Constrained Dynamic Movement Primitives for Collision Avoidance in Novel Environments

Seiji Shaw*, Devesh K. Jha‡, Arvind U. Raghunathan‡, Radu Corcodel‡,
Diego Romeres‡, George Konidaris†, and Daniel Nikovski‡

***Abstract*— Dynamic movement primitives are widely used for learning skills that can be demonstrated to a robot by a skilled human or controller. While their generalization capabilities and simple formulation make them very appealing to use, they possess no strong guarantees to satisfy operational safety constraints for a task. We present constrained dynamic movement primitives (CDMPs), which can allow for positional constraint satisfaction in the robot workspace. Our method solves a non-linear optimization to perturb an existing DMP's forcing weights to admit a Zeroing Barrier Function (ZBF), which certifies positional workspace constraint satisfaction. We demonstrate our approach under different positional constraints on the end-effector movement on multiple physical robots, such as obstacle avoidance and workspace limitations.**

I. Introduction

Trajectory generation for manipulators for tasks that require complex, dynamic movements is often difficult and time-consuming. Learning from Demonstration (LfD) offers an efficient approach to teach motor skills to robots [1], [2], since expert demonstrations can be directly used to learn a suitable representation of the movement. However, these skills will need to contend with new operational constraints when executed in new environments, such as collision avoidance to new obstacles. Our goal is to find a way to adapt motor skills learned from demonstration to new environmental constraints.

One of the most popular LfD approaches is the dynamic movement primitive (DMP) [3], [4], a nonlinear dynamical system that can fit expert demonstration trajectories by decoupling a nonlinear forcing function from the nominal attraction behavior. DMPs have been applied to learning a wide range of skills [5], [6], [7], [8], [9]. While they can be reparameterized by their start and goal positions and preserve the qualitative shape of the trajectory, adapting them to new environments with novel operational constraints (see Figure 1 for an example) is not obvious.

For dynamical systems like DMPs, safety can be defined by whether the trajectories of the system remain within a specified 'safety set.' Such a set can be defined by removing obstacles from the set that defines the robot's workspace. Ames et al. [10] introduce Zeroing-Barrier Functions (ZBFs), a way to certify whether a dynamical system is *forward-invariant* in a set, i.e. whether the state of the system is always in that set.

In this paper, we use ZBFs to guarantee that a learned DMP will avoid obstacles and respect workspace limits. We present Constrained DMPs (CDMPs), which adapts a learned DMP to novel collision avoidance constraints introduced by executing the skill in new environments. This is achieved by perturbing the weights of the DMP's learned forcing functions to satisfy positional constraints specified in a ZBF, given by the user. The weight perturbations are computed by forming an nonlinear optimization problem that changes the weights until the conditions for ZBF forward-invariance certification in the safety set is satisfied. The optimization is transcribed as a nonlinear program (NLP) that can be solved using off-the-shelf non-linear program solvers (such as IPOPT [11], [12]). We include demonstrations of succesful CDMP computation on two robots in three different domains where we can successfully compute CDMPs which satisfy operational constraints in the robot workspace. Compared to the past work on incorporating collision avoidance constraints in DMPs [13], [14], [15], [16], we provide guarantees for safety set invariance for the new trajectories, which encompass obstacle-avoidance and novel workspace constraints.

II. Related Work

Learning from demonstration, or imitation learning, is an active area of research in the robot learning community [1]. In general, there are two common behavior representations trained by LfD: a stochastic policy or a dynamical or geometric representation of a concrete trajectory. A case of the latter, DMPs have been widely successful in learning motor skills for robots because they present a simple dynamical system that can encode a wide range of motor skills.

Training stochastic policies often involves behavior cloning, or optimizing a parameterizable policy to match the expert demonstration [2]. Diffusion-based models [17] are a recent development of a behavior model that can scale to high-dimensional output spaces (e.g. images) and have been shown to produce surprisingly robust controllers. The amount of time and computation required to train these models, however, make approaches like these intractable for our problem.

On the other hand, DMPs enjoy wide application in a variety of robot tasks, but are unable to incorporate operational constraints with strong guarantees. Collision avoidance

*Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA. seijis@mit.edu. †Department of Computer Science, Brown University, Providence, RI. gdk@cs.brown.edu. ‡Mitsubishi Electric Research Labs (MERL), Cambridge, MA. {jha,raghunathan,corcodel,romeres,nikovski}@merl.com

