Task-dependent Distribution and Constrained Optimization of Via-points for Smooth Robot Motions

ChangHyun Sung[1], Manuel Mühlig[2], Michael Gienger[2] and Yoji Uno[1]

Abstract— **This paper presents an effective method for planning and optimizing robot motions in joint space using via-points. The via-point formulation allows for a sparse movement representation which is inherently smooth according to a minimum jerk criterion. In this research we focus on two aspects. First, we present an initialization method to find a feasible number of via-points and their distribution in time. This initialization takes the difficulty of the planning problem into account and finds a tradeoff between representation complexity (number of via-points) and optimization time. Second, we propose a two-step optimization approach to improve the via-point parameters to satisfy constraints over the whole motion duration. We show that our methods generate feasible and smooth robot motions with a high success rate for tasks of varying complexity. The methods are evaluated in simulation using four different scenarios with two different robots, a 7-DOF robot arm and a 16-DOF humanoid upper body. Additionally, the experiments with the robot arm are shown in a real-world experiment.**

I. INTRODUCTION

With an increasing complexity of both, robots and the environment in which they act, it becomes more and more difficult to efficiently generate smooth robot motions. This is called the problem of motion planning which refers to finding a movement trajectory for a many-degrees-of-freedom robot to reach a certain goal. Additional to reaching the goal, a variety of constraints, such as collision avoidance and the robot's joint limitations have to be considered over the whole motion duration.

One successful way of tackling the problem is to parametrize the trajectory and to formulate motion planning as an optimization problem. For example, the authors of [1] propose an optimization scheme for trajectories that are represented as a sequence of task space attractors. Another approach is the work presented in [2] where the trajectory is represented as uniformly distributed waypoints that are optimized with their proposed method of Covariant Hamiltonian Optimization for Motion Planning (CHOMP). The same trajectory representation is used in [3] that proposed Stochastic Trajectory Optimization for Motion Planning (STOMP) that shows an improvement of the success rate of CHOMP. To efficiently calculate an optimal trajectory, all of the mentioned methods apply gradient-based unconstrained optimization. This however does not guarantee satisfying the constraint conditions, since constraints are only considered as a part of a cost function.

To overcome this problem, constrained optimization has potential in dealing with constraint conditions over the whole motion duration. Generally, constraints such as collision avoidance, balancing, or joint limit avoidance are considered as inequality constraints, while the conditions for reaching a target are represented by equality constraints. Sequential Quadratic Programming (SQP) [4] has often been applied to resolve the problem of motion planning for humanoid robots, which typically have many degrees of freedom [5]. It has been shown that SQP is able to handle different constraints, such as motions with many physical contacts [6]. Recently, Quadratic Programming (QP) formulations even have been used for real-time constrained optimization for planning humanoid motions in task-space [7].

When applying such gradient-based constrained optimization methods, the initialization of the trajectory is an important topic. Typically a two-stage approach for motion planning is proposed, such as in [8]. They initialize the trajectory using a constrained path planning algorithm and optimize the planned trajectory for smoothness and feasibility using SQP. The computational cost of their optimization however is very high as every timestep of the initial trajectory is taken into account. Recently, also the authors of [9] emphasized the importance of initialization in the optimization process. They manually select a fixed number of waypoints and prepare multiple initial trajectories in form of straight lines between start, waypoint, and goal. The obvious drawback of this approach is that the number of waypoints has to be designed and it might be required to generate more complex initial trajectories using more waypoints if the task complexity is higher. Another approach of initializing the trajectory is the use of sampling-based planning methods [10], [11]. They have for example been applied to planning humanoid motions such as dual-arm manipulation tasks [12]. Additionally, it is possible to plan smooth trajectories by employing postprocessing approaches [13]. The referenced work also mentions the importance of an initial path to avoid convergence to a suboptimal solution.

The paper at hand is based on our earlier works [14], [15] in which trajectories are parametrized using via-points. The via-point representation has its origin in the analysis of human motor skills, such as reaching movements of the human arm. It has been found that dynamic movements of a human can be expressed by assigning a number of via-points and calculating joint trajectories using optimization [16], [17].

[1]ChangHyun Sung and Yoji Uno are with the Department of Mechanical Science and Engineering, Graduate School of Engineering, Nagoya University, Furo-cho, Chikusa-ku, Nagoya-shi, 464-8603, Japan, {s_changhyun, uno}@nuem.nagoya-u.ac.jp

[2]Manuel Mühlig and Michael Gienger are with the Honda Research Institute Europe GmbH, Carl-Legien-Strasse 30, 63073, Offenbach/Main, Germany, {manuel.muehlig, michael.gienger}@honda-ri.de

Additionally, via-points showed to be useful for transferring motions from a human demonstration to a redundant robot manipulator for performing skillful tasks [18]. In motion planning for robots, a via-point representation can be used for assigning sufficient conditions to satisfy constraint conditions of the movement. An advantage of the via-point representation is that via-points, as an optimization parameter, are explicitly associated with some physical conditions. Indeed, the via-point parameters consist of the position and velocity in joint space at a specific timestep. Furthermore, in our work, via-points are connected using fifth-order polynomials that describe a minimum jerk criterion. Therefore, the smoothness of the overall trajectory is always guaranteed. In our previous work, we applied optimization methods to plan whole-body motions that fulfill various constraints. In that case, the specification of the via-points and deciding for an appropriate number of via-points are essential problems. However, only a fixed number of via-points was used and it was assumed that the timings of the via-points were appropriately given.

