

Fast Manipulability Maximization Using Continuous-Time Trajectory Optimization

Filip Marić^{*†}, Oliver Limoyo^{*}, Luka Petrović[†], Trevor Ablett^{*}, Ivan Petrović[†], and Jonathan Kelly^{*}

Abstract—A significant challenge in manipulation motion planning is to ensure agility in the face of unpredictable changes during task execution. This requires the identification and possible modification of suitable joint-space trajectories, since the joint velocities required to achieve a specific end-effector motion vary with manipulator configuration. For a given manipulator configuration, the joint space-to-task space velocity mapping is characterized by a quantity known as the manipulability index. In contrast to previous control-based approaches, we examine the maximization of manipulability during planning as a way of achieving adaptable and safe joint space-to-task space motion mappings in various scenarios. By representing the manipulator trajectory as a continuous-time Gaussian process (GP), we are able to leverage recent advances in trajectory optimization to maximize the manipulability index during trajectory generation. Moreover, the sparsity of our chosen representation reduces the typically large computational cost associated with maximizing manipulability when additional constraints exist. Results from simulation studies and experiments with a real manipulator demonstrate increases in manipulability, while maintaining smooth trajectories with more dexterous (and therefore more agile) arm configurations.

I. INTRODUCTION

Motion planning is a fundamental challenge for robotic manipulators executing complex tasks. To perform a task successfully, the motion planner must generate a joint space trajectory that respects constraints induced by the task (e.g., collision avoidance). The existence of these constraints may cause the planning algorithm to generate trajectories that contain configurations with suboptimal joint space-to-task space mappings. Consequently, in scenarios where the manipulator is operating autonomously in non-static environments (e.g., during collaborative task execution), large joint motions may be required in order for the manipulator to adapt to unexpected changes in constraints during task execution. A configuration’s capacity for movement in the task space can be inferred from the manipulability ellipsoid [1], whose axis lengths give a measure of how effectively joint velocities map to directions in task space. The manipulability index introduced by Yoshikawa in [2] is proportional to this ellipsoid’s

This research was supported in part by a Dean’s Catalyst Professorship from the University of Toronto and the European Regional Development Fund under the grant KK.01.1.1.01.0009 (DATACROSS).

^{*} Filip Marić, Oliver Limoyo, Trevor Ablett, and Jonathan Kelly are with the University of Toronto, Institute for Aerospace Studies, Space and Terrestrial Autonomous Robotic Systems Laboratory, Canada. {<first name>.<last name>@robotics.utoronto.ca}

[†] Filip Marić, Luka Petrović, and Ivan Petrović are with the University of Zagreb, Faculty of Electrical Engineering and Computing, Laboratory for Autonomous Systems and Mobile Robotics, Croatia. {<first name>.<last name>@fer.hr}

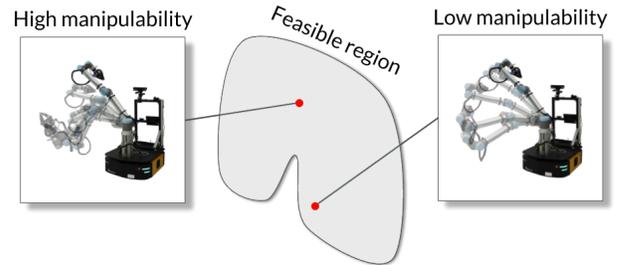


Fig. 1: Comparison of two solutions for reaching a position goal from a given near-singular starting configuration (caused by fully extending the arm initially). The right image shows a solution based purely on inverse kinematics, which maintains low manipulability throughout. The image on the left shows a trajectory generated by our method, which avoids excessive arm extension.

volume and is commonly used in pose-tracking controllers to avoid particularly unfavourable mappings that are known as *singularities*. It follows that, by maximizing manipulability, we can ensure that the manipulator possesses a higher level of overall dexterity throughout the planned trajectory—while avoiding large and potentially hazardous joint movements. This enables rapid and predictable manipulator responses in safety-critical applications as diverse as robotically-assisted surgery [3] or satellite capture [4]. Rather than relying on the incorporation of singularity avoidance in the tracking controller itself, our method generates a motion plan that preemptively maximizes the overall manipulability throughout the manipulator’s trajectory.

Our goal is to optimize the trajectory such that, in the face of unexpected task changes (e.g., to the final end-effector pose), the manipulator is able to adapt its motion with minimal joint position changes. The arm trajectory shown on the right side of Fig. 1 is an example in which the configuration is initially (and throughout the motion) nearly singular, with the arm fully extended. Hence, even small movements of the end-effector along the extended axis will result in large joint velocities at the elbow. On the left side of Fig. 1 is a trajectory with high manipulability that reaches the same end-effector goal position in task space while avoiding the elbow singularity. By choosing to represent the complete joint trajectory as a sample from a continuous-time Gaussian process (GP) [5], the manipulability maximization problem defined above can be formulated as probabilistic inference; a *maximum a posteriori* (MAP) estimator can be used [6], [7] to find a solution that is, locally, relatively far from near-singular regions while also enforcing a notion of smoothness through a trajectory prior. We build on the approach presented in [7], by introducing a likelihood factor

which helps avoid low manipulability configurations induced by task constraints. To the best of the authors' knowledge, this is the first method to directly integrate manipulability maximization within a trajectory optimization formulation. We make the following contributions:

- (i) we formulate manipulability maximization as a continuous-time trajectory optimization problem,
- (ii) we demonstrate that this approach can be applied to a variety of planning scenarios,
- (iii) we show that, for our chosen trajectory representation, the problem can be efficiently solved, and
- (iv) we compare our approach to existing singularity avoidance and manipulability maximization techniques.