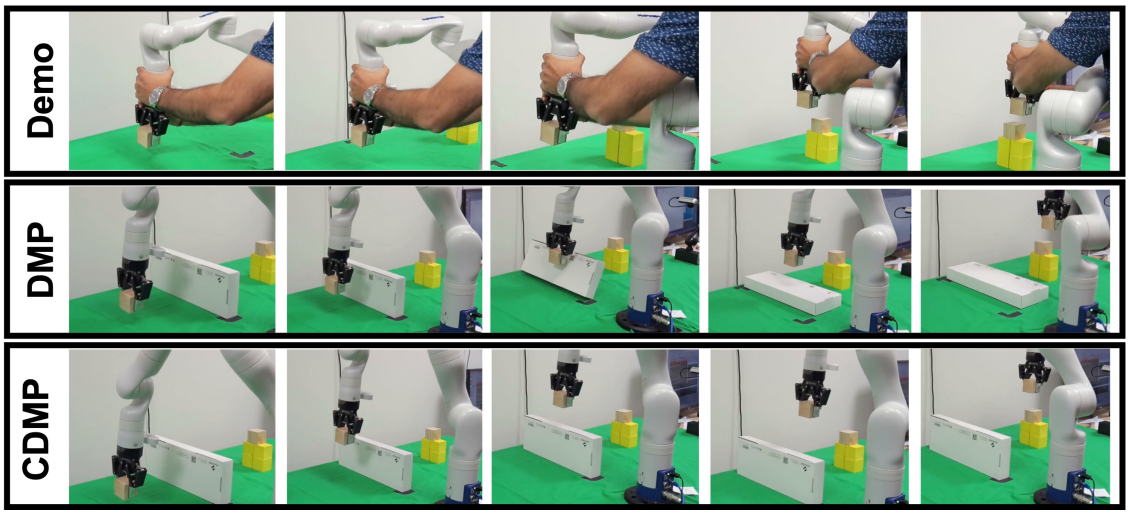


Fig. 1: A simple block stacking environment using the proposed CDMP algorithm. The demonstration (top row) was provided in a collision-free environment. A box obstacle is introduced during execution and a simple DMP collides (middle row) with the obstacle during execution. The proposed CDMP (bottom row) is able to successfully avoid the novel obstacle.

is usually incorporated by combining DMPs with artificial potential fields (or using other similar heuristics) [13], [14]. In general, the technique lacks a formulation that can naturally incorporate any form of positional constraint without the time-consuming experimental tuning processes or risks of local minima that artificial potential field-like approaches possess ([14], Sec. V-C). More recently, there has been some work on using DMPs with sampling based methods to allow collision avoidance [18]. These methods use sampling to find a state from which the DMP can avoid obstacles. In contrast, in the proposed method, we perturb the learned forcing function to satisfy constraints on the robot's workspace.

We also draw inspiration from approaches in safety-critical control that uses optimization-based approaches to constrain the control inputs of a dynamical system to maintain strong guarantees of the system's safety. At the center of these approaches are control Lyapunov functions (CLFs) [19] and control barrier functions (CBFs) [10], [20], [21], which certify convergence and safety set forward-invariance, respectively. In a similar vein, Tedrake et al. [22] compute Lyapunov-stable regions for locally-defined LQR controllers via sum-of-squares optimization. Since a DMP is just a nonlinear dynamical system, we leverage Ames et al.'s zeroing barrier function (ZBF) formalism (which Ames et al. extends to control-affine systems to define CBFs) to certify the forward-invariance of a DMP in a predetermined safety set [10], [20].

A key insight of our work is to perturb the weights of an already-learned DMP to admit an already-existing ZBF so that the DMP is guaranteed to be forward-invariant in a user-defined safety set. There is a large body of work that constrains more general parameterizable dynamical systems via Lyapunov functions [23], [24], [25] and contraction mappings [26]. While these techniques guarantee convergence of the dynamical system, such methods cannot maintain the DMP's ability to generalize motions over changes in the attraction point, a property that our approach preserves. Nomotista et al. [27] leverage a diffeomorphism to map non-convex obstacles into a complex space, where they constrain dynamics of a control system via CBF. Khansari-Zadeh et al. [28] propose a real-time dynamical-reshaping approach for avoidance of convex obstacles. While these approaches work for obstacles that follow their assumptions, it is unclear if such an approach can be modified for avoidance of non-compact sets like the space outside workspace boundaries. Ohnishi et al. [29] consider CBFs for learning agents to avoid more abstract task-specific constraints for agents whose dynamics are learned via reinforcement learning. We solve a task-specific learning-by-demonstration problem instead.

III. Background

In this section, we review the mathematical details of DMPs and ZBFs relevant to our formulation of our CDMP.

A. Dynamic Movement Primitives

DMPs were first introduced by Ijspeert et al. [3], [4]. To remove explicit time dependency, they use a canonical system to keep track of the progress through the learned behavior:

$$\tau \dot{s} = -\alpha_s s \tag{1}$$

where $s = 1$ at the start of DMP execution (and $\alpha_s > 0$) and $\tau > 0$ specifies the rate of progress through the DMP.

To capture attraction behavior, DMPs use a spring-damper system (the transformation system) with an added nonlinear forcing term. Writing the DMP equations as a system of coupled first-order ODEs yields

$$\tau \dot{z} = \alpha_z(\beta_z(g - y) - z) + f(s) \tag{2}$$

$$\tau \dot{y} = z, \tag{3}$$

where g denotes the goal pose. The forcing term is defined as a sum of radial-basis functions

$$f(s) = \frac{\sum_{i=1}^{N} w_i \psi_i(s)}{\sum_{i=1}^{N} \psi_i(s)} \tag{4}$$

$$\psi_i(s) = \exp\left(-h_i(s-c_i)^2\right), \tag{5}$$

where h_i and c_i denote the inverse width and center of the Gaussian basis functions, respectively. The forcing term is learned from the demonstration by solving a locally weighted regression to fit the demonstration provided by an expert.

