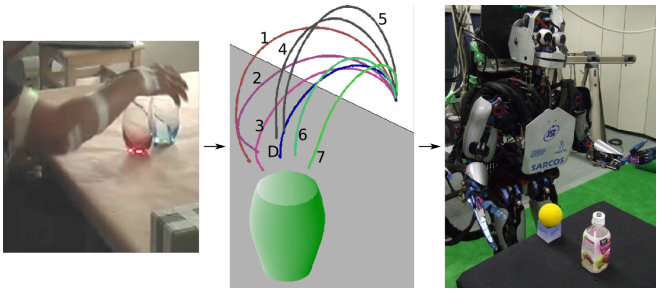


Fig. 1. Extracting avoidance strategies from human reaching motion data, and using them to control a humanoid robot.

Encoding motor primitive variations in this way leads to:

Compact representations. Directly mapping a task context to a motor primitive (variation) does not require an on-line search for the solution, and therefore places much less computational load on the system during task execution.

Predictable motion. The response to a static obstacle is not determined on-line during task execution, but is rather pre-planned, by encoding the obstacle in the desired trajectory. As the trajectory is pre-planned, it is reproduced in a similar way in similar situations. This reproduction causes the reaching behavior to be more predictable, which makes it easier to learn internal models for them. As the same motor primitive solution is reproduced over and over again for the same task, it can also be optimized, as humans do [6].

Legible motion. Predictable human-like behavior facilitates the interpretation of intentions for outside observers, i.e. the behavior becomes more legible. This will enable humans to more easily perform perspective taking and intention recognition [7], which is necessary to enable implicit coordination in joint human-robot tasks. Explicitly representing task variations as Dynamic Movement Primitives enables the robot to also recognize these variations in the behavior of others, and determine that it is a variation of a default trajectory, rather than a trajectory in its own right.

In the long run, we expect robots in human environments

[§]*Computational Learning and Motor Control Lab, University of Southern California, Los Angeles, CA, USA*

[†]*National Institute of Information and Communications Technology (NICT), Kyoto, Japan*

[‡]*Computational Neuroscience Laboratories, Advanced Telecommunications Research Institute International, Kyoto, Japan.*

[¶]*Intelligent Autonomous Systems Group, Technische Universität München, Munich, Germany*

Contact E-mail: stulp@clmc.usc.edu

to not have just a fixed set of motor primitives, but rather a growing library of 'compiled' solutions and variations upon them, which the robot autonomously tailors to the specific environment it is in, through chunking and optimization, as humans do [6].

The main contributions of this paper are: 1) Presenting a novel application of trajectory comparison methods from robotics and computer vision to human reaching data; 2) Demonstrating the advantages of explicitly representing standard solutions to standard tasks and task variations; 3) Showing how such variations are compactly represented as Dynamic Movement Primitives [4]; 4) Evaluating this approach in the context of a state-of-the-art humanoid robot.

The rest of this paper is structured as follows. In the next section, we discuss related work. In Section III, we describe how principal trajectories are extracted from human motion data, and in Section IV how these trajectories are executed on the robot using Dynamic Movement Primitives. We present the results of the empirical evaluation in Section V, and conclude with Section VI

II. Related Work

In recent years, impressive capabilities in performing manipulation in complex, cluttered and dynamic environments have been demonstrated by very general approaches based on, for instance, search [8] or potential fields [9]. However, by treating each problem as a novel one that needs to solved on-line (either through search or on-line adaptation), these approaches do not exploit task regularities that arise in everyday operation. The world is much simpler than the most general imaginable case, and these regularities, which Ian Horswill calls 'loopholes of life' [1] are there to be exploited. By using standard solutions, we achieve the advantageous properties listed in the previous section. However, such standard solutions will not work for completely novel tasks (e.g. very cluttered scenes), or very dynamic scenarios (e.g. quickly moving obstacles). From an engineering perspective, we see our approach not as a replacement, but as a complement to methods that deal with such circumstances [8], [9]. In Section VI, we give an example of how they can be combined.

Considerable work on determining the influence of obstacles on reaching motion has been done in experimental psychology [10], [11]. These studies use features of trajectories to determine if a trajectory is affected by an obstacle, such as lateral deviation from the default behavior in the xy-plane [10], or movement time, maximum grip aperture and maximum speed [11]. However, to derive compact models for robot control, we need to consider the trajectories as a whole, and cannot reduce it to only several features. Applying trajectory comparison methods from robotics and computer vision is one of the contributions of this paper.

