Receding Horizon Optimization of Robot Motions generated by Hierarchical Movement Primitives

Manuel Mühlig[1], Akinobu Hayashi[1], Michael Gienger[1], Soshi Iba[2] and Takahide Yoshiike[2]

Abstract— **This paper introduces a motion generation framework that integrates a hierarchical movement primitive (MP) layer with optimal control in form of receding horizon optimization. In order to benefit from fast reactions on the MP-layer, the optimal control layer can be overridden in *risky* situations to generate quick, though non-optimal solutions. By this, the system fulfills four desirable properties. It continuously adapts the robot's motion without noticeable delay (1) by optimizing for collision and joint limit avoidance based on a future time horizon instead of the current state only (2). It accounts for the full robot motion that may result from multiple active MPs at the same time (3) and despite a possibly slow optimization still provides the robustness and quick reaction capabilities of MPs (4). The framework has been validated in an experiment in which a humanoid robot performed a task, optimized wrt. collisions and joint limit avoidance, but still could react within 50 ms after detection of a potential risk.**

I. INTRODUCTION

In the recent years, the concept of MPs has become a major trend in the research field of robotics. Although there is not yet a full consensus about their definition, the common understanding is that MPs are parametric representations of elementary motions that can be recalled efficiently. Besides their biological motivation [1], there are several features that roboticists appreciate. For one thing, many MP representations use dynamical systems for generating motions, which makes them inherently robust to disturbances. This is especially true for simple attractors [2], for Dynamic Motion Primitives (DMPs) [3], but also for representations where robustness is the target of an optimization process [4]. For another, the compact parametrization of MPs proved to be beneficial for applying machine learning techniques to learn or to improve motions [5].

One particular drawback however is that MPs alone rarely can account for generating optimal motions in new, and unknown situations. For real world tasks, especially collisions and the robot's limitations have to be regarded. Further, for co-articulated movement skills which are characterized by multiple active MPs in parallel, the resulting motion has to satisfy constraints imposed by the robot's capabilities.

Several approaches to incorporate such criteria in the motion generation process have been proposed in the past. One group of work regards them as external disturbances that reactively influence the robot's state. The dynamical system representation of the MP then accounts for such disturbances by generating a new motion starting from the modified state. As an example, the authors of [6] and [7] adapt the DMP formulation by including a term for collision avoidance. Objects in the scene generate a potential field in task space that applies virtual repelling forces to the robot's end-effector.

Another body of work accounts for robot-specific criteria within the MP representation by utilizing explorative learning. In [8] it is shown how such a learning approach can consider the robot's unmodeled dynamics and adapt to generalize a task to new situations. The authors of [9] employ learning to embed a minimum-jerk criterion into the MP representation.

A third category of work to account for the specifics of the situation and the robot is based on optimization methods. For example, the authors of [10] optimize the robot's motion to follow a recorded human trajectory, and at the same time to respect the robot's joint limits. A tradeoff that needs to be considered there is the optimization quality vs. the calculation time. A long delay before performing the movement is often considered to be unnatural. In our recent work [11], we employed optimization before performing a learned movement primitive. We circumvented the delay problem by optimizing the movement in parallel during a preceding phase of preparatory movements. Another alternative approach to employ optimization is presented by [12] who directly endow MPs with probabilistic planning capabilities. When executing the movement, the MP transparently plans the motions that are suitable for the current situation.

Looking at the state of the art of movement generation with MPs, a coherent solution to generate optimal motions is not yet existing. We believe that such an approach needs to account for

1) a continuous adaptation of the robot's motion to the actual situation,
2) by considering a future time horizon and not only the current state,
3) regarding the full body motion of the robot that results from the combination of multiple MPs in parallel,
4) and still providing the robustness and quick response of a reactive system using MPs.

In this paper we propose a two-layered system that addresses these points by combining a hierarchical movement primitive framework with receding horizon optimization. The first section introduces the motion generation concept, which is based on a previously published framework based on Continuous-Time Recurrent Neural Networks (CTRNNs) to

[1]Manuel Mühlig, Akinobu Hayashi, and Michael Gienger are with the Honda Research Institute Europe GmbH, Offenbach/Main, Germany, {firstname.lastname}@honda-ri.de

[2]Soshi Iba and Takahide Yoshiike are with Honda R&D Co., Ltd. Fundamental Technology Research Center, Wako-shi, Saitama, Japan, {firstname_lastname}@n.f.rd.honda.co.jp

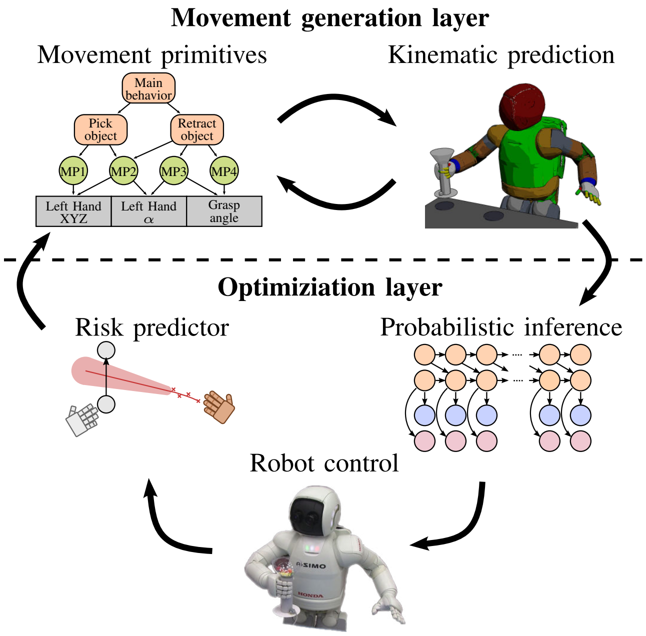

Fig. 1. Overview of the receding horizon optimal control framework. The initial motion is continuously generated by the upper layer and optimized by the lower layer. The detected risk of the motion is directly coupled into the MP hierarchy.