B. Zeroing Barrier Functions

Since DMPs are formulated as autonomous nonlinear dynamical systems, we can certify them for safety set forward-invariance using zeroing barrier functions [20], [10].

First, let $h(x) : \mathbb{R}^n \to \mathbb{R}$ be a continuous differentiable function. Define set $\mathscr{C}$ to be the following super-level set:

$$\mathscr{C} = \{x \in \mathbb{R}^n : h(x) \geq 0\} \tag{6}$$

$$\partial\mathscr{C} = \{x \in \mathbb{R}^n : h(x) = 0\} \tag{7}$$

$$\text{Int}(\mathscr{C}) = \{x \in \mathbb{R}^n : h(x) > 0\}. \tag{8}$$

Assume we have a nonlinear dynamical system of the form:

$$\dot{x} = f(x). \tag{9}$$

We call $h(x)$ ZBF if the following inequality holds:

$$\dot{h}(x) \geq -\alpha(h(x)), \tag{10}$$

where $\alpha : \mathbb{R} \to \mathbb{R}$ is a class-$\mathscr{K}$ function [10]. Like Ames et al., we consider the specific case

$$\dot{h}(x) \geq -\gamma h(x). \tag{11}$$

with constant $\gamma > 0$. Given a dynamical system (9), a valid ZBF $h(x)$ exists if and only if our dynamical system $\dot{x} = f(x)$ is forward-invariant in the set $\mathscr{C}$, i.e. if $x_0 \in \mathscr{C}$ and $\dot{x} = f(x)$ then:

$$x \in \mathscr{C},\ \forall t \in [0, \infty). \tag{12}$$

ZBFs, given the 'if and only if' relation above, are a general way to think about safety-set invariance for dynamical systems of any kind. They are also amenable to be written as dynamic constraints in a mathematical optimization problem (Section IV-A).

IV. Constrained Dynamic Movement Primitives

We present our formulation of the proposed constrained DMP (CDMP), which we cast as a nonlinear optimization problem.

The main insight behind our proposed formulation of CDMPs is to take an existing DMP with a forcing function learned from an expert trajectory and then minimally change the dynamics governed by the forcing function weights to admit positional constraints induced by a ZBF. We introduce additional parameters that perturb the weights $\{w_i\}$ of the forcing function of the learned DMP. These perturbations are computed by solving an optimization problem, where the ZBF inequality (eq. 11) is added as a constraint.

A. CDMPs as a Nonlinear Optimization Problem

As was explained earlier in Section III-A, the system of equations and the forcing function for DMP could be written and expressed by the following system of equations:

$$\begin{bmatrix} \dot{s} \\ \dot{z} \\ \dot{y} \end{bmatrix} = \frac{1}{\tau} \begin{bmatrix} -\alpha_s s \\ (\alpha_z(\beta_z(g-y) - z) + f(s)) \\ z \end{bmatrix}, \tag{13}$$

where

$$f(s) = \frac{\sum_{i=1}^{N} w_i \psi_i(s)}{\sum_{i=1}^{N} \psi_i(s)} \tag{14}$$

is the forcing function expressed as a weighted sum of radial basis functions, where the weights are learned from the provided expert demonstration and govern the evolution of the DMP trajectory.

We introduce new weight perturbations $\{\zeta_i\}$ so that the DMP with a new forcing function with perturbed weights $\{w_i - \zeta_i\}$ can satisfy the ZBF inequality for forward-invariance in the user-defined safety set. The perturbed forcing function of the new DMP can be expressed as

$$\tilde{f}(s) = \frac{\sum_{i=1}^{N} (w_i - \zeta_i) \psi_i(s)}{\sum_{i=1}^{N} \psi_i(s)}. \tag{15}$$

We compute $\{\zeta_i\}_{i=1,\ldots,N}$ by casting them as decision variables in the following minimization problem:

$$\min_{\zeta_1,\ldots,\zeta_N} \quad \sum_{i=1}^{N} \zeta_i^2, \tag{16}$$

subject to the dynamic constraints

$$\tau \begin{bmatrix} \dot{s} \\ \dot{z} \\ \dot{y} \end{bmatrix} + \begin{bmatrix} \alpha_s s \\ -\left(\alpha_z(\beta_z(g-y) - z) + \tilde{f}(s)\right) \\ -z \end{bmatrix} = \mathbf{0} \tag{17}$$

$$\dot{h}(x) + \gamma h(x) \geq 0 \tag{18}$$

$$x(t_0) = x_0, \tag{19}$$

where the state $x = [s, z, y]^T$. Equation (18) represents the user-specified ZBF that represents the new positional constraints. The intuition behind minimizing these perturbations as a part of the optimization objective is to (1) ensure a relatively smooth trajectory and (2) keep the new trajectory closer to the demonstration trajectory given by the expert. The constraint is enforced at a discrete number of 'collocation points' along the trajectory. While adding more collocation points allows for stricter constraint satisfaction for the physical system, the amount of computation required to solve the problem also sharply increases (Sec. V-B).