Imitating trajectories was first used in the context of industrial robots 30 years ago, in very constrained task contexts and with fixed goals. By using Dynamic Movement Primitives to model the trajectories, adapting to novel goals is possible. Furthermore, we relate external task relevant parameters (the position of the target object) to internal motor parameters, i.e. which DMP to use to avoid the obstacle.

Jenkins and Matarić also propose a method for deriving modular skills from kinematic motion data of humans [12]. Their focus is on segmenting and clustering motion primitives from extended human dance routines. In contrast to relating obstacle positions to trajectory variations, the behavioral meaning of such statistical segments is often not clear. Also, they do not have inherent stability properties.

A related approach uses Gaussian mixture models to encode a set of trajectories [13]. One main difference to our approach is that Calinon et al. use kinesthetics (i.e. the human teacher moves the robot's actuators), whereas we use human motion data. As we strive for natural human-like motion, human motion data is essential to our approach. As to the methodology, Calinon et al. model the variance in the trajectory sets with Gaussian mixture models. One downside of this approach is that unwanted averaging effects may arise when multi-modal solutions exist. For instance, for several obstacle positions, the subject usually chooses avoid the obstacle by going around it on the left side, but sometimes also on the right side. The average, going in between, would lead to a certain collision. We deal with multi-modality by clustering the trajectories before processing them further.

III. Determining Principal Trajectories from Human Motion Data

This section explains how we 1) gather human data with a motion tracking system 2) discern between reaching motions that were influenced by an obstacle, and those that were not; 3) perform a clustering to acquire stereotypical reaching movements for the default and avoidance behaviors[1]. The goal of this analysis is to have the robot not imitate just any reaching motion, but have it imitate a *few* that enable it to deal with the task variations that occur. We call these few stereotypical trajectories the *principal trajectories*. How they are represented by the robot is described in Section IV.

A. Data Acquisition

The reaching motions were captured with a Polhemus Liberty magnetic position/orientation tracker. One sensor was attached to the hand, as depicted to the left in Fig. 1, and another sensor was attached to the glass to measure the exact time when the lifting movement started.

In the experiment, the subject sits at a table, and is asked to repeatedly reach for, grasp, and lift a target glass. The hand always starts in the black square in Figure 2. Before each reaching motion, an obstacle glass is placed on different positions on a 40x80cm grid on the table. In Figure 2 for example, the obstacle glass is at position $D6$. The target glass is always at position $B4$. The obstacle glass was placed 10 times on each of the 29 positions. Furthermore, 30 reaching motions were performed without any obstacle glass. These are the '*default*-trajectories' The total number of reaching

[1]Section III is a brief overview of the work presented in more detail in [14].

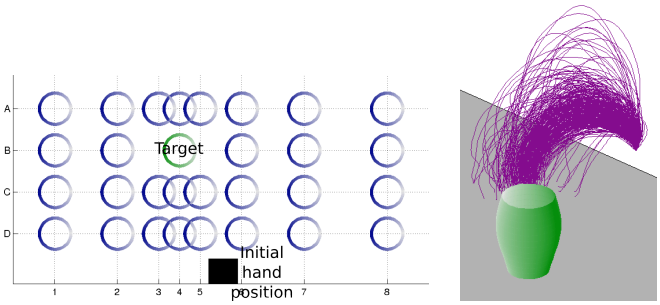


Fig. 2. Left: Positions of the obstacles on the table. The green glass is the target glass, which is always at position B4. The flat black region is the initial location of the fingers. Right: 320 recorded reaching trajectories.

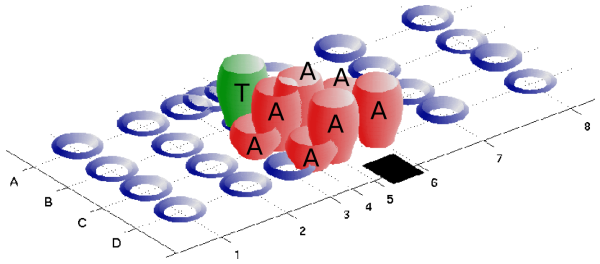


Fig. 3. The height of the glass represents d for that obstacle position. d_{thres} is 7.51 for this graph. The red glasses (at $C5$, $D5$, $D6$, $C4$, $C6$, $D4$, $C3$, marked 'A') lie above this threshold.

motions is therefore 29*10+30 = 320. The order of obstacle placement was random, to avoid learning effects.

After data acquisition, the trajectories are cropped so that they start when the hand starts moving, and end when the target cup starts moving. All trajectories are resampled using cubic spline interpolation so that they contain 100 samples. This is necessary to apply the Point Distribution Model, described in the next section.