sequentialize MPs [13], [14]. In Section III, we present the optimization paradigm and the criteria used to adapt the motion to the actual situation. After this, Section IV describes how the two layers – movement generation and optimization – are combined in a way that still allows for quick reactions by bypassing the optimization if a potential risk is predicted. Finally, Sections V and VI conclude with a demonstration of the system in an experiment and an outlook, respectively.

II. MOTION GENERATION

In this section we briefly review the motion generation process of the movement generation layer of the framework depicted in Figure 1. Firstly, we describe the modular structure of the MPs that allows to organize them in a hierarchy in order to compose complex skills. Secondly, we introduce Continuous-Time Recurrent Neural Networks (CTRNNs) as a special kind of MPs that allow a flexible arbitration of MPs to realize sequential and co-articulated movements.

A. Movement Primitives

In the following, the term *movement primitive* should be understood as a representation for elementary, goal-directed motions described in joint or task space. A specific representation itself is not the focus of this paper and basically any that follows the dynamical systems equation can be used:

$$\dot{\boldsymbol{y}} = f(\boldsymbol{y}, \boldsymbol{\theta}) \tag{1}$$

In the equation, the velocity $\dot{\boldsymbol{y}}$ in state space is the result of a possibly nonlinear function dependent on the current state $\boldsymbol{y}$ and a parametrization $\boldsymbol{\theta}$. The state space can be different for each MP and is a subset of the full state space (e. g., the position of an end-effector). As mentioned in the introduction, there is large body of work about different kinds of MP formulations. In this paper, we employed simple linear attractor dynamics that can be easily designed by hand and are inherently robust. They are designed to be stable throughout the full state space.

In order to seamlessly use the MPs within a hierarchy, we extend the definition of a MP with following elements:

1) An *activation level* $a \in [0, 1]$ to scale its output and such the activation of connected lower-level MPs.
2) A *goal distance* $g \in [0, 1]$ that introduces a metric to describe the difference between current and desired state.
3) A *prediction error* $p \in [0, 1]$ which is a measure of the difference between the generated motion and the expected one. Deviations might be the result from disturbances or due to co-activation of multiple MPs.
4) A *responsibility* $r \in [0, 1]$ to model how well the MP fits to the current situation.

The activation level a is an input for the MP coming from a higher level in the hierarchy. In contrast, the magnitudes g, p, and r are calculated by each MP as a form of situation assessment to signalize the MP's internal state to higher-level MPs.

On the lowest level of the hierarchy the output of the MPs is being integrated and the error $\Delta \boldsymbol{e}$ between the current state in task space and the desired state resulting from the lowest-level MPs is mapped to a change of joint angles $\delta \boldsymbol{q}$ using a standard resolved motion rate control formulation:

$$\delta \boldsymbol{q} = \boldsymbol{J}^{\#} \Delta \boldsymbol{e} - \alpha (I - \boldsymbol{J}^{\#} \boldsymbol{J}) \left(\frac{\partial H}{\partial \boldsymbol{q}} \right)^T \tag{2}$$

The term $\boldsymbol{J}^{\#}$ stands for the regularized (λ), weighted, left pseudoinverse of the task Jacobian $\boldsymbol{J}$:

$$\boldsymbol{J}^{\#} = \left(\boldsymbol{J}^T \boldsymbol{W}_{\mathrm{x}} \boldsymbol{J} + \mathrm{diag}(\lambda) \right)^{-1} \boldsymbol{J}^T \boldsymbol{W}_{\mathrm{x}} \tag{3}$$

Matrix $\boldsymbol{W}_{\mathrm{x}}$ weights the contributions of the different tasks. It is a diagonal matrix whose elements correspond to the task activations a_i from the MP outputs. The cost function H, mapped into the nullspace with weight α, is composed of joint limit and collision avoidance criteria. The resulting joint angle displacements are integrated and sent to the control system of the robot.

B. Neural Dynamics

In the hierarchy of MPs, the higher levels are responsible to coordinate parallel and sequential activation of lower-level MPs. In order to discriminate these complex behaviors from the simple goal-directed motions, the higher-level MPs are referred to as *skills*. Skills, as defined in this paper, are MPs that have two specific properties. For one, skills generate a continuous stream of activations that modulate the activity of connected MPs in contrast to computing desired task-level commands. For another, the activation dynamics are implemented as dynamical systems in form of Continuous-Time Recurrent Neural Networks (CTRNNs).