The optimization problem represented by Equations (16)-(19) is converted to a finite dimensional discretized problem using PyRoboCOP [12], which is then transcribed into a nonlinear program. The resulting NLP is then solved using IPOPT in the PyRoboCOP environment. We compute new DMP weights $\{w_i - \zeta_i\}$ using the solution perturbations found by PyRoboCOP (see Algorithm 1).

We need to solve (16) every time a novel constraint is introduced in the environment. This can be equivalent to adding new obstacles in the workspace of the robot. In general, the solution to the NLPs can not be done in real-time. However, these trajectories need to be computed only once before operation and executed to complete some repetitive tasks. An online version of the algorithm is left for future work.

Algorithm 1 Compute CDMP

Input: weights $\{w_i\}$, x_{start}, x_{goal}, α (DMP proportional gain), β (DMP damping gain) , $h(\cdot)$ (ZBF), γ (class-$\mathscr{K}$ linear fun. proportional constant), n (number of collocation points)

Output: weights $\{w_i'\}$

Computes perturbation w.r.t. optimization 17-19.

$\{\zeta_i\} \leftarrow$ SolvePerturbations($\{w_i\}$, x_{start}, x_{goal}, α, β, $h(\cdot)$, γ, n)

Compute new weights CDMP that admits $h(\cdot)$

for k in $[1..|\{w_i\}|]$ **do**

  $w_i' \leftarrow w_i - \zeta_i$

end for

return $\{w_i'\}$

B. Perturbation from Original Trajectory

Another important consideration during computation of constrained DMP is to limit the amount of the deviation of the CDMP from the original DMP trajectory. The design of CDMP could be seen as a trade-off between constraint satisfaction and the original forcing function. This trade-off could be controlled using a hyperparameter which constrains the maximum allowable deviation between the original DMP and CDMP, which can be added as another constraint to the trajectory optimization problem. More formally, let us denote the original DMP trajectory as $\{\tilde{y}(t)\}$, $t \in [t_0, t_f]$ and the hyperparameter for the deviation from the original trajectory as ε. Then, the additional constraint could be represented as

$$||y(t) - \tilde{y}(t)||_2 \leq \varepsilon \tag{20}$$

and added to the optimization problem described in equations (17-19). Depending on the value of ε, we can obtain a family of CDMPs that will allow different amount of deviation of the new trajectory from the original DMP trajectory.

C. ZBFs from Signed-Distance Functions

Given an obstacle set Ω with boundary $\partial\Omega$ in $\mathbb{R}^3$, we use the usual definition of a signed distance function (SDFs) $\sigma : \mathbb{R}^3 \to \mathbb{R}$ to the boundary of the obstacle [30]

$$\sigma(x) = \begin{cases} d(x, \partial\Omega), & \text{if } x \notin \Omega, \\ -d(x, \partial\Omega), & \text{if } x \in \Omega, \end{cases}$$

where the distance (or metric) $d(x, \partial\Omega)$ is defined by

$$d(x, \partial\Omega) = \inf_{y \in \partial\Omega} ||x, y||_2.$$

We know that $|\nabla\sigma| = 1$ wherever this gradient is well-defined (and for convex polytopes, this is true whenever $\sigma(x) > 0$). We also know that the implicit surface embedded in $\mathbb{R}^3$ defined by the set of all points $x \in \mathbb{R}^3$ where $\sigma(x) = 0$ is also the boundary of the obstacle, and $\sigma(x) < 0$ whenever x is inside the obstacle. Thus, SDFs fulfill the requirements for a function to designate a safety set (what we denote as $h(x)$ above) and certify that a DMP trajectory is also invariant in the safety set Ω^c.

Given two SDFs σ_1 and σ_2 representing obstacles Ω_1 and Ω_2, we would typically find the SDF of $\Omega_1 \cup \Omega_2$ by taking their minimum $\sigma(p) = \min(\sigma_1(p), \sigma_2(p))$. However, this poses problems of non-differentiability, which is necessary for an optimizer to solve the optimization problem. We resolve this issue by computing a twice-differentiable lower-bound approximation of the minimum function as shown in [31], which is necessary because the ZBF constraint contains a first-order derivative of ZBF $h(x)$:

$$\begin{aligned} \text{smin}(d_1, d_2, k) = &\min(d_1, d_2) \\ &- k\left(\frac{1}{6}\right)\left(\frac{\max(k - |d_1 - d_2|, 0.0)}{k}\right)^3, \end{aligned}$$

where k can be interpreted as a blending radius. With repeated application of the smin operation, we can have a twice-differentiable SDF union of all the obstacles known in the workspace.

We note that closed-form SDFs are sufficient for representing obstacles, because they are differentiable outside the boundary of obstacle set Ω. To implement workspace constraints, we recommend to take the smooth under-approximation of signed-distances functions from bounding planes.

Given our construction of the CDMP, we state the following lemma:

Lemma IV.1. *Let $\dot{x} = g(x)$ be a DMP, $\gamma > 0$, $h : \mathbb{R}^3 \to \mathbb{R}$ be a ZBF, and $\mathscr{C} = \{x \in \mathbb{R}^3 : h(x) > 0\}$. If there exists $\zeta_1, ..., \zeta_n$ such that the CDMP $\dot{x} = \tilde{g}(x)$ admits the ZBF inequality $\dot{h}(x) + \gamma h(x) \geq 0$ along trajectory $x(t)$, then $\tilde{g}$'s trajectory $x(t)$ is forward-invariant in $\mathscr{C}$.*

Proof. Application of Ames et al.'s Proposition B.1 [10]. □

As a corollary of the lemma above, we can claim that if our solver is able to find feasible solution to (16)- (19) to satisfy the ZBF condition, the optimized CDMP trajectory is guaranteed to stay within the safety set where the ZBF certification constraint is enforced at the collocation points.