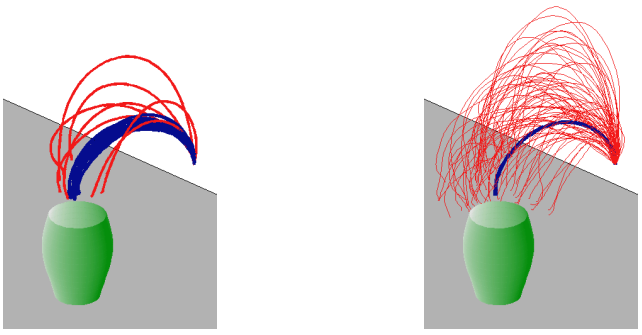
B. Discerning between default/avoid trajectory sets

If obstacles are far away from the target glass (e.g. at positions $A1$ or $C8$), we expect them not to have an influence on the reaching motion. One of the goals of this experiment is to determine the region in which obstacles influence the reaching motion. In this section, we describe a distance measure between sets of trajectories. For instance, we expect the distance between $A1$- and $C8$-trajectories to the *default*-trajectories to be small. We use this distance measure to discern between default and avoid trajectory sets.

We used the trajectory comparison approach described Roduit et al. [15]. Here, the difference measure between two sets of trajectories is computed by 1) computing a Point Distribution Model (PDM) of the two sets of trajectories by performing a Principal Component Analysis on the merged sets of trajectories 2) taking only the first n components of the deformation matrix in the PDM, by inspecting the eigenvalues of the covariance matrix of the merged trajectories 3) computing the Mahalanobis distance d between the coefficients of the two sets of trajectories. A more detailed explanation of this method can be found in [15].

This distance measure d is computed between all sets of trajectories $A..D1..8$, and the *default*-trajectories. The height of the glasses in Figure 3 represents d for the set of reaching motions when the obstacle was at that position. We automatically determine an appropriate threshold on d (called d_{thres}) by determining the valley point of the histogram of the d values. The distance d between the $C5$, $D5$, $D6$, $C4$, $C6$, $D4$, $C3$-trajectories and the *default*-trajectories is higher than this threshold. These seven positions are depicted as red glasses labeled 'A' in Figure 3.

Fig. 4(a) depicts the mean of the 10 trajectories for each obstacle position. The central blue bundle are the means for those obstacle positions for which $d < d_{thres}$, i.e. the obstacle did not influence the reaching motion. The red trajectories alongside are those when the obstacle *did* influence the reaching trajectory, and $d > d_{thres}$.



(a) Mean per obstacle position

(b) *AVOID* trajectories

Fig. 4. Discerning between default and avoidance trajectory sets.

In Fig. 4(b), the thick blue trajectory in the center is the mean of all default trajectories. The thin red trajectories are the 70 trajectories corresponding to the 7 obstacle positions where $d < d_{thres}$, i.e. when the obstacle influenced the reaching motion. These will be used for clustering in the next section.

C. Determining principal trajectories with clustering

The next step is determining principal trajectories that represent qualitatively different strategies for avoiding the obstacle. We do so by performing a k-means clustering on the 70 trajectories in the *AVOID*-trajectories set[2].

The clustering is performed in the 3D PCA space, being the first three deformation modes as defined in the previous section. The distance between two trajectories is determined by the angle between the two 3-dimensional vectors representing the deformation modes of the PDM. 7 clusters

[2]The reason why we do include all 320 trajectories in the clustering, is because it might be biased towards default behavior. For instance, suppose the table would have been $10m$ by $10m$, and we had placed obstacles at 10.000 positions on this table. We would expect that obstacles at only a few (e.g. the 4 we determined in the previous section) positions would affect reaching behavior. Including the unaffected trajectories for the other 9.996 positions in the clustering would lead to an over-representation of unaffected trajectories, and hence a bias. Therefore, we first split the sets of trajectories *DEFAULT* and *AVOID*, and perform clustering only on *AVOID*.

are chosen, because increasing the number of clusters leads the differences between the clusters to be lower than the threshold d, used to discern between *DEFAULT* and *AVOID* trajectories.

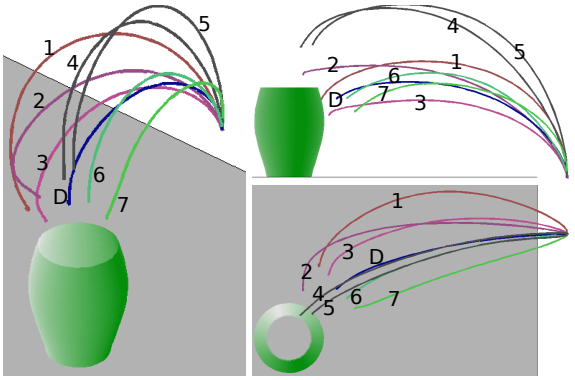


Fig. 5. The principal trajectories, seen from different angles. The default trajectory is in the center (D), and seven avoid trajectories are around it (1..7).