As described in previous publications [13], [14], the formulation of the activation dynamics is strongly influenced by earlier work of Sandamirskaya, Schöner et al. [15]. Each neuron of the CTRNN corresponds to one MP that is modulated by the activation dynamics. The activation of each MP i is computed from the neuron's potential u_i using a Sigmoid function $a_i = f_\sigma(u_i)$. The change of one neuron's potential is governed by the following dynamical system equation[1]:

$$\tau \dot{u}_i = -u_i - \kappa \cdot u_i^3 + h + w_0 \cdot f_\sigma(u_i) + s_i \tag{4}$$

with time constant τ, a resting level $h < 0$, self-excitation weight w_0, and additional stimulus or inhibition s_i. We chose the parameter $w_0 = -2 \cdot h$ with $h = -1.8$ and thus the dynamical system establishes two symmetric stable fix points at resting level h and $h + w_0$ if there is no external stimulus s_i.

The stimulus s_i is composed of different terms that account for the activation of the skill itself (a_s), a recurrent inhibition, and the current state of the controlled MP represented by its goal distance, responsibility, and prediction error:

$$\begin{aligned} s_i = \quad & w^\mathrm{e} \cdot a_\mathrm{s} && \text{(base excitation)} && (5)\\ & - \sum_{k, k\neq i} w^\mathrm{i}_{ik} \cdot f_\sigma(u_k) && \text{(recurrent inhibition)} && (6)\\ & - \sum_{j} w^\mathrm{g}_{ij} \cdot g_j && \text{(goal distance)} && (7)\\ & - w^\mathrm{r} \cdot (1 - r_i) && \text{(responsibility)} && (8)\\ & - w^\mathrm{p} \cdot p_i && \text{(prediction error)} && (9)\\ & - w^\varrho \cdot \varrho && \text{(estimated risk)} && (10) \end{aligned}$$

The weights w are used to define the influence of each term on the activation of the specific MP i. For the skills used this paper's experiments they have been hand-defined. The last term (Equation 10) is an external stimulus that is not computed by each MP individually. Rather it is a global *estimated risk*, which for the experiment results from the predicted motion of a human disturber. Its detailed calculation and its role will be explained in Sections IV and V.

C. Kinematic prediction

As mentioned earlier, all MPs including the skills implement a common interface which allows to combine them hierarchically. In combination with the controller on the lowest level, the hierarchy of MPs can be used to control a robot reactively. This has already been shown previously in [13], where a simulated Barret WAM arm performed different grasping sequences. A shortcoming of that approach however is that it cannot handle joint limits and collision avoidance in task space properly. To account for this, we combine the reactive system with an optimization approach that takes a future time horizon into account.

To provide a prior to the optimization a prediction of the robot's movement is generated using a kinematic simulation. It is assumed that the models of the environment and the robot are known and that the robot can locate itself and the objects in the scene. The simulation is initialized with the current state of the real robot and the perceived object transformations. Starting from this state the simulation is iterated for a number of time steps T_h (prediction horizon). At the same time, the motion generation using the MPs generates the simulated robot's movements as if the real robot would be controlled. To account for the movement of the object after the robot grasps it, a heuristic rather than a computationally expensive physics simulation is used. The object is moved according to the robot's hand movement if their distance is small and the hand is closed.

Along with the simulation, the robot's joint space trajectory $\boldsymbol{Q} = (\boldsymbol{q}_0, \ldots, \boldsymbol{q}_{T_\mathrm{h}-1})$ and the trajectory of MP activations $\boldsymbol{A} = (\boldsymbol{a}_0, \ldots, \boldsymbol{a}_{T_\mathrm{h}-1})$ are recorded. They serve as input to the subsequent optimization process.

[1] See [13] for a more compact matrix formulation.

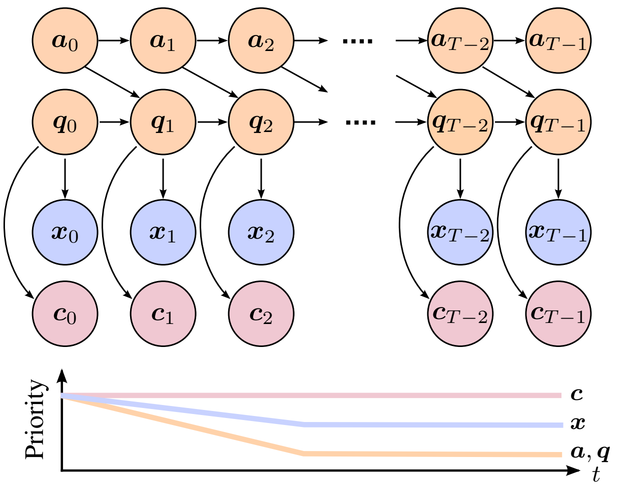

Fig. 2. Bayesian network representation of the system (top) and the concept of setting the priority of each constraint over time (bottom).

III. MOVEMENT OPTIMIZATION

In order to adapt the prior movement, described by $\boldsymbol{A}$ and $\boldsymbol{Q}$, to the actual situation, a movement optimization step is being carried out. Its role is to adapt the movement to satisfy criteria such as joint limit and collision avoidance. Here, we chose a probabilistic inference approach for the optimization, as it is straightforward to incorporate prior information. Note however, that the overall concept does not depend on this and practically any movement optimization approach can be applied (e. g., [16], [17], [18]).