In this research, we want to improve our past work by two aspects. First, we consider the estimation of the required number of via-points and their timings in an initialization procedure. We propose an initialization method that can decide for an appropriate number of via-points and their timings depending on the complexity of the task. Second, we sequentially solve a two-step gradient-based constrained optimization problem. In a first optimization step, inequality constraints are considered as part of the cost function, while in the second step they are represented as hard constraints. It is shown that this approach can lead to better solutions in complex situations than constrained optimization alone as it can better overcome local minima. We apply our motion planning framework to plan reaching tasks with a 7-DOF robot arm and bi-manual reaching tasks for a 16-DOF humanoid upper body respectively in two scenarios of different complexity. It is shown that the proposed methods yield a high success rate for each scenario compared to an initialization using straight-line trajectories.

II. MOVEMENT GENERATION USING VIA-POINTS

The assumption for the underlying movement representation is that it is sufficient that the robot's movement passes through certain points to achieve a certain task under various constraints. Thus, the full movement of the robot is described by these important points, which we call via-points. The problem of generating feasible robot motions then reduces to finding the parameters of these via-points.

A. Via-points formulation

When controlling robots, it is often desired to generate smooth and continuous motions. For this, minimizing jerk has been proven to be a suitable criterion [19]. Thus, in this work, the robot's joint angle trajectory is expressed by a minimum-jerk trajectory that has to pass through certain via-points. If the number of via-points is m, the state of the via-points is defined as a vector of joint angles $\boldsymbol{q}_{\text{via},i}$ and angular velocities $\dot{\boldsymbol{q}}_{\text{via},i}$ at the time $t_{\text{via},i}$ (for $i = 1, \cdots, m$). The joint space trajectory generated with m via-points is denoted as the vector-valued function $\boldsymbol{q}_m(t)$. For $t_{\text{via},1} < \cdots, < t_{\text{via},m}$, the joint space trajectory has to satisfy the following conditions:

$$\begin{bmatrix} \boldsymbol{q}_m(t_{\text{via},1}) - \boldsymbol{q}_{\text{via},1} \\ \underline{\dot{\boldsymbol{q}}_m(t_{\text{via},1}) - \dot{\boldsymbol{q}}_{\text{via},1}} \\ \vdots \\ \boldsymbol{q}_m(t_{\text{via},m}) - \boldsymbol{q}_{\text{via},m} \\ \underline{\dot{\boldsymbol{q}}_m(t_{\text{via},m}) - \dot{\boldsymbol{q}}_{\text{via},m}} \end{bmatrix} = \boldsymbol{0} \quad . \tag{1}$$

Using the Lagrange multipliers $\boldsymbol{\pi}_i$ and $\boldsymbol{\lambda}_i$ (for $i = 1, \cdots, m$), we can formulate the minimum jerk trajectory with m via-points. The derivations of the equations are described in detail in [14] and left out here for brevity. The minimum jerk trajectory is defined as:

$$\boldsymbol{q}_m(t) = \begin{cases} \boldsymbol{a}_5 t^5 + \boldsymbol{a}_4 t^4 + \boldsymbol{a}_3 t^3 + \boldsymbol{a}_2 t^2 + \boldsymbol{a}_1 t + \boldsymbol{a}_0 & \\ \qquad\qquad (0 \le t \le t_{\text{via},1}) & \\ \boldsymbol{a}_5 t^5 + \boldsymbol{a}_4 t^4 + \boldsymbol{a}_3 t^3 + \boldsymbol{a}_2 t^2 + \boldsymbol{a}_1 t + \boldsymbol{a}_0 & \\ + \boldsymbol{f}_1(t) \qquad\qquad (t_{\text{via},1} \le t \le t_{\text{via},2}) & \\ \qquad\qquad \vdots & \\ \boldsymbol{a}_5 t^5 + \boldsymbol{a}_4 t^4 + \boldsymbol{a}_3 t^3 + \boldsymbol{a}_2 t^2 + \boldsymbol{a}_1 t + \boldsymbol{a}_0 & \\ + \boldsymbol{f}_1(t) + \cdots + \boldsymbol{f}_m(t) \qquad (t_{\text{via},m} \le t \le t_{\text{f}}) & \end{cases}$$

$$\boldsymbol{f}_i(t) = \underline{\boldsymbol{\lambda}_i (t - t_{\text{via},i})^4 / 24} + \boldsymbol{\pi}_i (t - t_{\text{via},i})^5 / 120 \ , \tag{2}$$

where t_{f} is the overall movement time, which we assume is appropriately given as a multiple of the sampling time Δt. If the number of degrees of freedom of the robot is n, each coefficient vector of the polynomial has n dimensions, respectively.

For the initialization algorithm 1, later described in Section III, we additionally use another formulation without joint velocities. The formulas can be easily obtained by leaving out the underlined terms in Eqs. (1) and (2).

The polynomial in Eq. (2) has $n(2m + 6)$ coefficients. We can determine the coefficients using the $6n$ conditions of the initial and final state and the $2mn$ conditions of joint angle and angular velocity at the via-points (Eq. (1)). Since we set the current state of the robot as the initial state and velocity and acceleration at the final state as zero, the motion planning problem is to determine the $n(2m + 1)$ conditions to fully describe a $\boldsymbol{q}_m(t)$ that achieves the desired motion task. Before this however it is required to determine the appropriate number of via-points m and their timings $\boldsymbol{t}_{\text{via}} = [t_{\text{via},1} \cdots t_{\text{via},m}]$.