In Fig. 5, the averages of trajectories in the seven clusters are depicted. We call these the 'principal trajectories'. Furthermore, Fig. 6 depicts which trajectories are used to avoid which obstacle. For example, if the obstacle was at position '$D6$' (lower right), the subject used a reaching motion belonging to cluster 7 almost on out of four times, as indicated by the arrow.

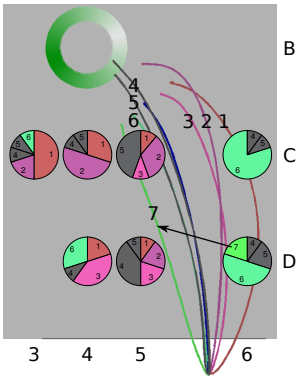


Fig. 6. The mapping from obstacle positions to principal trajectories. The default trajectory is used for all obstacle positions.

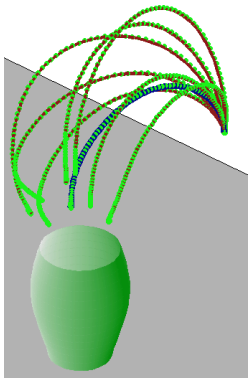


Fig. 7. Original principal trajectories ($\mathbf{x}$, dark lines), and their reproduction with a DMP ($\mathbf{x}_{\text{DMP}}$, bright markers).

IV. Trajectory Imitation with Dynamic Movement Primitives

This section briefly describes the dynamic movement primitive framework, discusses movement generalization to new goals, presents our modified DMP formulation, and its extension to obstacle avoidance.

A. Dynamic Movement Primitives

Dynamic Movement Primitives (DMPs) can be used to generate discrete and rhythmic movements. Here, we focus on discrete movements. A one dimensional movement is generated by integrating the following set of differential equations[3], which can be interpreted as a linear spring system perturbed by an external forcing term:

$$\tau \dot{v} = K(g-x) - Dv - K(g-x_0)s + Kf(s) \quad (1)$$

$$\tau \dot{x} = v \ , \quad (2)$$

where x and v are position and velocity of the system; x_0 and g are the start and goal position; τ is a temporal scaling factor; K acts like a spring constant; the damping term D is chosen such that the system is critically damped, and f is a non-linear function which can be learned to allow the generation of arbitrarily complex movements. This first set of equations is referred to as a transformation system. The non-linear function is defined as

$$f(s) = \frac{\sum_i w_i \psi_i(s) s}{\sum_i \psi_i(s)} \ , \quad (3)$$

where $\psi_i(s) = \exp(-h_i(s-c_i)^2)$ are Gaussian basis functions, with center c_i and width h_i, and w_i are adjustable weights. The function f does not directly depend on time; instead, it depends on a phase variable s, which monotonically changes from 1 towards 0 during a movement and is obtained by the equation

$$\tau \dot{s} = -\alpha \, s \ , \quad (4)$$

where α is a pre-defined constant. This last differential equation is referred to as canonical system. These sets of equations have some favorable characteristics: 1) Convergence to the goal g is guaranteed (for bounded weights) since $f(s)$ vanishes at the end of a movement. 2) The weights w_i can be learned to generate any desired smooth trajectory. 3) The equations are spatial and temporal invariant, i.e., movements are self-similar for a change in goal, start point, and temporal scaling without a need to change the weights w_i. 4) The formulation generates movements which are robust against perturbation due to the inherent attractor dynamics of the equations.

To learn a movement from demonstration, first, a movement $x(t)$ is recorded and its derivatives $v(t)$ and $\dot{v}(t)$ are computed for each time step $t = 0, \ldots, T$. Second, the canonical system is integrated, i.e., $s(t)$ is computed for an appropriately adjusted temporal scaling τ. Using these arrays, $f_{\text{target}}(s)$ is computed based on (1) according to

$$f_{\text{target}}(s) = \frac{\tau \dot{v} + D\, v}{K} - (g-x) + (g-x_0)\, s \ . \quad (5)$$

where x_0 and g are set to $x(0)$ and $x(T)$, respectively. Thus, finding the weights w_i in (3) that minimize the error criterion $J = \sum_s \big(f_{\text{target}}(s) - f(s)\big)^2$ is a linear regression problem, which can be solved efficiently. We solve it using Locally Weighted Projection Regression (LWPR) [16], which is a locally weighted learning approach, in which the number of local linear models required to approximate the function is

[3]We use the formalization of DMPs as proposed in [9, Section III].

determined automatically. The centers c_i and bandwidths h_i of the basis functions representing the receptive fields of the local linear models are also determined automatically.