II. RELATED WORK

Manipulability maximization has been extensively studied from the perspective of robust kinematic control. Redundancy resolution schemes such as [8] and [9] have long been used for singularity avoidance, and consequently to increase overall manipulability. More recently, quadratic programming (QP) has been examined as an efficient method for manipulability maximization in constrained inverse kinematics solvers [10]–[12]. These kinematic control policies are efficient for end-effector tracking tasks, although their success ultimately depends on the trajectory and on manipulator redundancy. Moreover, many common tasks cannot be efficiently defined in this manner (e.g., the task of reaching a goal configuration while avoiding an obstacle).

Previous attempts to generate joint-space trajectories or paths with high manipulability have resorted to methods that suffer from high computational cost. For example, in [13], a maximum manipulability discrete joint-space path for a five degrees-of-freedom (DOF) manipulator is produced by a genetic algorithm. The resolution-complete search in [14] generates a sequence of high manipulability configurations from an end-effector path in obstacle-free environments. In [15], a singularity-free joint-space path is generated by parameterizing the end-effector trajectory using Bezier curves and finding optimal configurations through simulated annealing. In [16] a manipulability cost is learned as one of several cost terms in a formulation similar to CHOMP [17], with the aim of completing a ‘disentangling’ task from a provided demonstration. Additional interpolation may also be required in order to ensure smooth transitions between the states generated by these methods. However, no prior information will be available on the manipulability or collision of the arm with the environment for these interpolated states.

Trajectory optimization algorithms [17]–[20] minimize a cost function composed of both optimality and feasibility terms and have been used for online planning and replanning due to their low computational demands. However, avoiding singularities presents a difficult problem for such approaches, since they require dense discretizations to produce smooth and feasible solutions when handling complicated constraints. This is especially true when the problem is

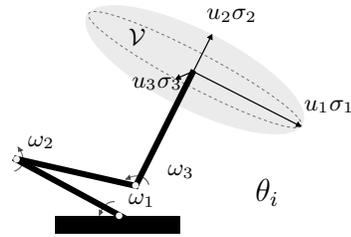


Fig. 2: Illustration of the manipulability ellipsoid of volume \mathcal{V} for a manipulator end-effector at configuration θ_i . Larger axis lengths indicate higher mobility.

additionally constrained by the end-effector pose or by the need to avoid obstacles in the environment.

A continuous-time trajectory representation can sidestep many of these difficulties by reducing the number of states used in the optimization. Mukadam et al. [7] use a GP trajectory representation, which allows them to treat the motion planning problem as probabilistic inference on a factor graph; as such, they are able to interpolate over a trajectory and generate additional gradient information. Consequently, highly constrained problems can be solved efficiently using a MAP estimator while requiring only a relatively small number of states. The resulting trajectory can be queried at any point, allowing the robot’s manipulability to be monitored throughout.

III. MANIPULABILITY

Consider a joint configuration θ_i as the state of a trajectory θ at time t_i . The kinematic relationship between configuration space and task space velocities at θ_i for an n -DOF robot is defined as

$$\dot{\mathbf{x}} = \mathbf{J}(\theta_i) \boldsymbol{\omega}, \quad (1)$$

where $\mathbf{J}(\theta_i) \in \mathbb{R}^{p \times n}$ is the manipulator Jacobian matrix at θ_i , while $\boldsymbol{\omega} \in \mathbb{R}^n$ and $\dot{\mathbf{x}} \in \mathbb{R}^p$ are the configuration and task space velocities at t_i , respectively. Now, consider an n -dimensional sphere in the space of unit joint velocities $\|\boldsymbol{\omega}\|^2 = 1$; using Eq. (1) we can define the mapping to the Cartesian (task) velocity space as

$$\|\dot{\mathbf{x}}\|^2 = \dot{\mathbf{x}}^T (\mathbf{J}\mathbf{J}^T)^{-1} \dot{\mathbf{x}}. \quad (2)$$

From Eq. (2), we see that the scaling of joint velocities to the task space depends on the conditioning of the symmetric positive semi-definite matrix $\mathbf{J}\mathbf{J}^T$. Manipulability provides a computationally tractable measure of the conditioning of $\mathbf{J}\mathbf{J}^T$ for any joint configuration [2].