B. Problem definition

The motion planning framework using via-points is composed of two phases: initialization and optimization. In the initialization phase, the appropriate values of m and $\boldsymbol{t}_{\text{via}}$ are automatically decided and the via-point parameters are initialized for efficient optimization. Then in the second phase, the coefficients of the min-jerk polynomial are optimized for

satisfying certain constraints. The set of parameters to be optimized are described as $\boldsymbol{X}_{\text{opt}}$:

$$\begin{aligned} \boldsymbol{X}_{\text{opt}} &= \left[\boldsymbol{Q}_{\text{via}}\ \dot{\boldsymbol{Q}}_{\text{via}}\ \boldsymbol{q}_{\text{f}}\right] \\ \boldsymbol{Q}_{\text{via}} &= [\boldsymbol{q}_{\text{via},1}\ \cdots\ \boldsymbol{q}_{\text{via},m}] \\ \dot{\boldsymbol{Q}}_{\text{via}} &= [\dot{\boldsymbol{q}}_{\text{via},1}\ \cdots\ \dot{\boldsymbol{q}}_{\text{via},m}] \quad , \end{aligned} \tag{3}$$

where $\boldsymbol{q}_{\text{f}}$ is the joint angle vector at the final state.

In this paper, we apply constrained optimization for planning safe robot motions. There are a number of constraints that have to be satisfied throughout the movement. They are considered in terms of inequality constraints for collision avoidance (including self-collisions) and joint range limits. To obtain the collision cost $c_{\text{col}}(\boldsymbol{q})$, we define a collision model using rigid primitive shapes such as capsules or spheres. Their positions and orientations are fully described by the given joint angles $\boldsymbol{q}$. The value of $c_{\text{col}}(\boldsymbol{q})$ is then calculated based on the closest point distance between specific collision pairs. For a detailed description about the calculations of the collision cost and its gradient see our earlier work [1]. Also described there is the cost for avoiding joint range limitations $c_{\text{joint}}(\boldsymbol{q})$ which is expressed as the weighted squared sum of deviations between the current joint angle and the joint's center position. In addition to these inequality constraints, the conditions for target achievement in task space, such as reaching a certain position with the end-effector, are represented as equality constraints.

III. INITIALIZATION

We propose an initialization method that finds the number of via-points considering the complexity of the task. The initialization method is composed of two steps. First, a feasible joint space trajectory is found using sampling-based motion planning. Then, we incrementally add new via-points along the planned path until a feasible or close-to-feasible trajectory is found. The result is the number of via-points m, their timings and a set of via-point parameters $\boldsymbol{X}_{\text{ini}}$ that are used as initialization for the subsequent optimization in Section IV.

A. Path planning

At first, path planning is used to find a valid sequence of joint space states between the initial posture $\boldsymbol{q}_{\text{s}}$ and final posture $\boldsymbol{q}_{\text{f}}$. While $\boldsymbol{q}_{\text{s}}$ is assumed to be the robot's current posture, a posture $\boldsymbol{q}_{\text{f}}$ has to be found that satisfies the target constraints such as reaching with the end-effector for a certain task space position. To get $\boldsymbol{q}_{\text{f}}$ we iteratively apply inverse kinematics from random initial postures until a valid posture has been found. Once the two postures are known, we apply sampling-based path planning in form of RRT-CONNECT [20]. Once the method finds a solution, it returns a sequence of valid joint space states whose distances are below a certain threshold and which connect start and final state. These states are then re-interpolated to obtain a sequence $\boldsymbol{Q}_{\text{pp}} = [\boldsymbol{q}_1^{\text{pp}} \cdots \boldsymbol{q}_T^{\text{pp}}]$ with $T = (t_{\text{f}}/\Delta t + 1)$ valid joint space states. These states serve as possible candidates for the timing and the position of the via-points in the second initialization step.

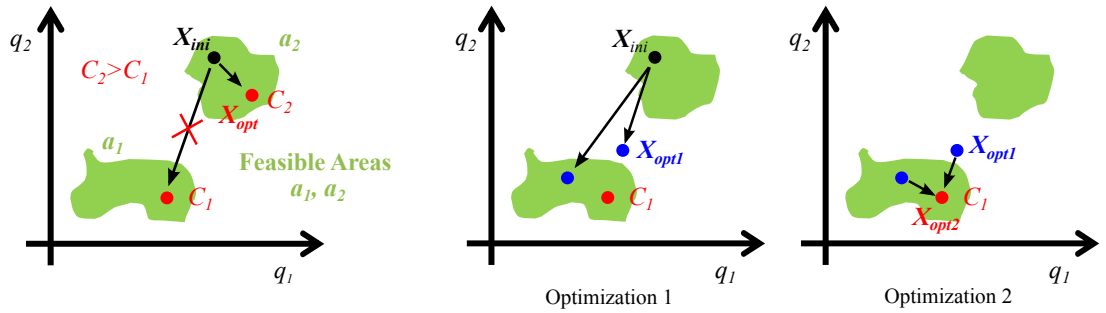
Algorithm 1 Initialization and task-dependent distribution of via-points