A movement plan is generated by reusing the weights w_i, specifying a desired start x_0 and goal g, setting $s = 1$, and integrating the canonical system, i.e. evaluating $s(t)$. The obtained phase variable then drives the non-linear function f which in turn perturbs the linear spring-damper system to compute the desired attractor landscape.

The trajectory of the end-effector in 3D Euclidean space is acquired by training one such DMP per dimension, and coupling their phase s during movement execution.

B. Application to Reaching Trajectories

For each of the reaching motions in Fig. 5, a DMP is trained on the 3D trajectory of the end-effector in Euclidean space. Each dimension starts of with 4 basis functions, which is too few to describe the trajectory accurately. During training LWPR then adds basis functions and adapts their centers and weights until the trajectory is approximated well enough, i.e. the error criterion J summed over all dimensions drops below 0.005. In our experience, this value yields a good trade-off between the accuracy of the trajectory following and the number of basis functions used.

The trajectories these DMPs generate are depicted in Fig. 7, alongside the original trajectories. We introduce the following notation for these concepts:

- $\mathbf{x}$: An end-effector trajectory in 3D Euclidean space
- $\mathbf{x}^{\text{—}}$: The *default* trajectory in 3D Euclidean space
- $\mathbf{x}^{\sim n}$: The n^{th} *avoid* trajectory. The — and $\sim$ symbols are mnemonics, referring to a relatively 'straight' default or 'curved' avoidance trajectory.
- DMP$^{\text{—}}$/DMP$^{\sim n}$: The DMP learned from the default/n^{th} avoid trajectory.
- $\mathbf{x}^{\text{—}}_{\text{DMP}}$ or $\mathbf{x}^{\sim n}_{\text{DMP}}$: The trajectory generated by the DMP learned from the default or n^{th} avoid trajectory.

From Fig. 7, it is clear that reproduced trajectories are very similar to the original ones. The mean number of basis functions per trajectory required to represent these trajectories is 41.8. Note that this is 13.9 basis functions for each of the three DMP dimensions, representing the x,y and z component of the trajectory.

C. Compact representation of avoidance strategies

Learning one DMP for each trajectory, independently of the other trajectories, ignores the strong relationship between them. An alternative representation is to use the DMP$^{\text{—}}$ to represent the default trajectory $\mathbf{x}^{\text{—}}$, and train other DMPs with the *difference* between the default and avoid trajectories.

There are several advantages to representing solutions of task variations as modifications to the default trajectory, rather than as solutions in their own right. From a behavioral point of view, there are no advantages, as the resulting behavior is the same (compare Fig. 7 and Fig. 9). However we believe this representation is conceptually preferable, and generalizes better to novel situations.

First of all, related actions are encoded together, and with the same representation, which makes the relation between them explicit. This leads to a more compact representation of the trajectories, as we shall see in Table II. Asfour et al. [17], who use a similar 'difference strategy' to encode trajectories, motivate that such a representation generalizes better to new trajectories. Furthermore, recent work has demonstrated that it is possible to directly learn a mapping from external task-relevant parameters to internal motor parameters, if a compact representations for actions are available, for instance as B-splines [18]. We are currently evaluating how compact representation of obstacle avoidance strategies enable a direct mapping from obstacle position to motor primitive parameterization. Finally, DMPs can also be used for movement recognition [4]. We expect that explicitly representing task variations as DMPs enables the robot to also recognize these variations in the behavior of others, and determine that it is a variation of a default trajectory, rather than a trajectory in its own right.

To learn difference trajectories, we first generate a trajectory with the default DMP$^{\text{—}}$, but scaled to the goal of the avoid trajectory $\mathbf{x}^{\sim n}$. This yields a trajectory which we denote $\mathbf{x}^{\text{—}}_{\text{DMP}\Rightarrow g^n}$. Even if DMP$^{\text{—}}$ is scaled to the novel goal, there are still qualitative differences between $\mathbf{x}^{\text{—}}_{\text{DMP}\Rightarrow g^n}$ and $\mathbf{x}^{\sim n}$, as can be seen in Fig. 8. We therefore compute the difference between the generated scaled trajectory, and the avoid trajectory, which yields $\mathbf{x}^{\Delta}$.