A. Manipulability

The matrix $\mathbf{J}\mathbf{J}^T$ in Eq. (2) also defines the *manipulability ellipsoid* [1] of the end-effector. The principal axes $\sigma_1 \mathbf{u}_1, \sigma_2 \mathbf{u}_2, \dots, \sigma_p \mathbf{u}_p$ of this ellipsoid can be determined through singular value decomposition of $\mathbf{J} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$. The manipulability measure (index) of a given kinematic chain at θ_i is defined as

$$m = \sqrt{\det(\mathbf{J}\mathbf{J}^T)} = \sigma_1 \sigma_2 \dots \sigma_k \dots \sigma_p, \quad (3)$$

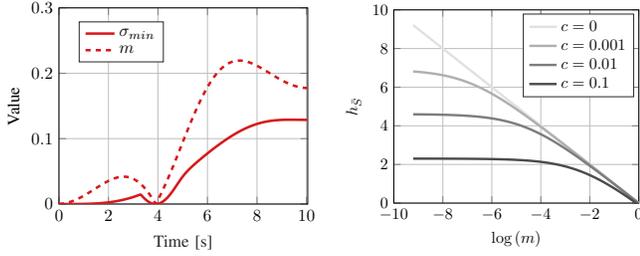


Fig. 3: Left: Comparison of the manipulability measure (dashed line) and the smallest singular value of the Jacobian throughout a trajectory. Right: Shape of the likelihood function in Eq. (10), which depends on the parameter c .

and is proportional to the volume, \mathcal{V} , of the manipulability ellipsoid [1]. Here, $\sigma_k \geq 0$ is the k -th largest singular value of \mathbf{J} , while \mathbf{u}_k is the k -th column vector of \mathbf{U} . A low manipulability corresponds to a low volume of the manipulability ellipsoid, inhibiting motion in the task space. An example of the manipulability ellipsoid of the end-effector frame of a simple manipulator is depicted in Fig. 2. The gradient of Eq. (3) can be calculated numerically [10], but it is also possible to derive the gradient analytically with respect to the j -th joint of the configuration θ_i using Jacobi's identity [9],

$$\frac{\partial m}{\partial \theta_{i,j}} = m \text{Tr} \left(\frac{\partial \mathbf{J}}{\partial \theta_{i,j}} \mathbf{J}^\dagger \right). \quad (4)$$

Moreover, the components of Eq. (4), \mathbf{J} and $\frac{\partial \mathbf{J}}{\partial \theta_{i,j}}$, can be calculated via geometrical methods [21].

B. Singularities

The concept of manipulability relates directly to the conditioning of the manipulator Jacobian matrix. Configurations that result in the matrix $\mathbf{J}\mathbf{J}^T$ in Eq. (2) being non-invertible are termed *singularities*. Consider a kinematic chain and corresponding manipulability ellipsoid with a volume $\mathcal{V} \propto m$, as shown in Fig. 2. If the ellipsoid contains one or more zero-length principal axes, it follows that $\mathcal{V} = 0$ and $m = 0$; configurations yielding such ellipsoids are known as *singular configurations*. We can define a minimum acceptable ellipsoid volume $\mathcal{V}_S \in \mathbb{R}_+$, and regard configurations that result in a manipulability $m < m(\mathcal{V}_S)$ to be *nearly singular*.

Conversely, a high manipulability value does not guarantee that a configuration is not nearly singular, as an ellipsoid with one 'degenerate' (i.e., of very small magnitude) axis may still have a large overall volume. The volume of any manipulability ellipsoid for the chain is bounded by the value \mathcal{V}_{max} , determined by the chain's kinematic parameters. Assuming that the axes $\sigma_1 \mathbf{u}_1, \sigma_2 \mathbf{u}_2, \dots, \sigma_p \mathbf{u}_p$ are of an acceptable length for all such ellipsoids, we infer that configurations whose ellipsoid volume are sufficiently close to \mathcal{V}_{max} are *not* nearly singular (labelled as \bar{S}). Fig. 3 compares the smallest singular value of the manipulator Jacobian to the manipulability measure (index) throughout a sample trajectory; the manipulability measure roughly follows the magnitude of the smallest singular value, matching its peaks and troughs.

IV. MANIPULABILITY MAXIMIZATION FORMULATION

If we consider the joint space trajectory θ as a function which maps every time instance $0 \leq t \leq T$ to a configuration $\theta(t)$, manipulability maximization can be formulated as trajectory optimization,

$$\underset{\theta(t)}{\text{minimize}} \quad \mathcal{F}[\theta(t)] + \lambda \mathcal{M}[\theta(t)] + \mu \mathcal{C}[\theta(t)], \quad (5)$$

where $\mathcal{F}[\theta(t)]$ is a cost functional encoding smoothness, $\mathcal{M}[\theta(t)]$ is a cost functional relating manipulability to the trajectory space, and $\mathcal{C}[\theta(t)]$ is a cost functional that enforces collision avoidance (necessary in many environments).