V. Experiments and Results

We experimentally verify that our algorithm could reliably generate feasible trajectories that avoids obstacles while preserving the flexible learning and reparameterization capabilities of the original DMP. Our experiments are conducted both numerically and on physical robots, using a Python-based interface to IPOPT presented by Raghunathan et al. [12]. A video with all the physical experiments could be seen here.

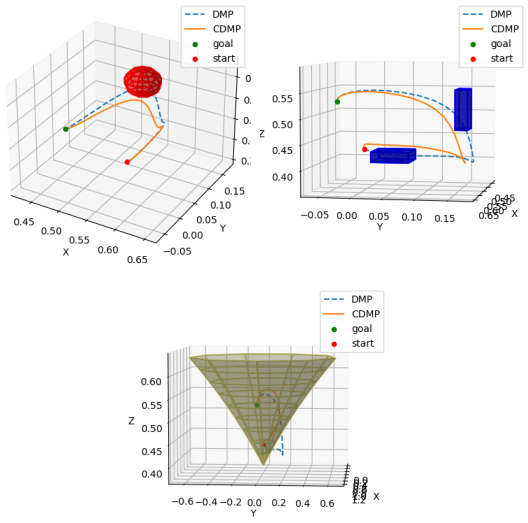


Fig. 2: The orange CDMP trajectory is constrained to lie outside spherical and cuboidal obstacles, and within the conical boundaries.

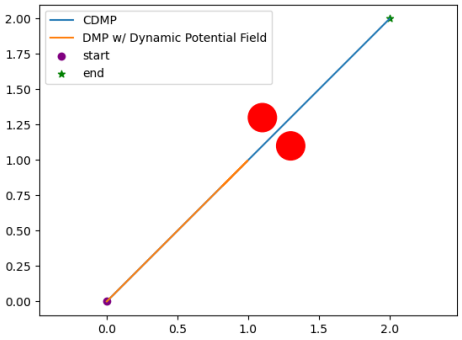


Fig. 3: CDMP and DMP with a dynamic potential field learned from generated straight-line demonstration. The former is able to reach the goal, while the latter is caught in a local minimum (best seen in color).

A. *Evaluation of Obstacle Avoidance*

We first verify that the CDMP algorithm is able to avoid obstacles represented by the ZBF. We perform numerical experiments to verify whether the optimizer can handle a large variety of ZBFs and to study the CDMP's ability to handle narrow passages. We also evaluate the algorithm in real-robot scenarios to verify that the trajectories produced by CDMPs can be easily executed on a real robot.

Our numerical experiments make use of an end-effector position trajectory kinesthetically demonstrated by a human expert in $\mathbb{R}^3$ (see Figure 2). We then insert different kinds of obstacles along the trajectory, including spheres, boxes, and a conically-shaped workspace boundary. As could be seen in Figure 2, the CDMP trajectory is able to satisfy the new constraints in all problem instances.

To evaluate CDMP's ability to handle narrow passages, we compare our method to the dynamic (velocity-dependant) artificial potential field method introduced by Park et al. [14] using a generated straight-line trajectory. As with many potential field methods, it is not difficult to construct an adversarial obstacle configuration that causes the DMP to be trapped in a local minima (see Fig. 3). Even with these obstacles, our CDMP method is able to successfully reach the goal.

Our physical experiments tested CDMP on two different physical robots in three different domains: a wall-hopping domain (see Figure 1), a chess-piece moving domain (see Figure 4), and a whiteboard drawing domain (see Figure 5).

In the wall-hopping and chess-piece moving domains, a user kinesthetically demonstrates the robot a movement in a workspace free of obstacles. New obstacles are then introduced in the environment and are represented using cuboidal bounding volumes. After CDMP computation, the obtained trajectories are unrolled and tracked in both position and velocity using a PD controller. The original DMP (top row) collides with the new obstacle, but CDMPs (bottom row) generates a new non-colliding trajectory.

In the whiteboard experiments, we use a Mitsubishi Electric Assista industrial manipulator arm equipped with a force/torque (F/T) sensor at the wrist to provide feedback for an admittance controller [32], [33] to move the robot in a kinesthetic teaching mode to record expert demonstrations. An expert moves the end-effector to draw shapes for demonstrations (Figure 5). We then add some obstacle sets in the original shapes. The CDMP and admittance controller are able track the CDMP's trajectory that avoids the new obstacles.

B. *Trade-off Between Obstacle Avoidance Fidelity and Computational Complexity*

The computational time for the CDMP depends on the size of the NLP created using the original demonstration trajectory. The size of the problem depends on the number of collocation points where the ZBF constraints are enforced in the problem. The number of collocation points must be strategically chosen to sufficiently cover the path so that a constraint is not violated between two collocation points (in the presence of obstacles). To study the trade-off between CDMP's runtime and number of collocation points chosen for the NLP, we measure the runtime of CDMP computation on the wall-hopping problem (Figure 1) with different number of collocations points.