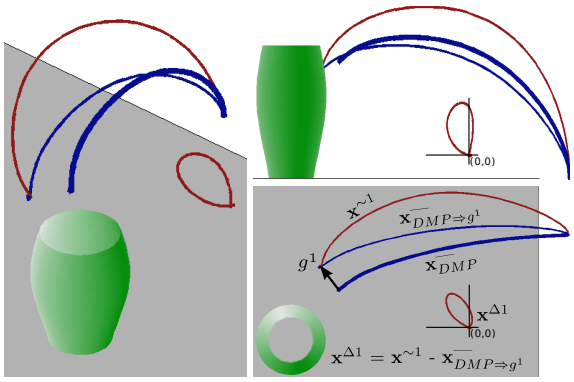


Fig. 8. The process of computing the difference trajectory $\mathbf{x}^{\Delta 1}$ from the avoid trajectory $\mathbf{x}^{\sim 1}$ and the trajectory $\mathbf{x}^{\text{—}}_{\text{DMP}\Rightarrow g^1}$, generated by the default DMP, but with the goal g^1. Note that $\mathbf{x}^{\Delta 1}$ does not share the same coordinate frame with the other trajectories, as indicated by its own small axes. As the start- and end-point of $\mathbf{x}^{\sim n}$ and $\mathbf{x}^{\text{—}}_{\text{DMP}\Rightarrow g^n}$ always coincide, each $\mathbf{x}^{\Delta n}$ starts and ends at (0,0,0).

The final step is to train a DMP for each of the $\mathbf{x}^{\Delta}$ trajectories, which yields the Dynamic Movement Primitives $\mathbf{x}^{\Delta}_{\text{DMP}}$. In Fig. 9, the original and reproduced trajectories are depicted (the butterfly-like shape). Now, to reconstruct the n^{th} avoid trajectory, we compute $\mathbf{x}^{\sim n} = \mathbf{x}^{\text{—}}_{\text{DMP}\Rightarrow g^n} + \mathbf{x}^{\Delta n}_{\text{DMP}}$.

That this representation is more compact becomes clear from Table II, where this approach is listed in the second row. Of course, the default trajectory is still represented with DMP$^{\text{—}}$, so its number of basis functions does not change. However, by representing the avoid trajectories with

Obst-acle pos.	Simulation Default SR (20x)	MD	Simulation Avoid Trajs. used (20x)	SR (20x)	MD	Real robot Default SR (4x)	Real robot Avoid Trajs. used (1x)	¬used	SR
C3	1.00	4.1	1 2 4 5 6	1.00	7.1	1.00	1 2 4 5 6	3 7	1.00
C4	0.00	N.A.	1 2 4 5	1.00	2.5	0.00	1 2 4 5	~~3~~ ~~6~~ ~~7~~	1.00
C5	0.00	N.A.	1 ~~2~~ ~~3~~ 5	0.75	1.0	0.00	~~1~~ ~~2~~ ~~3~~ 5	4 ~~6~~ ~~7~~	0.40
C6	1.00	5.2	4 5 6	1.00	5.7	1.00	4 5 6	~~1~~ ~~2~~ ~~3~~ 7	1.00
D4	0.00	N.A.	1 3 4 6	1.00	1.2	0.00	1 3 4 ~~6~~	2 ~~5~~ ~~7~~	0.70
D5	0.00	N.A.	~~1~~ 2 ~~3~~ 4 ~~5~~	0.60	0.8	0.00	1 2 3 4 5	~~6~~ ~~7~~	1.00
D6	1.00	2.5	4 5 6 7	1.00	4.5	1.00	4 5 6 7	~~1~~ ~~2~~ ~~3~~	1.00

TABLE I

SUCCESS RATES IN SIMULATION AND ON THE REAL ROBOT (SR=SUCCESS RATE, MD=MINIMUM DISTANCE).

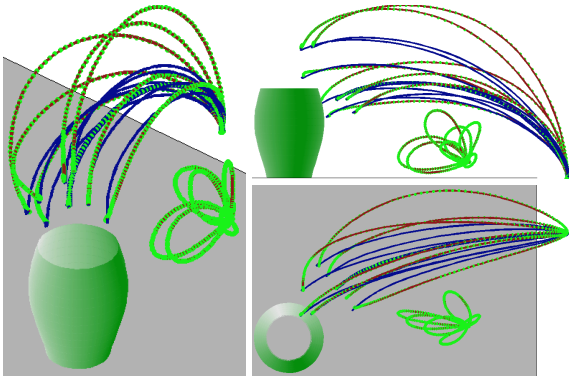


Fig. 9. Reproducing the trajectories using DMPs, as a summation of $\mathbf{x}_{\overline{\text{DMP}\Rightarrow g^n}}$ and $\mathbf{x}^{\Delta n}_{\text{DMP}}$.