A. Representing the Trajectory as a GP

Using the definition of trajectory optimization in Eq. (5), we follow the derivation in [6], representing the continuous-time trajectory as a sample from a vector-valued GP, $\theta(t) \sim \mathcal{GP}(\boldsymbol{\mu}(t), \mathbf{K}(t, t'))$, with mean $\boldsymbol{\mu}(t)$ and covariance $\mathbf{K}(t, t')$, generated by a linear time-varying stochastic differential equation (LTV-SDE),

$$\dot{\theta}(t) = \mathbf{A}(t)\theta(t) + \mathbf{u}(t) + \mathbf{F}(t)\mathbf{w}(t), \quad (6)$$

where \mathbf{A} and \mathbf{F} are system matrices, \mathbf{u} is a known control input and $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_c)$. For any set of times \mathbf{t} , the corresponding *support states* $\theta = [\theta_0 \dots \theta_N]^T$ can be matched to an exponential prior distribution resulting from the system in Eq. (6), with the mean $\boldsymbol{\mu}$ and kernel \mathbf{K} :

$$p(\theta) \propto \exp\left\{-\frac{1}{2}\|\theta - \boldsymbol{\mu}\|_{\mathbf{K}}^2\right\}. \quad (7)$$

The Markovian property of the process in Eq. (6) allows us to factor the prior in Eq. (7) as

$$p(\theta) \propto f_0^p(\theta_0) f_N^p(\theta_N) \prod_{i=0}^{N-1} f_i^{gp}(\theta_i, \theta_{i+1}). \quad (8)$$

where f_0^p and f_N^p define the prior distributions on the start and end states, and f_i^{gp} is the GP prior factor as defined in [7]. Furthermore, this property allows for interpolation of the trajectory in $\mathcal{O}(1)$ time [6].

B. Manipulability Likelihood Function

Representing a trajectory using the GP in Eq. (6) allows us to intuitively (locally) search for high manipulability variants using probabilistic inference. We formulate the likelihood of the trajectory θ being free of singular (low manipulability) configurations, denoting this event by \bar{S} . The singularity factor $f_i^{\bar{S}}$ defines this likelihood for the support states, while $f_\tau^{\bar{S}}$ does so for the interpolated states at times $t_i < \tau < t_{i+1}$:

$$\begin{aligned} f_i^{\bar{S}} &= \exp\left\{-\frac{1}{2}\|h_{\bar{S},i}(\theta_i)\|_{\Sigma_{\bar{S}}}^2\right\}, \\ f_\tau^{\bar{S}} &= \exp\left\{-\frac{1}{2}\|h_{\bar{S},i}(\boldsymbol{\mu}(\tau) + \boldsymbol{\Lambda}(\tau)(\theta_i - \boldsymbol{\mu}_i) \right. \\ &\quad \left. \boldsymbol{\Psi}(\tau)(\theta_{i+1} - \boldsymbol{\mu}_{i+1})\|_{\Sigma_{\bar{S}}}^2\right\} \end{aligned} \quad (9)$$

Matrices $\boldsymbol{\Lambda}$ and $\boldsymbol{\Psi}$ are defined as in [7], and $h_{\bar{S},i}$ is the cost function. Parametrizing the distributions of the factors

in Eq. (9) using the manipulability index allows us to find a maximum manipulability posterior using a MAP estimator.

The manipulability index may vary by several orders of magnitude throughout a trajectory, as it is proportional to a p -dimensional volume. Additionally, large changes in manipulability can be caused by small shifts in the manipulator configuration. This presents a problem when maximizing a likelihood of the form in Eq. (9), as $\|h_{\bar{s},i}\|_{\Sigma_{\bar{s}}}^2$ needs to be minimized. To this end, we choose the cost $h_{\bar{s},i}$ to be logarithmic,

$$h_{\bar{s},i} = \log \left(\frac{m_{max} + c}{m + c} \right), \quad (10)$$

where the value m_{max} is the manipulability value at \mathcal{V}_{max} or higher. The constant c serves to limit the log-linear cost change when the value is below a certain order of magnitude, resulting in the gradient

$$\frac{\partial h_{\bar{s},i}}{\partial \theta_{i,j}} = -\frac{m}{m+c} \text{Tr} \left(\frac{\partial \mathbf{J}}{\partial \theta_{i,j}} \mathbf{J}^\dagger \right). \quad (11)$$

This reduces the range of possible gradient values, simplifying the choice of weighing parameter $\Sigma_{\bar{s}}$. In Fig. 3, on the right we can see the effect of the value c on the overall cost in Eq. (11).

C. Optimizing the Trajectory

By representing the initial trajectory as the prior in Eq. (8), we can find a continuous trajectory which (locally) maximizes the likelihood in Eq. (9) by using a MAP estimator over the support states

$$\theta^* = \arg \max_{\theta} p(\theta) l(\theta; \bar{S}) l(\theta; \bar{C}). \quad (12)$$

The maximization in Eq. (12) can be achieved by computing

$$\theta^* = \arg \min_{\theta} \frac{1}{2} \|\theta - \mu\|_{\mathbf{K}}^2 + \frac{1}{2} \|\mathbf{h}_{\bar{S}}(\theta)\|_{\Sigma_{\bar{S}}}^2 + \frac{1}{2} \|\mathbf{h}_{\bar{C}}(\theta)\|_{\Sigma_{\bar{C}}}^2. \quad (13)$$

This is equivalent to the negative log of the posterior, where $\mathbf{h}_{\bar{S}}$ and $\mathbf{h}_{\bar{C}}$ are vectors of manipulability and collision costs for both support and interpolated states. By linearizing Eq. (13) around the current trajectory θ , we arrive at a least squares problem.