All experiments were run using unoptimized code on a Lenovo Thinkpad equipped with an Intel i7-8565U processor and 16Gb of RAM. While fewer collocation points result in faster compute times, the trajectory in between collocation points is more likely to intersect with obstacles (Fig. 7). Figure 6 shows a non-linear trend between the number of collocation points and the computational time. This is not surprising since we use an interior-point method (IPM) for a non-convex optimization, where the computational time can

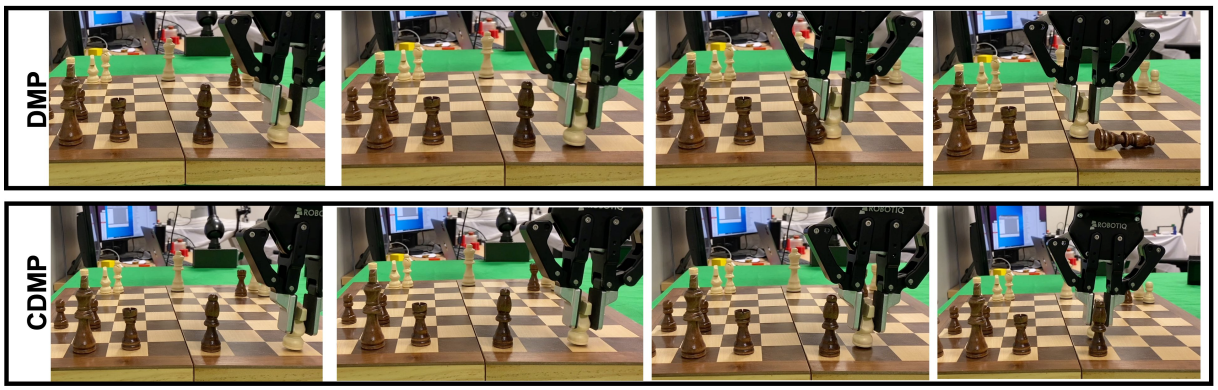
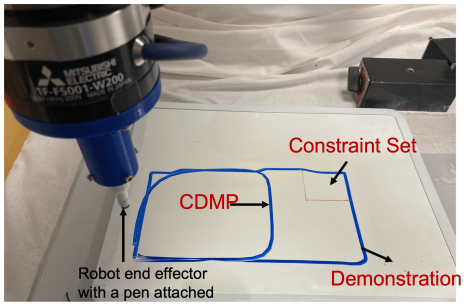
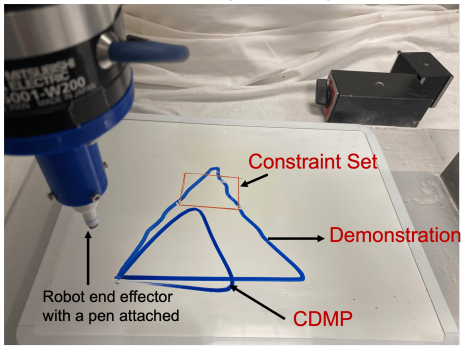


Fig. 4: This shows movement of pieces on a chess board using CDMP. The top row shows that the original DMP collides with an obstacle on the way while the CDMP (bottom row) is able to avoid the obstacle during movement.



(a) Drawing a rectangle.



(b) Drawing a triangle.

Fig. 5: The proposed CDMP algorithm is successfully able to draw demonstrated shapes in the presence of constraints.

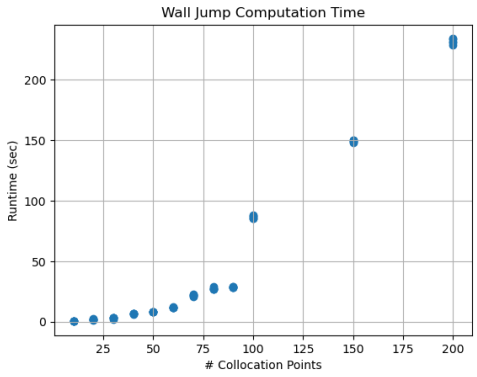


Fig. 6: Experimental runtime of wall-jump problem.

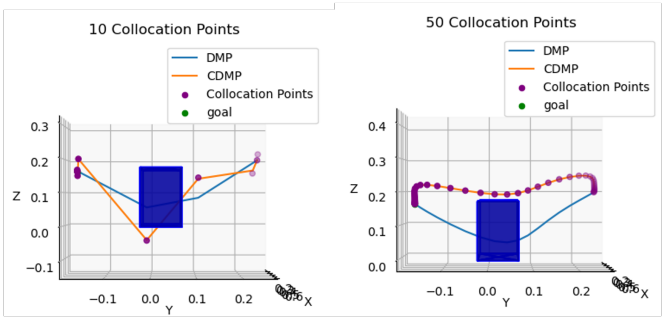


Fig. 7: While $n = 10$ collocation points was much faster to compute (Fig. 6), the trajectory was still in collision with the obstacle since the ZBF condition was too coarsely enforced relative to $n = 50$.

increase sharply with problem size.