Output:
m: number of via-points
$\boldsymbol{t}_{\text{via}} = [t_{\text{via},1} \cdots t_{\text{via},m}]$: via-point timings
$\boldsymbol{Q}_{\text{via}}$: joint angles at the via-points
$\dot{\boldsymbol{Q}}_{\text{via}}$: joint angular velocities at the via-points
$\boldsymbol{X}_{\text{ini}}$: initial parameters for optimization
Variables:
i: timestep
Δt: sampling time
T: total number of timesteps
$\boldsymbol{q}_{\text{s}}, \boldsymbol{q}_{\text{f}}$: joint angles at the initial and final state
$\boldsymbol{q}_m(t)$: minimum jerk trajectory with m via-points
$\boldsymbol{Q}_{\text{pp}} = [\boldsymbol{q}_1^{\text{pp}} \cdots \boldsymbol{q}_T^{\text{pp}}]$: path planning trajectory
$\boldsymbol{Q}_{\text{via}}^*$: temporary via-points
$C_{\text{ini}}(m)$: initialization cost (either Eq. (4) or (5))
$\boldsymbol{C}_{\text{eval}} = [C_1^{\text{eval}} \cdots C_T^{\text{eval}}]$: evaluation vector
tol: tolerance
Algorithm:
$m \Leftarrow 0$
$\boldsymbol{q}_{\text{s}}, \boldsymbol{q}_{\text{f}} \Leftarrow \boldsymbol{q}_1^{\text{pp}}, \boldsymbol{q}_T^{\text{pp}}$
while $C_{\text{ini}}(m) > tol$ **do**
  $m \Leftarrow m + 1$
  for $i = 1$, $i \leq T$ **do**
    $\boldsymbol{Q}_{\text{via}}^* \Leftarrow \left[\boldsymbol{Q}_{\text{via}}\ \boldsymbol{q}_i^{\text{pp}}\right]$
    $C_i^{\text{eval}} = C_{\text{ini}}(m)$ for $\boldsymbol{Q}_{\text{via}}^*$ w/o vel. (Eq. (2) w/o $\dot{\sim}$)
  end for
  $i \Leftarrow \arg\min_{i \in [1,T]} \left(\boldsymbol{C}_{\text{eval}}\right)$
  $\boldsymbol{t}_{\text{via}} \Leftarrow [\boldsymbol{t}_{\text{via}}\ \ i \cdot \Delta t]$
  sort $\boldsymbol{t}_{\text{via}}$ ascending
  for $j = 1$, $j \leq m$ **do**
    $\boldsymbol{Q}_{\text{via}} \Leftarrow \left[\boldsymbol{Q}_{\text{via}}\ \boldsymbol{q}_k^{\text{pp}}\right]$ with $k = \left(t_{\text{via},j}/\Delta t + 1\right)$
  end for
  $\dot{\boldsymbol{Q}}_{\text{via}} \Leftarrow \arg\min \left(C_{\text{ini}}(m)\right)$
end while
keep m, $\boldsymbol{t}_{\text{via}}$
$\boldsymbol{X}_{\text{ini}} \Leftarrow \left[\boldsymbol{Q}_{\text{via}}\ \dot{\boldsymbol{Q}}_{\text{via}}\ \boldsymbol{q}_{\text{f}}\right]$

B. Distribution of via-points

In the second step, we determine the via-points using the result of the path planning $\boldsymbol{Q}_{\text{pp}}$. The method to find the initial via-point parameters $\boldsymbol{X}_{\text{ini}}$ is described in Alg. 1. The idea behind the algorithm is to incrementally add via-points at positions of the initial path that minimize, what we call, the initialization cost.

As initialization cost, we investigated two different cost functions: C_{ini1} and C_{ini2}. The strategy pursued with the first initialization cost is to find a minimal number of via-points with which the planned path $\boldsymbol{Q}_{\text{pp}}$ can be matched. Since the path planning result already considers the constraint conditions, an initial joint trajectory $\boldsymbol{q}_m(t)$ is also assumed



(a) Single optimization

(b) Sequential optimization

Fig. 1. Conceptual sketch for single and sequential optimization. Axes q_1 and q_2 are assumed to be via-point parameters. C_1 and C_2 are the local minima in each feasible (i. e., constraints are fulfilled) area a_1 and a_2. Applying only constrained optimization (a) from the initialization $\boldsymbol{X}_{\text{ini}}$ cannot traverse to the feasible area a_1 which contains the smaller local minimum C_1. In contrast, when applying the sequential optimization (b), the first optimization step can move towards the better local minimum, which allows the constrained optimization to reach it.

to be feasible by closely following $\boldsymbol{Q}_{\text{pp}}$. The cost function C_{ini1} is expressed as

$$C_{\text{ini1}}(m) = \sum_{i=1}^{T} \left\| \boldsymbol{q}_m(i \cdot \Delta t) - \boldsymbol{q}_i^{\text{pp}} \right\|_2 \quad . \tag{4}$$

It is guaranteed that cost function C_{ini1} converges to zero because in the limit every point $q_m(i \cdot \Delta t)$ is associated to a via-point, which is initialized to q_i^{pp}. However typically the cost decreases below a reasonable threshold with much fewer via-points (see Table I).