A. Problem definition

Probabilistically the problem can be modeled in form of a Bayesian network as depicted in Figure 2. The progression of the MP activation and the joint angle trajectory over time is defined by:

$$\begin{pmatrix} \boldsymbol{q}_{t+1} \\ \boldsymbol{a}_{t+1} \end{pmatrix} = \begin{pmatrix} f(\boldsymbol{a}_t, \boldsymbol{q}_t) \\ \boldsymbol{a}_t \end{pmatrix} + \begin{pmatrix} \boldsymbol{\epsilon}_t \\ \boldsymbol{\xi}_t \end{pmatrix} \tag{11}$$

with joint angles $\boldsymbol{q}_t$, MP activations $\boldsymbol{a}_t$, and Gaussian noise $\boldsymbol{\epsilon}_t \sim \mathcal{N}(0, \sigma_t^\mathrm{q})$, and $\boldsymbol{\xi}_t \sim \mathcal{N}(0, \sigma_t^\mathrm{a})$. The function f models a control step following Equation 2. It calculates the joint configuration of the next time step given the current joint configuration and the activations of the MPs. The change of MP activations themselves is governed by the noise $\boldsymbol{\xi}_t$.

The cost function used for the optimization accounts for several criteria:

- Movement similarity
  A similarity to the movement computed by the kinematic prediction explained above. In the subsequent experiments, we utilize the left and right end-effector transformations (6D poses) and the grasping joint angles for the left and right hand (2 DoF per hand). This criterion leads to a bias towards the movement generated in the MP-layer if no other criteria is violated.
- Collision avoidance
  This criterion penalizes the proximity of a given set of body pairs and follows closely the algorithms in [19].
- Joint limit avoidance [19]

B. Optimization based on probabilistic inference

The optimization problem is to find trajectories of MP activations and joint angles that minimize the costs over a finite horizon. We apply Extended Kalman Smoothing (EKS), a probabilistic inference method to estimate the optimal trajectories (maximum likelihood trajectories). EKS is an efficient way of inferring, because it uses Taylor expansions to linearize the system and subsequently uses Gaussian messages for belief propagation. As we assume Gaussian messages in the probabilistic model, we approximate $\boldsymbol{a}_t$ as a vector of real values, instead of as a vector of values in the interval $[0, 1]$. When $\boldsymbol{a}_t$ is interpreted as activations of MPs its values are clipped. It is subject to future work to investigate other approaches that can handle the exact range of $\boldsymbol{a}_t$ in the probabilistic model.

For the inference, EKS needs gradients to approximate the nonlinear functions. For the tasks $\boldsymbol{x}$, the collision cost c_{ca} and joint limit cost c_{jl}, we use the analytical gradients as described in earlier work [19]. The system matrix's gradients $\partial \boldsymbol{q}_{t+1}/\partial \boldsymbol{q}_t$, $\partial \boldsymbol{q}_{t+1}/\partial \boldsymbol{a}_t$ are calculated using finite differences.

To weight the importance of the different criteria, the covariance of the nodes in the Bayesian network is used. The actual covariance values are not comparable across the different criteria, but the concept of prioritizing them is $cost > task > prior$ (where a low covariance means high priority). Figure 2 shows the concept of how to set the prioritization over time. In order to avoid discontinuities, the priority of all criteria is high at the initial time step. This means, the optimizer follows the prior relatively strict. By gradually decreasing the priority, the prior is respected during the initial time steps while during later time steps it is used as rough reference. The joint limit and collision criteria have a constant high priority, as they are most important to assure a safe movement.

The result of the optimization is a joint space trajectory of length T_{h} that can be executed on the real robot.

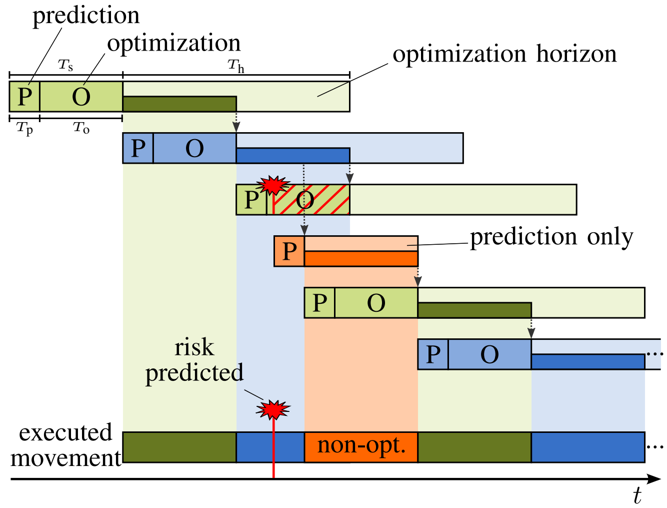

Fig. 3. The future robot movement is predicted and optimized in parallel to the execution of the current movement. A predicted increase of *risk* can lead to an override of the previously optimal movement with a quick, but non-optimal solution. The arrows pointing downward indicate which state of the previous optimization is used a starting point for the subsequent one.