C. Preservation of DMP Properties

Since the CDMP algorithm perturbs the learned weights of the DMP forcing function, we must experimentally verify that the CDMP algorithm still preserves the qualitative geometry of the DMP trajectory. More importantly, we verify that the CDMP could preserve the start and goal reparameterization properties of DMP. To evaluate whether CDMPs generalize trajectories to novel start and goal states, we apply the CDMP algorithm to a trajectory taught kinesthetically to a physical robot for a multiple bin-picking task (Figure 8). We observe that when no obstacles are present, the CDMP behaves exactly the same as the DMP (middle trajectory). When obstacles are present, the CDMP will be perturbed to avoid the obstacle (right trajectory).

In both numerical and physical robot experiments (Figures 2 and 5), the generated CDMP trajectory still captures much of trajectory defined by the DMP. For the numerical task learned from the bin-picking expert demonstration, we see that the overall trajectory inverts and scales as the original DMP would, but also perturbs when necessary to avoid obstacles.

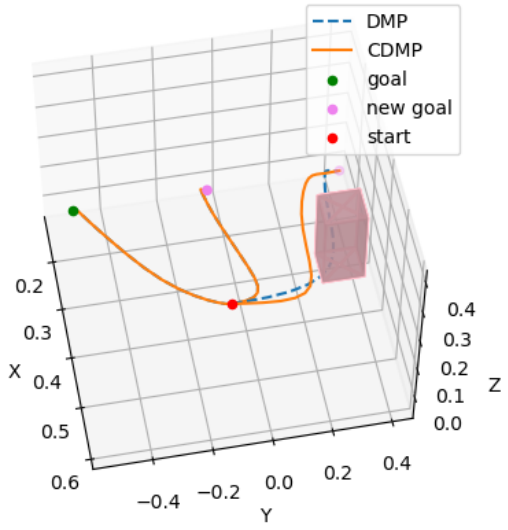


Fig. 8: The left-most trajectory is the unchanged DMP trajectory for a bin picking task. The middle and rightmost trajectories have reparameterized goals.

VI. Conclusion and Future Work

Simplifying motion generation for robots performing repetitive tasks in changing environments is still a significant challenge for robots. We propose a method for incorporating operational constraints while generating trajectories using demonstrations. Our method, constrained dynamic movement primitives, can incorporate task and environmental constraints when generalizing to novel conditions. The proposed CDMP was demonstrated on several examples for collision avoidance in the presence of obstacles of different shapes and sizes. We also confirm that the CDMP still preserves the qualitative geometry of the trajectory, and behaves identically to the DMP when no obstacles are present.

Even though ZBFs can, in theory, include complex operational constraints like self-collision, joint limits, etc., we need computationally efficient and smooth barrier function representation of these constraints to be able to compute safe trajectories. In the future, we would like to use the proposed formulation to consider more constraints past end-effector safety sets that can be more easily expressed in different parameterizations of the robot configuration space, such as joint limits and self-collision avoidance. These constraints result in a complex, constrained optimization problem which needs additional, non-trivial work to find a suitable zeroing barrier function. Another limitation is the computational efficiency with respect to the number of collocation points along the trajectory. While fewer collocation points reduce the amount of computation time, more of the trajectory will be 'exposed' to collide with an obstacle. A possible improvement is to make better use of warm-starting, which are current solver (IPOPT) does not leverage.

Acknowledgements

We thank Nadia Figueroa for her introducing us to the huge body of dynamical systems for learning from demonstration literature. Disclosure: George Konidaris is the Chief Roboticist of Realtime Robotics, a robotics company that produces a specialized motion planning processor.

References

[1] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and Autonomous Systems*, vol. 57, no. 5, pp. 469–483, 2009.

[2] H. Ravichandar, A. S. Polydoros, S. Chernova, and A. Billard, "Recent advances in robot learning from demonstration," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 3, pp. 297–330, 2020.

[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 (Cat. No.02CH37292)*, vol. 2, 2002, pp. 1398–1403 vol.2.

[4] S. Schaal, *Dynamic Movement Primitives-A Framework for Motor Control in Humans and Humanoid Robotics*. Tokyo: Springer Tokyo, 2006, pp. 261–280. [Online]. Available: https://doi.org/10.1007/4-431-31381-8_23

[5] M. Saveriano, F. J. Abu-Dakka, A. Kramberger, and L. Peternel, "Dynamic movement primitives in robotics: A tutorial survey," *arXiv preprint arXiv:2102.03861*, 2021.

[6] K. Muelling, J. Kober, and J. Peters, "Learning table tennis with a mixture of motor primitives," in *2010 10th IEEE-RAS International Conference on Humanoid Robots*, 2010, pp. 411–416.

[7] K. Mülling, J. Kober, O. Kroemer, and J. Peters, "Learning to select and generalize striking movements in robot table tennis," *The International Journal of Robotics Research*, vol. 32, no. 3, pp. 263–279, 2013.

[8] D. K. Jha, D. Romeres, W. Yerazunis, and D. Nikovski, "Imitation and supervised learning of compliance for robotic assembly," in *2022 European Control Conference*, 2022, pp. 1882–1889.