The other initialization cost function C_{ini2} is minimizing both, the collision cost $c_{\text{col}}(\boldsymbol{q})$ and the joint limit cost $c_{\text{joint}}(\boldsymbol{q})$:

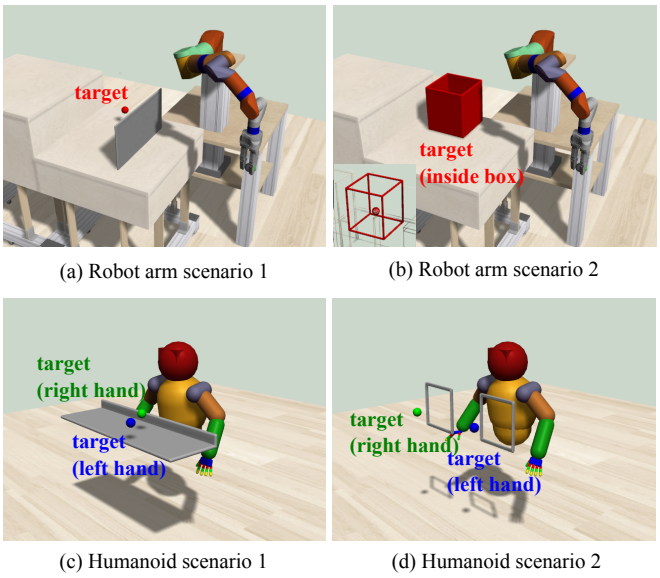
$$C_{\text{ini2}}(m) = \sum_{i=1}^{T} \Big(c_{\text{col}}\big(\boldsymbol{q}_m(i \cdot \Delta t)\big) + c_{\text{joint}}\big(\boldsymbol{q}_m(i \cdot \Delta t)\big) \Big) \quad . \tag{5}$$

The second method usually generates a trajectory which is qualitatively different from the path planning trajectory and only shares the positions and timings of the via-points. However, the initial trajectory $\boldsymbol{q}_m(t)$ also becomes a feasible trajectory when adding enough via-points, because of the consideration of the constraints in cost C_{ini2}.

The proposed algorithm using either of the two initialization cost functions decides for an appropriate number of via-points according to the complexity of the task. On the one hand, if the situation is easy, using only one via-point might already be sufficient, which leads to a faster optimization in the next step. On the other hand, in a complex situation with for example many obstacles, the method adds more via-points until a valid initial solution is found. This then gives the subsequent optimization additional free parameters and thus the possibility to find a better solution.

IV. SEQUENTIAL OPTIMIZATION

For the optimization described in this section, the number of via-points and their timings as a result from the initialization are fixed. Only the parameters $\boldsymbol{X}_{\text{ini}}$ are optimized using local gradient-based constrained optimization. Fig. 1 illustrates a conceptual sketch for the sequential optimization



(a) Robot arm scenario 1

(b) Robot arm scenario 2

(c) Humanoid scenario 1

(d) Humanoid scenario 2

Fig. 2. The four scenarios with the two different robots: 7-DOF robot arm and 16-DOF humanoid upper body.

approach consisting of two steps. The first step is an optimization that has no hard inequality constraints, but instead includes them in the cost to be minimized:

$$\begin{aligned} \boldsymbol{X}_{\text{opt1}} = \underset{\boldsymbol{X}_{\text{opt}}}{\arg\min} \ & \left(C_{\text{opt}} + \omega \left\| \boldsymbol{C}_{\text{ieq}} \right\|_1 \right) \\ \text{s.t.} \quad & \boldsymbol{C}_{\text{eq}} = \boldsymbol{0} \quad , \end{aligned} \tag{6}$$

with ω being a weighting factor.

The cost function to be minimized is C_{opt} and the inequality and equality constraints are described as $\boldsymbol{C}_{\text{ieq}}$ and $\boldsymbol{C}_{\text{eq}}$, respectively. The second optimization, which is initialized with results from the first optimization $\boldsymbol{X}_{\text{opt1}}$, incorporates $\boldsymbol{C}_{\text{ieq}}$ as a hard inequality constraint:

$$\begin{aligned} \boldsymbol{X}_{\text{opt2}} = \underset{\boldsymbol{X}_{\text{opt}}}{\arg\min} \ & C_{\text{opt}} \\ \text{s.t.} \quad & \boldsymbol{C}_{\text{ieq}} \leq \boldsymbol{0}, \ \boldsymbol{C}_{\text{eq}} = \boldsymbol{0} \quad . \end{aligned} \tag{7}$$

TABLE I

SIMULATION RESULTS OF THE MOTION PLANNING FRAMEWORK USING VIA-POINTS (AVERAGE ± STD)

	Task-dependent distribution of via-points and sequential optimization							
	Initialization cost: Eq. (4) *tol*: 0.2 (robot arm), 20 (humanoid)				Initialization cost: Eq. (5) *tol*: 20 (both robots)			
	Success [%]	Number of via-points	Cost	Calc. time [s]	Success [%]	Number of via-points	Cost	Calc. time [s]
Robot arm scenario 1	100	2.30 ± 1.60	0.026 ± 0.024	7.09 ± 7.48	100	1.05 ± 0.21	0.037 ± 0.033	4.51 ± 2.82
Robot arm scenario 2	98	2.71 ± 1.24	0.061 ± 0.039	11.17 ± 8.29	100	1.06 ± 0.24	0.058 ± 0.041	7.18 ± 3.61
Humanoid scenario 1	96	5.52 ± 2.74	0.32 ± 0.062	89.12 ± 85.36	98	1.74 ± 0.60	0.35 ± 0.14	28.41 ± 9.65
Humanoid scenario 2	98	3.86 ± 1.65	0.14 ± 0.020	96.77 ± 122.95	100	2.04 ± 0.78	0.14 ± 0.022	53.40 ± 24.20