DMP$^{\Delta}$ instead of DMP$^{\sim}$, the number of basis functions required to represent the avoid trajectories halves from 42.6 to 21.1, whilst the error between the original trajectory and the generated trajectory remains the same.

	Number of basis functions All (8x) $\sum_8$	Default (1x) $\sum_1$	Avoid (7x) $\sum_7$
1 DMP^{-} +			
7 DMP$^{\sim}$	41.8	36.0	42.6
7 DMP$^{\Delta}$	23.0	36.0	21.1

TABLE II

AVERAGE NUMBER OF BASIS FUNCTIONS NEEDED TO ENCODE THE TRAJECTORIES AS A DMP.

V. EMPIRICAL EVALUATION

The empirical evaluation was conducted with the humanoid robot Sarcos CB [5], both in simulation and on the real robot. To execute the DMPs, we compute the joint angles from the end-effector trajectory by using velocity-based inverse kinematics based on pseudo-inverse of the Jacobian.

In simulation, the robot performed 20 reaching motions for each obstacle position. In each episode, the trajectory chosen to avoid the obstacle was chosen probabilistically, according to the distributions depicted in Fig. 6. For comparison, the default trajectory was also used for each episode.

Table I summarizes the results in both simulation and on the real robot. The first column lists the seven obstacle positions, and then the success rates (SR) of applying the default and probabilistically chosen avoid trajectories. For the avoid trajectories 1..7, the column denoted 'Trajs. used' lists which chosen avoid trajectories were successful, and which failed. ~~n~~ indicates that the robot collided with the obstacle when executing principal trajectory n. As a quality measure, we also list the mean minimum distance (MD) to the obstacle over the successful reaching motions. 'N.A.' (not applicable) indicates that no trajectory was successful.

We draw the following conclusions from these results: 1) Using avoid trajectories substantially improves the success rate of reaching over using the default trajectory, and also increases the mean minimum distance to the obstacle. 2) There are three obstacles positions (C3,C6,D6), where humans choose an avoid strategy, but the robot can also successfully use the default trajectory. However, using an avoid trajectory does increase the minimum distance to the obstacle, so an advantage remains. 3) Not all avoid trajectories are successful for the robot. These can be excluded during operation. We see these results as the basis for further refinement (i.e. with Reinforcement Learning) of the avoidance strategies the robot should use.

VI. CONCLUSION

In this paper, we have presented a novel application of methods for comparison and clustering of robotic trajectories to human reaching data. We have demonstrated how the principal trajectories that arise from this analysis are compactly modeled as variations on a default motor primitive. Here, the variations share the representation with the default, but have a lower-dimensionality. By maintaining the specificity of motor primitives for a certain task, but also generalizing them to common variations on this task, we are essentially exploiting the loophole that the task and its variations frequently occur.

The Point Distribution Model has proven to be a versatile and effective tool for the comparison of sets of trajectories. Instead of choosing one or two trajectory features manually, this approach considers the trajectories as a whole, and objectively extracts relevant features from them automatically using PCA. We see great potential for using this technique

in other reaching motion experiments from experimental psychology.

Pre-planned trajectories can still easily be combined with potential fields to avoid dynamic objects. Although not implemented for this paper, Fig. 10 demonstrates how our approach of having standard solutions to avoiding state obstacles, can be elegantly combined with the standard DMP approach ([4]), extended with potential fields for on-line avoidance of dynamic obstacles ([9]).

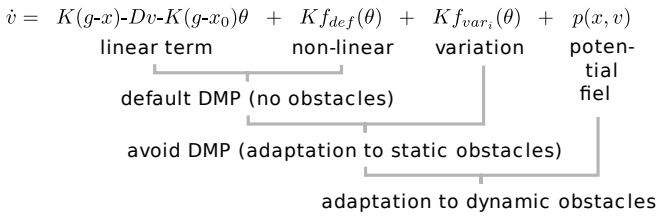


Fig. 10. The different parts of the extended DMP formula, and their role in adaptation to static and dynamic task variations. Here, $\mathbf{p}(\mathbf{x}, \mathbf{v})$ is a potential field that repels the end-effector away from the obstacle, depending on the distance and relative velocities between them [9].

Also, we are interested in generalizing over objects, by analyzing how different properties of an object (i.e. size, shape, etc.), rather than specific objects themselves, influence the trajectory. By finding commonalities between features, and generalizing over them, we believe the number of motor primitive variation required to deal with a large set of tasks can be kept quite low. We will also focus on integrating grasp planning algorithms with our approach, which for now focuses on reaching trajectories only.