Barfoot et. al [6] show that the GP generated by Eq. (6) results in a kernel \mathbf{K} that induces sparsity in the problem defined by Eq. (13), making it easily solvable using sparse Cholesky decomposition. Mukadam et. al [7] show that a constant velocity prior (i.e., forming a straight line in joint space between start and goal states), parametrized by the process noise covariance \mathbf{Q}_c , generates a kernel which penalizes deviation from the prior in a planning context. This clearly corresponds to the smoothness functional $\mathcal{F}[\theta(t)]$ in the trajectory optimization definition from Eq. (5).

The likelihood $l(\theta; \bar{S})$ assumes the role of $\mathcal{M}[\theta(t)]$, with the covariance $\Sigma_{\bar{S}}$ serving as the weighing term λ . Similarly, the collision-free likelihood $l(\theta; \bar{C})$, defined in [7] (derived from [17]) as the signed distance of the robot's body from an obstacle, naturally represents the collision

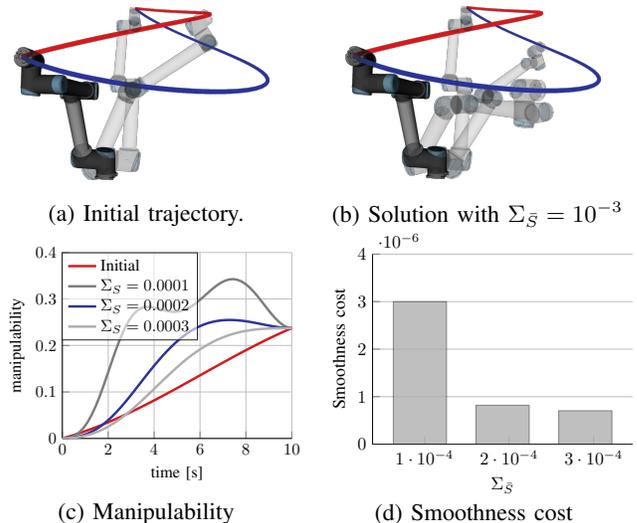


Fig. 4: Reaching a goal configuration. Top: Visualization, the final state is opaque. Bottom: The manipulability and smoothness costs for different $\Sigma_{\bar{S}}$.

avoidance functional $\mathcal{C}[\theta(t)]$. In [7], likelihoods constraining end-effector position and orientation over support states are also defined.

V. EXPERIMENTS

In this section, we demonstrate how our formulation can be used to optimize constrained trajectories, as well as to avoid constraint-induced singularities, and show how the GP trajectory representation can help make this process more efficient. We also compare the approach to manipulability maximization presented in this paper to existing control-based formulations, by quantitatively evaluating performance on a constrained planning problem with randomized initial manipulator configurations. Finally, we present results for a singularity avoidance scenario involving a real Universal Robots UR-10 manipulator available in our laboratory at the University of Toronto, which is able to move throughout a 6-DOF task space that contains many singular and near-singular configurations. All planning computations were performed on a laptop with an Intel i7-8750H CPU running at 2.20 GHz and with 16 GB of RAM.

A. Maximizing Manipulability

In the scenario shown in Fig. 4, the goal configuration needs to be reached by the manipulator in $T = 10$ s. A simple linear interpolation in joint space between the start and goal configurations is used as an initializing trajectory, with 11 support states, 89 interpolated states, $\mathbf{Q}_c = 10^5 \mathbf{I}$, and with the covariance in Eq. (8) of the start and goal state priors set to $\Sigma_{\theta} = 10^{-3} \mathbf{I}$.¹ Examining the manipulability of the initial trajectory in Fig. 4c, we can see that the starting configuration at $t = 0$ is clearly singular. After optimizing the joint trajectory to maximize our likelihood function (i.e., Eq. 10) using the method described in Section IV, the time spent in high manipulability configurations is much

¹Unless specified otherwise, we use these parameters in all experiments.

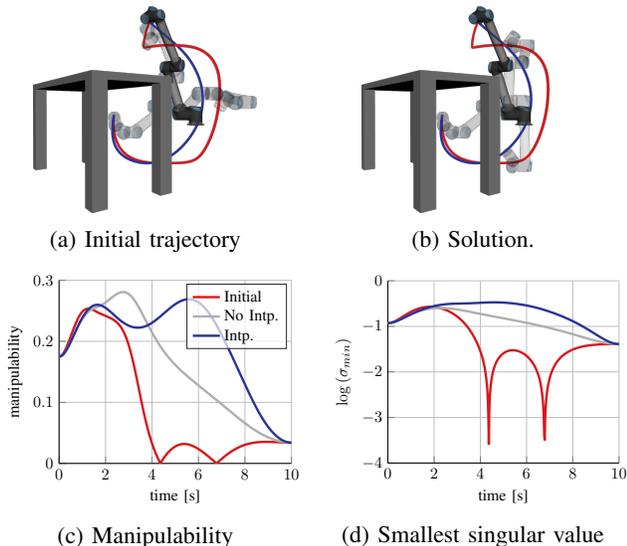


Fig. 5: Reaching a goal configuration while avoiding an obstacle. Top: Visualization, the final state is opaque. Bottom: The manipulability and smallest singular values with and without the use of interpolated states.