The intention behind splitting the optimization into two steps is to obtain a better solution in terms of minimizing C_{opt}. Especially for tasks with higher complexity (higher number of DOF and narrow environment), there are many local minima (e. g., C_1 and C_2 in Fig. 1). If the constraint conditions would be included already from the beginning, the potentially better minimum C_1 cannot be reached, because the optimization is restricted to remain in the feasible area a_2, as shown in Fig. 1a. Instead, when applying two optimization steps as described above, optimization 1 is able to move among the feasible areas towards a better minimum (from area a_2 to area a_1 in Fig. 1b). The inequality constraint conditions are still considered by being part of the cost function (as soft constraints). The subsequent optimization then solves the optimization problem with the inequality constraint conditions (as hard constraints). As a result, the solution can potentially reach a cost value C_1 which is lower than C_2. The quantification of this result is part of the evaluation in Section V.

A. Cost function and constraints

The cost function to be minimized is the length of the joint space trajectory $\boldsymbol{q}_m(t)$ of the robot:

$$C_{\text{opt}} = \frac{1}{2}\sum_{i=2}^{T} \left\| \boldsymbol{q}_m(i\cdot\Delta t) - \boldsymbol{q}_m\big((i-1)\cdot\Delta t\big) \right\|_2 \quad . \tag{8}$$

Additionally, the inequality constraint function deals with the collision cost c_{col} and the joint limit cost c_{joint} over the whole motion duration:

$$\boldsymbol{C}_{\text{ieq}} = \begin{bmatrix} \sum_{i=1}^{T} c_{\text{col}}\big(\boldsymbol{q}_m(i\cdot\Delta t)\big) \\ \sum_{i=1}^{T} c_{\text{joint}}\big(\boldsymbol{q}_m(i\cdot\Delta t)\big) \end{bmatrix} \leq \boldsymbol{0} \quad . \tag{9}$$

Finally, the equality constraint function includes the conditions for achieving a certain target in task space at the final state:

$$\boldsymbol{C}_{\text{eq}} = \left[\ \boldsymbol{p}(T) - \boldsymbol{p}_{\text{des}}\ \right] = \boldsymbol{0} \quad , \tag{10}$$

where $\boldsymbol{p}(T)$ is a function using forward kinematics for calculating the state of the task space at the final timestep T. Vector $\boldsymbol{p}_{\text{des}}$ is the given desired target in task space and can be composed of an arbitrary combination of tasks, such as desired positions or orientations of different bodies. In the experiments in the next section $\boldsymbol{p}_{\text{des}}$ is composed of the 3D positions of the end-effectors in world coordinates.

B. Gradient computation

For solving the optimization problem, a gradient-based approach can be used efficiently, because all gradients are available in their analytic form. They are explained here using the general form of a cost function $\boldsymbol{C}$. The gradients with respect to the via-point parameters $\boldsymbol{X}_{\text{opt}}$ can be expressed as follows:

$$\frac{\mathrm{d}\boldsymbol{C}}{\mathrm{d}\boldsymbol{X}_{\text{opt}}} = \sum_{i=1}^{T}\left(\frac{\partial \boldsymbol{C}}{\partial \boldsymbol{q}_m(i\cdot\Delta t)}\cdot\frac{\mathrm{d}\boldsymbol{q}_m(i\cdot\Delta t)}{\mathrm{d}\boldsymbol{X}_{\text{opt}}}\right) \tag{11}$$
$$\boldsymbol{C} \ni \{C_{\text{opt}}, \boldsymbol{C}_{\text{ieq}}, \boldsymbol{C}_{\text{eq}}\} \quad .$$

Since the joint angle trajectory is formulated as a fifth-order polynomial function, we can analytically derive the second term $\mathrm{d}\boldsymbol{q}_m(i\cdot\Delta t)/\mathrm{d}\boldsymbol{X}_{\text{opt}}$ which then also becomes a function of a fifth-order polynomial. For the gradients of the equality constraints $\boldsymbol{C}_{\text{eq}}$, it becomes the Jacobian matrix of the tasks. The derivation of all other gradients of the cost functions is straightforward, as they are formulated in terms of quadratic cost functions.

V. RESULTS

We applied our method to generating reaching movements for a 7-DOF robot arm and bi-manual reaching motions for a humanoid upper body with 16 DOF (each arm has 7 DOF and the waist has 2 DOF). Fig. 2 shows the four different scenarios for the two different robots in simulation. For each robot, an easy and a comparably harder reaching task have been evaluated. The movement duration has been set to 10 s and 7 s for the robot arm and the humanoid model, respectively.