VII. Acknowledgements

We would like to thank Etienne Bousquié, Ingo Kresse, Alexis Maldonado, and Federico Ruiz for their assistance in gathering the data. Freek Stulp was supported by a Post-doctoral Research Fellowship from the Japanese Society for the Promotion of Science. The research described in this article is also partially funded by the CoTeSys cluster of excellence (Cognition for Technical Systems, `http://www.cotesys.org`), part of the Excellence Initiative of the DFG.

References

[1] I. D. Horswill, "Specialization of perceptual processes," Ph.D. dissertation, MIT, Cambridge, MA, USA, 1993.

[2] T. Flash and B. Hochner, "Motor primitives in vertebrates and invertebrates," *Current Opinion in Neurobiology*, vol. 15, 2005.

[3] S. Schaal and N. Schweighofer, "Computational motor control in humans and robots," *Current Opinion in Neurobiology*, vol. 15, pp. 675–682, 2005.

[4] A. J. Ijspeert, J. Nakanishi, and S. Schaal, "Movement imitation with nonlinear dynamical systems in humanoid robots," in *International Conference on Robotics and Automation (ICRA2002)*, 2002.

[5] G. Cheng, S. Hyon, J. Morimoto, A. Ude, J. Hale, G. Colvin, W. Scroggin, and S. C. Jacobsen, "CB: A humanoid research platform for exploring neuroscience," *Journal of Advanced Robotics*, vol. 21, no. 10, pp. 1097–1114, 2007.

[6] R. Sosnik, B. Hauptmann, A. Karni, and T. Flash, "When practice leads to co-articulation: the evolution of geometrically defined movement primitives," *Experimental Brain Research*, no. 156, 2004.

[7] E. Oztop, D. Franklin, T. Chaminade, and G. Cheng, "Human-humanoid interaction: Is a humanoid robot perceived as a human?" *International Journal of Humanoid Robotics*, vol. 2, 2005.

[8] D. Berenson, R. Diankov, K. Nishiwaki, S. Kagami, and J. Kuffner, "Grasp planning in complex scenes," in *IEEE-RAS International Conference on Humanoid Robots*, 2007.

[9] H. Hoffmann, P. Pastor, D.-H. Park, and S. Schaal, "Biologically-inspired dynamical systems for movement generation: Automatic real-time goal adaptation and obstacle avoidance," in *IEEE International Conference on Robotics and Automation*, 2009.

[10] C. S. Chapman and M. A. Goodale, "Missing in action: the effect of obstacle position and size on avoidance while reaching," *Experimental Brain Research*, 2008.

[11] M. Mon-Williams, J. R. Tresilian, V. L. Coppard, and R. G. Carson, "The effect of obstacle position on reach-to-grasp movements," *Experimental Brain Research*, vol. 137, pp. 497–501, 2001.

[12] O. C. Jenkins and M. J. Matarić, "Performance-derived behavior vocabularies: Data-driven acquisition of skills from motion," *International Journal of Humanoid Robotics*, vol. 1, no. 2, 2004.

[13] S. Calinon, F. Guenter, and A. Billard, "On learning, representing and generalizing a task in a humanoid robot," *IEEE Transactions on Systems, Man and Cybernetics, Special issue on robot learning by observation, demonstration and imitation*, vol. 37, no. 2, 2007.

[14] F. Stulp, I. Kresse, A. Maldonado, F. Ruiz, A. Fedrizzi, and M. Beetz, "Compact models of human reaching motions for robotic control in everyday manipulation tasks," in *Proceedings of the 8th International Conference on Development and Learning (ICDL). To appear.*, 2009.

[15] P. Roduit, A. Martinoli, and J. Jacot, "A quantitative method for comparing trajectories of mobile robots using point distribution models," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2007, pp. 2441–2448.

[16] S. Vijayakumar, A. D'Souza, and S. Schaal, "Incremental online learning in high dimensions," *Neural Computation*, vol. 17, 2005.

[17] T. Asfour, K. Welke, A. Ude, P. Azad, J. Hoeft, and R. Dillmann, "Perceiving objects and movements to generate actions on a humanoid robot," in *ICAR Workshop: From features to actions - Unifying perspectives in computational and robot vision*, 2007.

[18] A. Ude, M. Riley, A. Kos, B. Nemec, T. Asfour, , and G. Cheng, "Goal-directed action synthesis from a library of example movements," in *Proc. of the IEEE-RAS Int. Conf. on Humanoid Robots*, 2007.