greater; the computation (planning) time is 20 ms. Figs. 4a and 4b illustrate that, because we have only constrained the maximization by fixing the starting and final manipulator configuration and the velocity, the optimized trajectory results in a different end-effector path (blue line) than the initialization (red line). The covariance parameter $\Sigma_{\bar{s}}$ serves as a gain parameter, where lower values put greater weight on the manipulability cost term. As shown in Figs. 4c and 4d, a greater $\Sigma_{\bar{s}}$ will reduce the effect of manipulability maximization, while a very low value will suppress overall smoothness, leading to a noisy and infeasible result. Even though trajectory optimization methods like the one described in Section IV find only a locally optimal solution, formulating the problem by constraining the trajectory in this manner results in low sensitivity to the exact parametrization when a reasonable initialization is used.

B. Collision Avoidance

Manipulability is often overlooked in tasks that include collision avoidance. In Fig. 5a, we visualize a trajectory generated by a motion planning algorithm that reaches a goal configuration while avoiding collision with a table-shaped obstacle in the workspace. The manipulability index and smallest singular value plots in Figs. 5c and 5d show that this trajectory contains singularities induced by collision avoidance. If a change in task space constraints were to happen during trajectory execution, and if this change required a rapid response (e.g., if another obstacle appeared or a collaborator pushed the end-effector in a certain direction), the poor conditioning of the Jacobian matrix would cause violent joint movements. To give a numerical example, had this happened at the 6.5 s mark, generating an end-effector velocity of 1 cm/s in the x direction in task space by computing the pseudo-inverse of the kinematic relationship in Eq. (1) would have resulted in joint velocities exceeding 150 rad/s.

This possibility is prevented by optimizing the initial trajectory for (locally) maximum manipulability. In addition to the start and goal states, we set the state prior at the fifth support state (4.5 s) to that of the initial trajectory, with a covariance of $\Sigma_{\theta} = 10^3 \mathbf{I}$. Since the initial trajectory successfully avoids collision, this improves the default straight line prior; multiple states can be fixed in this manner. To maintain distance from the obstacle, we add collision avoidance factors as described in [7] to each support state, with the parameters $\Sigma_{obs} = 10^2 \mathbf{I}$ and $\epsilon = 0.3$. Due to the local nature of the trajectory optimization method, the choices for Σ_{obs} and $\Sigma_{\bar{s}}$ determine the trade off between collision avoidance and manipulability maximization, and so this selection needs to be made carefully. The optimization is carried out within 5 ms and 50 ms without and with interpolated states, respectively. Fig. 5b shows that collisions, as well as the singularities caused by collision avoidance, are completely prevented.

C. Reaching Task

Reaching a Cartesian goal point with the end-effector is a very common manipulation task, often performed in situations where unexpected changes in the task are possible and where maintaining high manipulability is therefore very important (e.g., in kinesthetic teaching). As mentioned in Section II, most manipulability maximization methods use kinematic control or global optimization to follow a path initialized in Cartesian space. The local nature of our approach allows for fast computation, comparable to [10], [11], when used for planning a trajectory.

To further illustrate the benefits of our technique, we compare with the QP formulation in [11] and the established singularity avoidance approach in [8]. We plan a trajectory to reach a desired end-effector position from 50 random (feasible) initial configurations. Here, a joint-space trajectory (position and velocity profile) must be generated, with a time step of $T = 0.02$, that maximizes the average manipulability and maintains joint velocities below $\frac{\pi}{3}$ rad/s. We initialize the problem with 50 support states, parametrizing the optimization with $\mathbf{Q}_c = 10^6 \mathbf{I}$, $\Sigma_{\bar{s}} = 0.0013$ and $\Sigma_{\theta} = 10^{-3} \mathbf{I}$. To identify the final configuration used to initialize the trajectory, we iterate over 20 possible inverse kinematics solutions generated by the `quadprog` function in MATLAB. We avoid initializations that inevitably pass through a singularity by using the fast interpolation property to choose an initialization with the greatest minimum manipulability index.

In Table I, we compare the results produced by these three algorithms by examining the (average) minimum, maximum, and mean trajectory manipulability scores, as well as the computation times and the joint velocities involved. It is clear that, even when searching over a large number of possible initializations, our method achieves dramatically lower computation times. Consequently, it reaches the highest average and maximum manipulabilities, with the average lowest manipulability index value comparable to [11]. The computation time and success rate for [11] matches the

TABLE I: Trajectory generation performance comparison for reaching task.