[9] M. Sharma, K. Zhang, and O. Kroemer, "Learning semantic embedding spaces for slicing vegetables," *arXiv preprint arXiv:1904.00303*, 2019.

[10] A. D. Ames, X. Xu, J. W. Grizzle, and P. Tabuada, "Control barrier function based quadratic programs for safety critical systems," *IEEE Transactions on Automatic Control*, vol. 62, no. 8, pp. 3861–3876, 2016.

[11] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical programming*, vol. 106, no. 1, pp. 25–57, 2006.

[12] A. U. Raghunathan, D. K. Jha, and D. Romeres, "Pyrobocop: Python-based robotic control 'i&' optimization package for manipulation," in *2022 International Conference on Robotics and Automation*, 2022, pp. 985–991.

[13] A. Rai, F. Meier, A. Ijspeert, and S. Schaal, "Learning coupling terms for obstacle avoidance," in *2014 IEEE-RAS International Conference on Humanoid Robots*. IEEE, 2014, pp. 512–518.

[14] D.-H. Park, 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, 2008, pp. 91–98.

[15] H. Tan, E. Erdemir, K. Kawamura, and Q. Du, "A potential field method-based extension of the dynamic movement primitive algorithm for imitation learning with obstacle avoidance," in *2011 IEEE International Conference on Mechatronics and Automation*. IEEE, 2011, pp. 525–530.

[16] F. Stulp, E. Oztop, P. Pastor, M. Beetz, and S. Schaal, "Compact models of motor primitive variations for predictable reaching and obstacle avoidance," in *2009 9th IEEE-RAS International Conference on Humanoid Robots*. IEEE, 2009, pp. 589–595.

[17] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. C. Burchfiel, and S. Song, "Diffusion Policy: Visuomotor Policy Learning via Action Diffusion," in *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, July 2023.

[18] S. Sobti, R. Shome, S. Chaudhuri, and L. E. Kavraki, "A sampling-based motion planning framework for complex motor actions," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2021, pp. 6928–6934.

[19] A. D. Ames, K. Galloway, K. Sreenath, and J. W. Grizzle, "Rapidly exponentially stabilizing control lyapunov functions and hybrid zero dynamics," *IEEE Transactions on Automatic Control*, vol. 59, no. 4, pp. 876–891, 2014.

[20] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, "Control barrier functions: Theory and applications," in *2019 18th European control conference*. IEEE, 2019, pp. 3420–3431.

[21] M. Z. Romdlony and B. Jayawardhana, "Uniting control lyapunov and control barrier functions," in *53rd IEEE Conference on Decision and Control*. IEEE, 2014, pp. 2293–2298.

[22] R. Tedrake, I. R. Manchester, M. Tobenkin, and J. W. Roberts, "Lqr-trees: Feedback motion planning via sums-of-squares verification," *The International Journal of Robotics Research*, vol. 29, no. 8, pp. 1038–1052, 2010.

[23] S. M. Khansari-Zadeh and A. Billard, "Learning stable nonlinear dynamical systems with gaussian mixture models," *IEEE Transactions on Robotics*, vol. 27, no. 5, pp. 943–957, 2011.

[24] ——, "Learning control lyapunov function to ensure stability of dynamical system-based robot reaching motions," *Robotics and Autonomous Systems*, vol. 62, no. 6, pp. 752–765, 2014.

[25] N. Figueroa and A. Billard, "A physically-consistent bayesian non-parametric mixture model for dynamical system learning," in *Proceedings of The 2nd Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, vol. 87. PMLR, 29–31 Oct 2018, pp. 927–946.

[26] H. Ravichandar, I. Salehi, and A. Dani, "Learning partially contracting dynamical systems from demonstrations," in *Proceedings of the 1st Annual Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, vol. 78. PMLR, 13–15 Nov 2017, pp. 369–378.

[27] G. Notomista and M. Saveriano, "Safety of dynamical systems with multiple non-convex unsafe sets using control barrier functions," *IEEE Control Systems Letters*, vol. 6, pp. 1136–1141, 2021.

[28] "A dynamical system approach to realtime obstacle avoidance," no. 4, pp. 433–454, 2012.

[29] M. Ohnishi, G. Notomista, M. Sugiyama, and M. Egerstedt, "Constraint learning for control tasks with limited duration barrier functions," *Automatica*, vol. 127, p. 109504, 2021.

[30] M. W. Jones, J. A. Baerentzen, and M. Sramek, "3d distance fields: A survey of techniques and applications," *IEEE Transactions on visualization and Computer Graphics*, vol. 12, no. 4, pp. 581–599, 2006.

[31] I. Quilez, "3d distance functions." [Online]. Available: https://iquilezles.org/www/articles/distfunctions/distfunctions.htm

[32] D. K. Jha, D. Romeres, S. Jain, W. Yerazunis, and D. Nikovski, "Design of adaptive compliance controllers for safe robotic assembly," in *2023 European Control Conference*, 2023, pp. 1–8.

[33] D. K. Jha, S. Jain, D. Romeres, W. Yerazunis, and D. Nikovski, "Generalizable human-robot collaborative assembly using imitation learning and force control," in *2023 European Control Conference*, 2023, pp. 1–8.