IV. RISK-AWARE RECEDING HORIZON CONTROL

In the previous sections, the layers of the proposed framework – motion generation and motion optimization – have been presented. Together they fulfill two requirements, mentioned in the introduction, which are the "optimization taking into account a future time horizon" (2) and the motion resulting from the "combination of multiple MPs in parallel" (3). In this section the remaining two requirements (1,4) are investigated.

A. Interruptible receding horizon control

To achieve a continuous optimization of the robot's movement, a receding horizon approach is used [20]. This means, the robot's future movement is simulated by the MP-layer, and optimized in parallel to the robot's current movement.

Figure 3 shows an illustration of the approach. The bottom row represents the robot's motion that results from the parallel optimization that is shown in the upper part of the image. The idea of the receding horizon implementation is to start the prediction and optimization couple from an expected future state that results from a previous prediction and optimization. The sampling interval T_{s} is defined as the expected maximum time that is needed for the kinematic prediction $T_{\text{p}}^{\max}$ and the optimization $T_{\text{o}}^{\max}$ of a movement with a duration of T_{h}. One assumption is that the time needed for optimization is smaller than the duration of the optimization horizon $T_{\text{s}} < T_{\text{h}}$. This portion of the optimization horizon will later be executed by the robot in one block.

This however can pose problems in dynamic situations if the sampling interval is large (e. g., in the order of seconds). In the worst case, the system would need twice the sampling interval to react to a disturbance. Thus, in order to keep the advantages of MPs to quickly adapt to situation changes, a mechanism is included to interrupt the parallel optimization

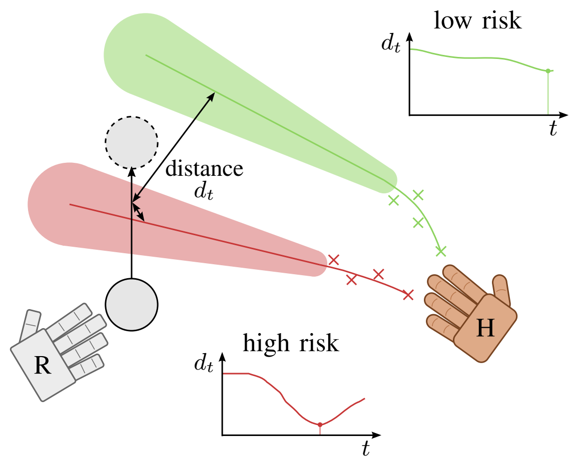

Fig. 4. Risk calculation based on the predicted object and human hand trajectories. The red trajectory leads to a high risk caused by the small distance between human hand and object trajectory. In contrast, the green example of human hand motion is not risky.

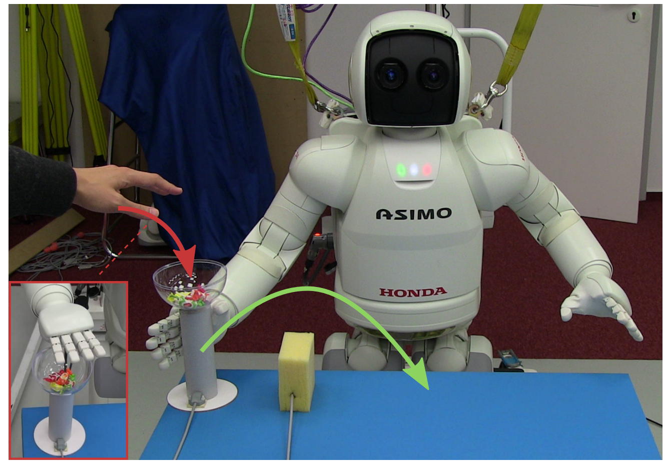

Fig. 5. The experimental setup with the robot standing in front of a blue table. The task is to pick and place the object while avoiding collisions and joint limits. A human can disturb the robot, which has to react with one out of three behaviors (e. g., covering motion, bottom left).

at any time if a disturbance is detected. By this, the reaction time can be reduced to T_{s}.

This delay could still be too long, if the optimization is computationally expensive. Therefore, if a disturbance is detected, the full optimization is bypassed and only the orders of magnitude faster kinematic prediction is performed. The resulting movement that accounts for the disturbance is not optimal, but is available after only the short duration of $T_{\mathrm{p}}^{\mathrm{max}}$. We argue that in most situations in which the robot would need to drastically change its behavior, a faster non-optimal solution is preferred over a delayed optimal solution. After one non-optimal period of T_{s}, the movement continues optimally.

For this interruption mechanism to work, the disturbance needs to be detectable. This could be achieved by continuously comparing the expected situation to the actual one or by a more sophisticated disturbance detection. For the experiment in this paper, the disturbance is the predicted risk that a human interferes with the object which the robot manipulates. A separate risk predictor is implemented that calculates the global risk value ϱ, which has already been mentioned in Section II-B (Equation 10). Thus the risk does not only interrupt the optimization process, but also can change the robot's movement to a qualitatively different strategy. The details are shown in the experiment section, after the following explanation of the risk predictor.