	Manipulability			Velocities [rad/s]		Solve Time [s]			
	Avg.	Min	Max	Max	Avg	Total	Opt.	Init.	Solved
This paper - No Intp.	0.1316	0.0460	0.1984	0.2830	0.2009	0.2560	0.0213	0.2347	50/50
This paper - Intp.	0.1422	0.0450	0.2206	0.3529	0.2425	0.3158	0.0870	0.2288	50/50
Method in [11]	0.1385	0.0481	0.2077	0.3178	0.1173	1.5064	1.5064	0	46/50
Method in [8]	0.0748	0.0286	0.1193	0.1310	0.0482	0.8155	0.8155	0	50/50

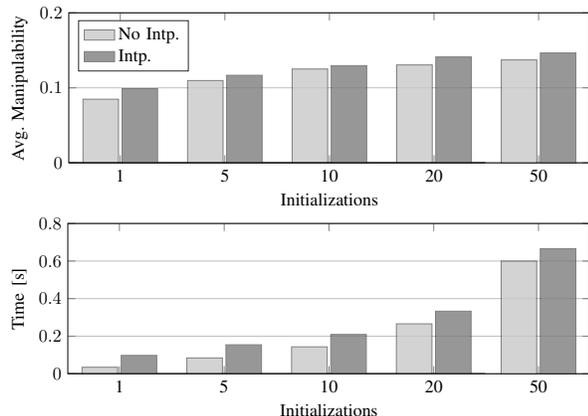


Fig. 6: Performance of our method for the task described in Section V-C with respect to the number of initial trajectories computed. Top: Average manipulability during task execution. Bottom: Solve time.

comparison in [12], where it is concluded that the volatility of the manipulability gradient has a significant effect on convergence. The method in [8] primarily focusses on avoiding singularities, and also reaches the lowest manipulability index values. However, it converges faster than [11] and it always finds a solution.

Initializing over multiple final states (IK solutions) helps avoid singular configurations that would inevitably need to be traversed in joint space, given a bad initialization. Consequently, the smoothness cost is centered around a trajectory with the minimum number of singularities, which we can then easily optimize as described in Section V-A. It is worth noting that most of the computation time for our method is taken up by finding the best initialization; this averages around 0.22 s for a 6-DOF manipulator. We posit that this could be substantially reduced by employing task- or robot-specific heuristics², or fast kinematics solvers like TRAC-IK [22]. Our hypothesis is supported by the results in Fig. 6, where the average manipulability value clearly rises with the number of precomputed solutions. There is a diminishing return, however, as increasing the number of solutions past 20 has little effect on the average value, with computation time growing proportionally.

D. Experiments on a Real Robot

Lastly, we show that the trajectories generated using our method are smooth and can be executed on a real UR-10 manipulator. The goal is again to generate a maximum manipulability trajectory reaching a final state which is

²For example, avoiding initializations that pass through the elbow singularity, which is trivial to do.

constrained by end-effector position. We pick a starting configuration that situates the manipulator to one side, as shown in Fig. 7. In Fig. 8 we can see that the velocities generated using our method remain smooth and relatively low, with some noise caused by dynamic effects. As expected, the method in [11] follows the shortest Cartesian path towards the goal position, reaching a similar final configuration to that of our method. However, our method maintains a higher manipulability value throughout the motion.

VI. CONCLUSION AND FUTURE WORK

We have presented a novel trajectory planning and replanning method that both maximizes the overall manipulability along a trajectory and inherently avoids singularities. Our work shows that maximizing manipulability is highly useful for tasks that must be carried out in uncontrolled environments, or when additional constraints are present. Unlike tracking methods that are commonly used for trajectory planning, our method searches for solutions in joint space, allowing for the efficient use of prior information about the task at hand. This results in lower computation times and greater overall performance for many tasks. As an interesting avenue for future work, we believe that our method can be extended to end-effector path tracking tasks by projecting the gradient information into the appropriate null space.

REFERENCES

- [1] L. Sciavicco and B. Siciliano, *Modelling and Control of Robot Manipulators*, ser. Advanced Textbooks in Control and Signal Processing. Berlin, Heidelberg: Springer Science & Business Media, 2012.
- [2] T. Yoshikawa, "Manipulability of Robotic Mechanisms," *Int. J. Robot. Res.*, vol. 4, no. 2, pp. 3–9, 1985.
- [3] J. San Martin, G. Trivino, and S. Bayona, "Mechanical Design of a Minimally Invasive Surgery Trainer Using the Manipulability as Measure of Optimization," in *Proc. IEEE Int. Conf. Mechatronics (ICM'07)*, Changchun, China, 2007.
- [4] K. Nanos and E. Papadopoulos, "Avoiding Dynamic Singularities in Cartesian Motions of Free-Floating Manipulators," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 51, no. 3, pp. 2305–2318, 2015.
- [5] C. E. Rasmussen, "Gaussian Processes in Machine Learning," in *Advanced Lectures on Machine Learning*, ser. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2004, vol. 3176, pp. 63–71.
- [6] T. D. Barfoot, C. H. Tong, and S. Särkkä, "Batch Continuous-Time Trajectory Estimation as Exactly Sparse Gaussian Process Regression," in *Proc. Robotics: Science and Systems (RSS)*, Berkeley, USA, 2014.
- [7] M. Mukadam, J. Dong, X. Yan, F. Dellaert, and B. Boots, "Continuous-time Gaussian process motion planning via probabilistic inference," *Int. J. Robot. Res.*, vol. 37, no. 11, pp. 1319–1340, 2018.
- [8] S. Chiaverini, "Singularity-Robust Task-Priority Redundancy Resolution for Real-Time Kinematic Control of Robot Manipulators," *IEEE Trans. Robot. Autom.*, vol. 13, no. 3, pp. 398–410, 1997.
- [9] G. Marani, J. Kim, J. Yuh, and W. K. Chung, "A real-time approach for singularity avoidance in Resolved Motion Rate Control of Robotic Manipulators," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA'02)*, vol. 2, Washington D.C., USA, 2002, pp. 1973–1978.