A. Simulation results

The computer used for the simulation was an Intel(R) Core(TM) i5-2400 CPU (3.10 GHz, 8 GB RAM) running

TABLE II

SIMULATION RESULTS OF MOTION PLANNING USING THE PREVIOUS INITIALIZATION METHOD (AVERAGE ± STD)

	Initialization from straight-line trajectory and sequential optimization								
	Number of via-points: 1			Number of via-points: 3			Number of via-points: 5		
	Success [%]	Cost	Calc. time [s]	Success [%]	Cost	Calc. time [s]	Success [%]	Cost	Calc. time [s]
Robot arm scenario 1	96	0.035 ± 0.048	3.31 ± 1.65	100	0.028 ± 0.031	5.51 ± 2.84	100	0.035 ± 0.041	5.65 ± 2.58
Robot arm scenario 2	80	0.062 ± 0.48	6.74 ± 3.12	80	0.061 ± 0.035	8.09 ± 4.10	84	0.054 ± 0.043	11.87 ± 7.85
Humanoid scenario 1	40	0.31 ± 0.024	19.47 ± 7.03	24	0.28 ± 0.051	44.50 ± 25.27	12	0.27 ± 0.037	92.06 ± 38.70
Humanoid scenario 2	4	0.15 ± 0.001	31.42 ± 8.11	0	-	-	0	-	-

64 Bit Ubuntu Linux 12.04. In this work, the C/C++ library **OMPL**[1] was used for path planning. Additionally, the **SLSQP** function of the C/C++ library **NLopt**[2] was used to solve the SQP problem.

We simulated 50 trials for each scenario. Table I shows the simulation results using our motion planning framework with via-points. We evaluated the two proposed initialization costs and the sequential optimization technique.

At first, it can be confirmed that our method has a high success rate in all scenarios. Even for the difficult scenarios, the system could plan the motions with almost no failure. It can also be confirmed that the number of via-points increases with the complexity of the task. As the table shows, also the computation time increases together with the higher number of via-points. However, the resulting cost values do not rely on the type of initialization cost. When the initialization cost of Eq. (4) is used, a higher number of via-points is selected. This is because the initially planned path is not very short and it is necessary to distribute more via-points to reproduce the path. As a consequence, when using path planning, the cost formulation of Eq. (5) should be preferred over Eq. (4). However, we see potential for the criterion of Eq. (4) for example for imitation learning scenarios [21]. In this case, the path planning trajectory could be replaced by a trajectory demonstrated by the human tutor and it might be preferable to have the initial minimum jerk trajectory closer to the demonstrated path.

The resulting motion of the simulated scenarios is provided as a supplementary video. It can be seen, that even if the results of the path planning are not short and smooth, the via-point formulation and the optimization can generate a short motion with a guaranteed minimal jerk.

B. Comparison with previous initialization method

We compared the proposed initialization method with the previous method of using a straight-line trajectory that connects start and end poses. As the previous method does not determine the number of via-points automatically, we evaluated three different fixed numbers of via-points (i. e., one, three and five). The via-points where distribute equally in time along the initial straight path. Table II shows the motion planning results using the previous initialization together with sequential optimization.

It can be confirmed that the success rate deteriorated with increasing task complexity. Especially, in the humanoid scenario 2, it was almost impossible to plan the motion using the previous method. It can also be seen (robot arm scenario 2 and humanoid scenario 1) that increasing the number of via-points does not necessarily improve the performance. This indicates that it is very important to determine the appropriate timings of the via-points in addition to the number of via-points wrt. the task complexity. In contrast to the previous initialization method, this selection of feasible timings is achieved by the proposed method.

C. Comparison with single optimization

In this section, we compare the sequential optimization with single optimization. Table III shows the simulation results of the motion planning using the proposed initialization method but with a single constrained optimization. For this comparison, the initialization cost from Eq. (5) was used. For the robot arm scenarios, the resulting cost values are basically the same and there is no difference between single and sequential optimization. However, in the humanoid scenarios, the single optimization yields a much higher cost value (which means a longer movement in joint space). This result indicates that the sequential optimization technique is able to find a better local minimum for complex problems.

D. Robot experiments

The robot experiments based on the robot arm scenario 2 have been tested with a real robot. In this experiment, a SCHUNK light-weight arm with 7 DOF was used and the whole movement time was set to 15 s. Fig. 3 shows snapshots of one part of the experiments. Different examples of this experiment are also shown in the supplementary video. We confirmed that the robot could reach the target in the box under changing positions and orientations of the box and with an additional obstacle in the scene. The movements of

[1] http://ompl.kavrakilab.org

[2] http://ab-initio.mit.edu/nlopt

TABLE III

SIMULATION RESULTS OF MOTION PLANNING USING SINGLE OPTIMIZATION (AVERAGE ± STD)

	Success [%]	Number of via-points	Cost	Calc. time [s]
Robot arm scenario 1	100	1.02 ± 0.14	0.037 ± 0.040	2.61 ± 1.63
Robot arm scenario 2	100	1.04 ± 0.20	0.06 ± 0.048	4.60 ± 2.72
Humanoid scenario 1	40	1.85 ± 0.48	0.82 ± 1.51	13.78 ± 4.43
Humanoid scenario 2	98	1.48 ± 0.77	1.18 ± 1.58	24.28 ± 29.12

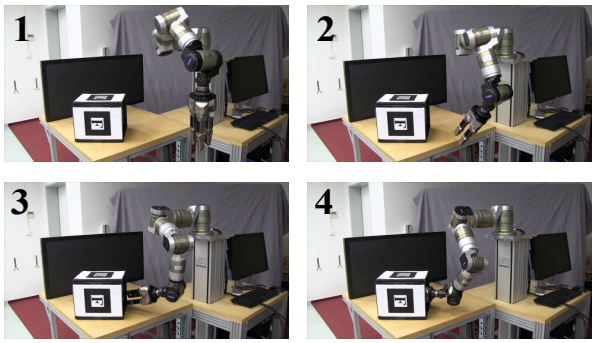


Fig. 3. Snapshots of the real robot experiment.

the robot are guaranteed to be smooth due to the minimum jerk formulation of the via-points.