B. Risk predictor

For the experiment, risk is defined as the possibility that a human interferes with the object that the robot should manipulate. To calculate this possibility, the future trajectory of the object (resulting from the previous optimization step) and a predicted trajectory of the human hand is taken into account. For the latter, a simple Kalman filter is used that is continuously updated with the human's hand position. The filter accounts for sensory noise and is able to predict the future hand trajectory by assuming that the hand velocity remains constant. The risk ϱ is then calculated from the following equation:

$$\varrho = \max_{0<t<T_\varrho} \left(1 - \frac{1}{1 + e^{-\alpha(d_t - \beta)}}\right) \tag{12}$$

with T_ϱ being the length of the risk calculation time window and d_t being the Euclidean distance between the predicted object and human hand position at time step t. Parameters α and β are tuning parameters to change the sensitivity. For the experiment, they are set to 20 and 0.3, respectively. Figure 4 illustrates the idea of the calculation. If the hand and object come sufficiently close together at some time during their predicted trajectories, a high risk is implied.

V. EXPERIMENT — "DON'T STEAL MY CANDY"

The purpose of the experiment presented in this section is to visualize two different aspects. Firstly, we want to emphasize the advantages of combining reactive movement generation using MPs and receding horizon movement optimization as such. Secondly, the experiment serves as an example which shows that it is often feasible to fall back to a non-optimal solution that is immediately available rather than accounting for a high risk too late.

Figure 5 shows the experimental setup. The robot is standing in front of a table and the task is to get the object (a cup filled with delicious candy) and put it to the center of the table. While doing this, the robot has to account for self-collisions, collisions with the yellow obstacle, and joint limit avoidance. Furthermore, the robot can be disturbed by a human that tries to get close to the object, which implies *risk* for the robot. To account for the risk, the normal behavior has to be interrupted and according to the situation that the robot is in, one out of three risk avoiding behaviors has to be performed. These are either covering the object with the left or the right hand, or retracting the object from the table when it is already grasped with the right hand.

The object, the obstacle, and the human hand are tracked using a magnetic field-based tracking system. The models

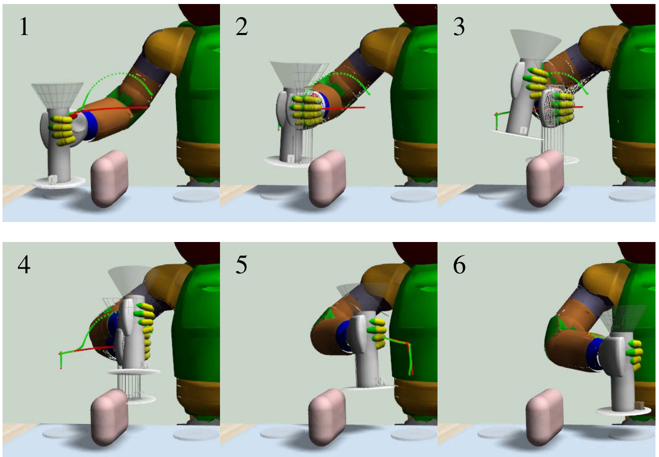

Fig. 6. The image sequence shows the result of the receding horizon optimization (green, solid) compared to the prior movement provided by the MPs (red, wireframe).

TABLE I

OVERVIEW OF THE SKILLS AND MPS USED DURING THE EXPERIMENTS.

	MPs		MPs	Skills
Head	Look at object	Risk right	Prepare covering	Approach cup (right)
Head	Look at disturber	Risk right	Cover cup	Get cup (right)
		Risk right	Resume from covering	Transport cup (right)
Right	Lift arm	Risk right	Retract cup	Put cup (right)
Right	Pre-grasp			Cover cup (right)
Right	Move to cup	Left	Lift arm	Cover cup (left)
Right	Grasp	Left	Cover cup	Head behavior
Right	Release	Left	Retract	Pick and place
Right	Lift over table			Main behavior
Right	Move to center			
Right	Put down			

of the objects and the robot are assumed to be known and therefore the sensor information sufficiently describes the full state. All computations were performed on a standard personal computer (Intel Core i5-2400 3.1 GHz).

According to the description in Section II, the behavior of the robot is defined by a set of MPs and skills. All in all there are 17 linear attractor MPs and 9 skills (CTRNNs), which are listed in Table I. They are defined with the following granularity: The MPs are basic goal-directed movements such as controlling the position of the right hand to reach a target point in the cup's reference frame. Each MP calculates a reasonable goal distance (usually a weighted Euclidean distance to its target) and a responsibility (e. g., right hand grasping is responsible if the object is in a region at the right side of the table). Skills activate the MPs sequentially or in parallel and compose more complex behaviors, such as opening the hand and moving it to a pre-grasp pose, grasping and lifting the object, moving it to the target location, or covering it with the left hand. One particular skill is responsible for the head movement and controls two MPs, which are looking at the object or looking at the disturber in case of a high risk. On the highest level there is one skill called *main behavior* that activates the head movement and the pick and place sequence. The parametrization of the high-level skills is chosen in a way that they switch the activation of lower-level skills not only according to the state information (goal distance, responsibility, ...), but also because of the predicted risk that is directly coupled into the CTRNNs. Thus the risk does not only interrupt the optimization, but also leads to appropriate behaviors.