Fig. 7: Experiment on real manipulator. The left image shows a trajectory generated using [11], while the image on the right shows a trajectory generated by our method.

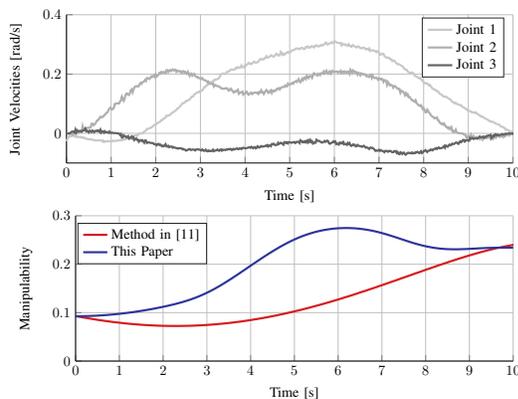


Fig. 8: Top: Velocities of the three largest UR-10 joints using our method. Bottom: Manipulability values compared to [11].

[10] K. Dufour and W. Suleiman, "On Integrating Manipulability Index into Inverse Kinematics Solver," in *Proc. IEEE/RSS Int. Conf. Intelligent Robots and Systems (IROS'17)*, Vancouver, Canada, 2017, pp. 6967–6972.

[11] Y. Zhang, X. Yan, D. Chen, D. Guo, and W. Li, "QP-based refined manipulability-maximizing scheme for coordinated motion planning and control of physically constrained wheeled mobile redundant manipulators," *Nonlinear Dynamics*, vol. 85, no. 1, pp. 245–261, 2016.

[12] L. Jin, S. Li, H. M. La, and X. Luo, "Manipulability Optimization of Redundant Manipulators Using Dynamic Neural Networks," *IEEE Trans. Ind. Electron.*, vol. 64, no. 6, pp. 4710–4720, 2017.

[13] R. Menasri, A. Nakib, H. Oulhadj, B. Daachi, P. Siarry, and G. Hains, "Path planning for redundant manipulators using metaheuristic for bilevel optimization and maximum of manipulability," in *Proc. IEEE Int. Conf. Robotics and Biomimetics (ROBIO'13)*, Shenzhen, China, 2013, pp. 145–150.

[14] L. Guilamo, J. Kuffner, K. Nishiwaki, and S. Kagami, "Manipulability Optimization for Trajectory Generation," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA'06)*, Orlando, USA, 2006, pp. 2017–2022.

[15] T. Rybus, T. Barciński, J. Lisowski, K. Seweryn, J. Nicolau-Kukliński, J. Grygorczuk, M. Krzewski, K. Skup, T. Szweczyk, and R. Wawrzaszek, "Experimental Demonstration of Singularity Avoidance with Trajectories Based on the Bézier Curves for Free-Floating Manipulator," in *Proc. IEEE 9th Int. Workshop Robot Motion and Control*, Kuslin, Poland, 2013, pp. 141–146.

[16] T. Osa, A. M. G. Esfahani, R. Stolkin, R. Lioutikov, J. Peters, and G. Neumann, "Guiding Trajectory Optimization by Demonstrated Distributions," *IEEE Robot. Autom. Lett.*, vol. 2, no. 2, pp. 819–826, 2017.

[17] N. Ratliff, M. Zucker, J. A. Bagnell, and S. Srinivasa, "CHOMP: Gradient Optimization Techniques for Efficient Motion Planning," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA'09)*, Kobe, Japan, 2009, pp. 489–494.

[18] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, and S. Schaal, "STOMP: Stochastic Trajectory Optimization for Motion Planning," in *IEEE Int. Conf. Robotics and Automation (ICRA'11)*, Shanghai, China, 2011, pp. 4569–4574.

[19] C. Park, J. Pan, and D. Manocha, "ITOMP: Incremental Trajectory Optimization for Real-Time Replanning in Dynamic Environments," in *Int. Conf. Automated Planning and Scheduling (ICAPS'12)*, São Paulo, Brazil, 2012, pp. 207–215.

[20] J. Schulman, J. Ho, A. X. Lee, I. Awwal, H. Bradlow, and P. Abbeel, "Finding Locally Optimal, Collision-Free Trajectories with Sequential Convex Optimization," in *Proc. Robotics: Science and Systems (RSS)*, Berlin, Germany, 2013.

[21] A. Hourtash, "The Kinematic Hessian and Higher Derivatives," in *Proc. IEEE Int. Symp. Computational Intelligence in Robotics and Automation (CIRA'05)*, Espoo, Finland, 2005, pp. 169–174.

[22] P. Beeson and B. Ames, "TRAC-IK: An Open-Source Library for Improved Solving of Generic Inverse Kinematics," in *Proc. IEEE Int. Conf. Humanoid Robots (Humanoids'15)*, Seoul, South Korea, 2015, pp. 928–935.