VI. CONCLUSION

In this paper, we presented a motion planning framework for planning smooth robot movements based on a minimum jerk formulation using via-points. We proposed an initialization method that provides a task-dependent distribution of these via-points. Our initialization method finds an appropriate number of via-points and their timing according to the complexity of the task. Furthermore, we introduced a sequential optimization technique that can potentially reach a lower cost than constrained optimization alone. The optimization procedure finds optimal via-point parameters to allow a robot to achieve a given task without violating constraints. Furthermore, the smoothness of the robot's motion is guaranteed due to the formulation based on a minimum jerk criterion.

For future work two main topics remain. The computational cost to plan the motions is too high for applying the framework to real-time motion control of robots with a large number of DOF. Additionally, we would like to deal with various kinds of constraint conditions such as keeping the position or orientation of an end-effector constant during parts of the motion. Thus in future, we intend to approach the motion planning with via-points using different kinds of constraint conditions.

REFERENCES

[1] 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.

[2] M. Zucker, N. Ratliff, A. D. Dragan, M. Pivtoraiko, M. Klingensmith, C. M. Dellin, J. A. Bagnell, and S. S. Srinivasa, "Chomp: Covariant hamiltonian optimization for motion planning," *The International Journal of Robotics Research*, vol. 32, no. 9-10, pp. 1164–1193, 2013.

[3] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, and S. Schaal, "Stomp: Stochastic trajectory optimization for motion planning," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, 2011, pp. 4569–4574.

[4] P. T. Boggs and J. W. Tolle, "Sequential quadratic programming," *Acta numerica*, vol. 4, pp. 1–51, 1995.

[5] S. Miossec, K. Yokoi, and A. Kheddar, "Development of a software for motion optimization of robots-application to the kick motion of the hrp-2 robot," in *Robotics and Biomimetics, 2006. ROBIO'06. IEEE International Conference on*. IEEE, 2006, pp. 299–304.

[6] S. Lengagne, J. Vaillant, E. Yoshida, and A. Kheddar, "Generation of whole-body optimal dynamic multi-contact motions," *The International Journal of Robotics Research*, vol. 32, no. 9-10, pp. 1104–1119, 2013.

[7] P. M. Wensing and D. E. Orin, "Generation of dynamic humanoid behaviors through task-space control with conic optimization," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, 2013, pp. 3103–3109.

[8] A. El Khoury, F. Lamiraux, and M. Taix, "Optimal motion planning for humanoid robots," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, 2013, pp. 3136–3141.

[9] J. Schulman, Y. Duan, J. Ho, A. Lee, I. Awwal, H. Bradlow, J. Pan, S. Patil, K. Goldberg, and P. Abbeel, "Motion planning with sequential convex optimization and convex collision checking," *The International Journal of Robotics Research*, vol. 33, no. 9, pp. 1251–1270, 2014.

[10] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *Robotics and Automation, IEEE Transactions on*, vol. 12, no. 4, pp. 566–580, 1996.

[11] S. M. LaValle, *Planning Algorithms*. Cambridge, U.K.: Cambridge University Press, 2006, available at http://planning.cs.uiuc.edu/.

[12] N. Vahrenkamp, D. Berenson, T. Asfour, J. Kuffner, and R. Dillmann, "Humanoid motion planning for dual-arm manipulation and regrasping tasks," in *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*. IEEE, 2009, pp. 2464–2470.

[13] K. Hauser and V. Ng-Thow-Hing, "Fast smoothing of manipulator trajectories using optimal bounded-acceleration shortcuts," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, May 2010, pp. 2493–2498.

[14] C. Sung, T. Kagawa, and Y. Uno, "Whole-body motion planning for humanoid robots by specifying via-points," *International Journal of Advanced Robotic Systems*, vol. 10, 2013.

[15] ——, "Efficient planning of humanoid motions by modifying constraints," *Paladyn, Journal of Behavioral Robotics*, vol. 4, no. 1, pp. 23–33, 2013.

[16] T. Flash and N. Hogan, "The coordination of arm movements: an experimentally confirmed mathematical model," *The journal of Neuroscience*, vol. 5, no. 7, pp. 1688–1703, 1985.

[17] Y. Uno, M. Kawato, and R. Suzuki, "Formation and control of optimal trajectory in human multijoint arm movement," *Biological cybernetics*, vol. 61, no. 2, pp. 89–101, 1989.

[18] H. Miyamoto, S. Schaal, F. Gandolfo, H. Gomi, Y. Koike, R. Osu, E. Nakano, Y. Wada, and M. Kawato, "A kendama learning robot based on bi-directional theory," *Neural networks*, vol. 9, no. 8, pp. 1281–1302, 1996.

[19] A. Piazzi and A. Visioli, "Global minimum-jerk trajectory planning of robot manipulators," *Industrial Electronics, IEEE Transactions on*, vol. 47, no. 1, pp. 140–149, 2000.

[20] J. Kuffner and S. M. LaValle, "RRT-connect: An efficient approach to single-query path planning," in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, vol. 2, no. Icra. Ieee, 2000, pp. 995–1001.

[21] M. Mühlig, M. Gienger, and J. J. Steil, "Interactive imitation learning of object movement skills," *Autonomous Robots*, vol. 32, no. 2, pp. 97–114, 2012.