Figure 6 illustrates the advantage of the receding horizon optimization to transparently modify the movement generated by the MPs. In this example, the risk does not increase, but an obstacle is placed on the table. The simulation shows the predicted (red) and optimized trajectory (green) tracked by the wireframe and solid model, respectively. While the movement in images 1–4 is being carried out, the succeeding movement (5–6) is already being optimized starting from the final (optimal) state in image 4. Thus a smooth movement without discontinuities is achieved. It can be seen that the optimization continues to follow the prior in the case of no violation of the optimization criteria. This is an advantageous behavior similar to the *minimal intervention principle* [21] given the reasonable assumption that the movement generation layer provides a good initial movement.

In Figure 7, the full behavior of the system is shown including avoidance of the obstacle (images 6, 9, 10) and reaction to the predicted risk with three distinct actions (2–4, 7–8, and 11–12). The sampling interval of the receding horizon optimization T_s is set to 40 (equals $1\,\mathrm{s}$ for $\Delta t = 25\,\mathrm{ms}$) and the optimization takes $2\,\mathrm{s}$ of future movement into account ($T_\mathrm{h} = 80$). In case of a detected risk the ongoing optimization will be interrupted and only the kinematic simulation of the next T_s steps is performed. The simulation takes approximately $T_\mathrm{s} \cdot 500\,\mu\mathrm{s} = 20\,\mathrm{ms}$ and thus the prediction cannot start from the current state, but with the state 2 time steps later in order to assure a continuous movement. Therefore, the reaction time of the system after detecting the risk is $50\,\mathrm{ms}$.[2] Together with the reaction, the parallel optimization starts and an interval of $1\,\mathrm{s}$ later, the movement is optimal again.

Overall the robustness of the system results from various factors. First, only stable linear attractor dynamics are used for the low-level MPs. Second, the sequence of MPs is generated by CTRNNs that do not instantaneously switch, but rather blend between MPs. Third, the receding horizon optimization accounts for local adaptation and seamlessly connects optimal trajectories. There is however no guaranteed stability and if the hand-defined top-level sequence does not cover the situation, the robot motion may converge to a local optimum.

VI. CONCLUSION AND OUTLOOK

In this paper a new method for generating continuous robot movements has been presented. It combines the advantages of a hierarchical movement primitive framework and a receding horizon optimal control approach. The MP layer is responsible for deciding which sequence of movements to perform, while the optimization layer can adapt the movement to the actual situation. In case of disturbances that require quick reactions, the optimal control layer is interrupted and the system falls back to the non-optimal movement provided by the MPs.

[2]The average human reaction time to simple stimuli is about $200\,\mathrm{ms}$, but including perception which is neglected here.

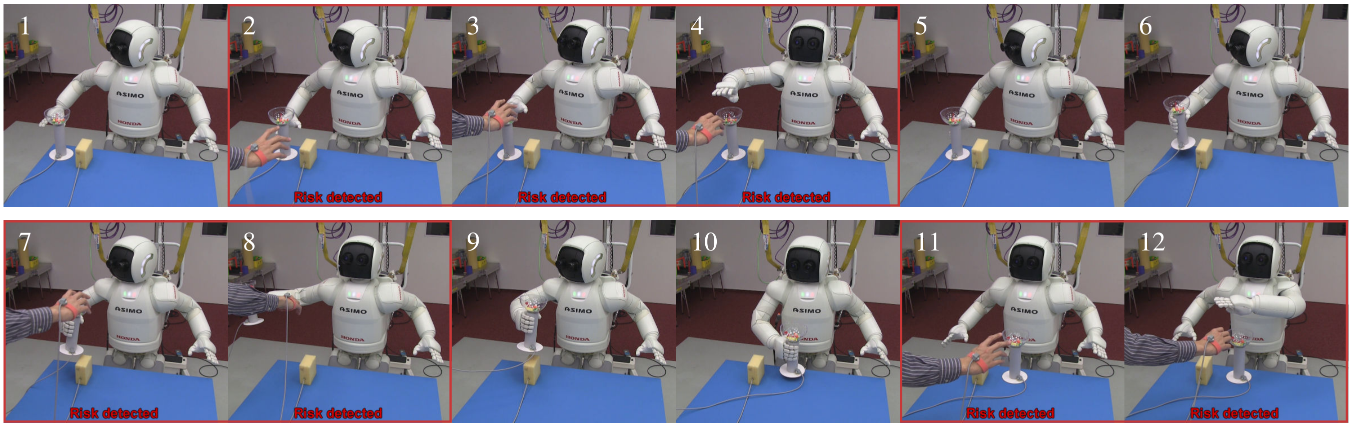

Fig. 7. The humanoid robot performs a pick and place sequence that is continuously optimized for avoiding joint limits, self-collisions, and collisions between the object and the obstacle. Three times (2–4, 7–8, 11–12) a human disturbs the robot by getting close to the object. The system interrupts the optimization and quickly reacts to the disturbances with three different movements according to which is feasible in the respective situation.

The proposed system fulfills four desirable properties: First, it continuously and without interruption optimizes the movement wrt. the current situation. Second, the optimization takes a future time horizon into account and thus can adapt the movement more effectively than the reactive MPs themselves. Third, the system optimizes the full movement that results from an arbitrary sequential or parallel activation of MPs. And fourth, the optimal control layer can be interrupted in situations that require quick reactions and where optimality is only the second concern.

Future work will focus on a closer integration of the two layers.

ACKNOWLEDGMENT

The authors thank Dr. Tobias Luksch for his valuable contributions concerning the formulation and use of movement primitives and recurrent neural networks.

REFERENCES

[1] T. Flash and B. Hochner, "Motor primitives in vertebrates and invertebrates." *Current opinion in neurobiology*, vol. 15, no. 6, pp. 660–666, Dec. 2005.

[2] R. Amit and M. J. Matarić, "Parametric primitives for motor representation and control," in *IEEE International Conference on Robotics and Automation*, vol. 1, Washington DC, 2002, pp. 863–868.

[3] A. Ijspeert, J. Nakanishi, and S. Schaal, "Movement imitation with nonlinear dynamical systems in humanoid robots," in *Proceedings 2002 IEEE International Conference on Robotics and Automation*, vol. 2. IEEE, 2002, pp. 1398–1403.

[4] S. Khansari-zadeh and A. G. Billard, "Imitation learning of Globally Stable Non-Linear Point-to-Point Robot Motions using Nonlinear Programming," in *Proceeding of the 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2010, pp. 2676–2683.

[5] S. Schaal, J. Peters, and J. Nakanishi, "Control, planning, learning, and imitation with dynamic movement primitives," in *Workshop on Bilateral Paradigms on Humans and Humanoids, IEEE International Conference on Intelligent Robots and Systems (IROS 2003)*, 2003, pp. 1–21.

[6] H. Hoffmann, P. Pastor, and S. Schaal, "Movement reproduction and obstacle avoidance with dynamic movement primitives and potential fields," in *Humanoids 2008 - 8th IEEE-RAS International Conference on Humanoid Robots*. IEEE, Dec. 2008, pp. 91–98.

[7] P. Pastor, H. Hoffmann, T. Asfour, and S. Schaal, "Learning and generalization of motor skills by learning from demonstration," in *2009 IEEE International Conference on Robotics and Automation*. Ieee, May 2009, pp. 763–768.

[8] J. Kober, A. Wilhelm, E. Oztop, and J. Peters, "Reinforcement learning to adjust parametrized motor primitives to new situations," *Autonomous Robots*, vol. 33, no. 4, pp. 361–379, Apr. 2012.

[9] A. Ude, A. Gams, T. Asfour, and J. Morimoto, "Task-Specific Generalization of Discrete and Periodic Dynamic Movement Primitives," *IEEE Transactions on Robotics*, vol. 26, no. 5, pp. 800–815, Oct. 2010.

[10] M. Do, P. Azad, T. Asfour, and R. Dillmann, "Imitation of human motion on a humanoid robot using non-linear optimization," in *Humanoid Robots, 2008. Humanoids 2008. 8th IEEE-RAS International Conference on*. IEEE, 2009, pp. 545–552.

[11] M. Mühlig, M. Gienger, and J. J. Steil, "Interactive imitation learning of object movement skills," *Autonomous Robots*, vol. 32, no. 2, pp. 97–114, Dec. 2011.

[12] E. a. Rückert, G. Neumann, M. Toussaint, and W. Maass, "Learned graphical models for probabilistic planning provide a new class of movement primitives." *Frontiers in computational neuroscience*, vol. 6, no. January, p. 97, Jan. 2013.

[13] T. Luksch, M. Gienger, M. Mühlig, and T. Yoshiike, "Adaptive movement sequences and predictive decisions based on hierarchical dynamical systems," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, Oct. 2012, pp. 2082–2088.

[14] ——, "A Dynamical Systems Approach to Adaptive Sequencing of Movement Primitives Dynamical Systems," in *Proceedings of the 7th German Conference on Robotics (ROBOTIK)*, 2012.

[15] Y. Sandamirskaya and G. Schöner, "Serial order in an acting system: A multidimensional dynamic neural fields implementation," in *2010 IEEE 9th International Conference on Development and Learning*. IEEE, Aug. 2010, pp. 251–256.

[16] E. Todorov and W. Li, "A generalized iterative LQG method for locally-optimal feedback control of constrained nonlinear stochastic systems," in *Proceedings of the 2005, American Control Conference, 2005.* IEEE, 2005, pp. 300–306.

[17] M. Toussaint, "Robot trajectory optimization using approximate inference," in *Proceedings of the 26th Annual International Conference on Machine Learning - ICML '09*. New York, New York, USA: ACM Press, 2009, pp. 1–8.

[18] N. Ratliff, M. Zucker, J. A. Bagnell, and S. Srinivasa, "CHOMP: Gradient Optimization Techniques for Efficient Motion Planning," in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2009.

[19] M. Toussaint, M. Gienger, and C. Goerick, "Optimization of sequential attractor-based movement for compact behaviour generation," in *2007 7th IEEE-RAS International Conference on Humanoid Robots*, no. 2. IEEE, Nov. 2007, pp. 122–129.

[20] W. H. Kwon, S. Han, and S. H. Han, *Receding Horizon Control: Model Predictive Control for State Models*, ser. Advanced Textbooks in Control and Signal Processing. Springer, 2005.

[21] E. Todorov and M. I. Jordan, "A Minimal Intervention Principle for Coordinated Movement," in *Advances in Neural Information Processing Systems*, 2002, pp. 27–34.