IEEE Transactions on Robotics (T-RO) paper presented at
2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Smooth Distances for Second Order Kinematic Robot Control

Vinicius Mariano Gonçalves, Anthony Tzes, Farshad Khorrami and Philippe Fraisse

***Abstract*—In this paper, we propose an algorithm for computing a smoothed version of the distance between two objects. As opposed to the traditional Euclidean distance between two objects, which may not be differentiable, this smoothed distance is guaranteed to be differentiable. Differentiability is an important property in many applications, in particular in robotics, in which obstacle-avoidance schemes often rely on the derivative/Jacobian of the distance between two objects. We prove mathematical properties of this smoothed distance and of the algorithm for computing it, and show its applicability in robotics by applying it to a second order kinematic control framework, also proposed in this paper. The control framework using smooth distances was successfully implemented on a 7 DOF manipulator.**

I. Introduction

The computation of distance between objects is very important for generating safe motion in robotics [1], since it is used as a measure of how close two objects are to colliding. Therefore, much effort has been put in developing strategies for distance computation. These strategies can either: (i) compute objects that encapsulate other objects so this new bounding object has interesting distance properties [2]–[5], (ii) define new measures of distance, other than the frequently used "Euclidean distance", that have interesting properties [6]–[11] or even (iii) propose improvements to algorithms that compute the Euclidean distance [12].

Several collision-avoidance schemes rely on the Jacobian of the distance function in the system configuration q [13]. However, this Jacobian may not exist since the distance function is not always differentiable. For example, considering the Euclidean distance between two objects $\mathcal{A}(q)$ and $\mathcal{B}(q)$ in the space, the *witness points* are the set of points $(a^*(q), b^*(q))$, with $a^*(q) \in \mathcal{A}$, $b^*(q) \in \mathcal{B}$, such that the Euclidean distance between the objects corresponds to the Euclidean distance between these points. If this pair of points is not unique in a configuration q, then the distance Jacobian may not exist [4] (see Figure 1), which can cause several problems in the control algorithms [4]. Furthermore, some control algorithms, such as the ones that work with acceleration/torque, necessitate the use of a distance function with a continuous *second* derivative.

One solution for the problem is to consider either two convex objects or at least one being strictly convex. [4] shows, rediscovering a previous result in [14], that in this case the witness points pairs are always unique and thus differentiability is ensured. Consequently, [4] proposes an algorithm to compute strictly convex objects that bound other objects. This approach falls in the category (i) listed before.

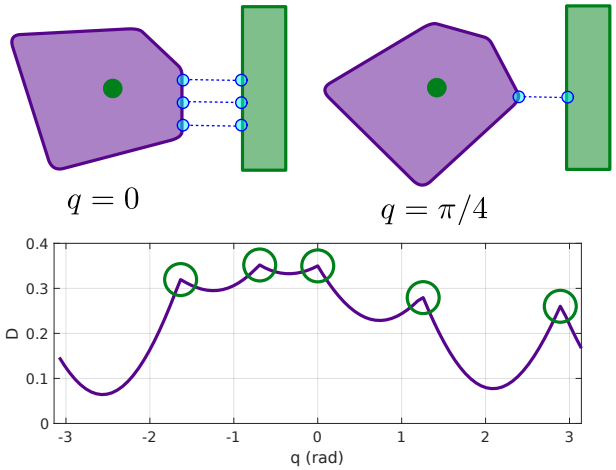


Fig. 1: Example of non-differentiability. The purple object rotates counterclockwise with an angle q around its center (green point). The Euclidean distance D between it and the green object is shown in the plot as a function of the angle q. Note some non-differentiable points. This happens whenever there are more than one witness points (blue points) between the two objects (e.g., when $q = 0$). It is differentiable, however, whenever the point is unique (e.g., when $q = \pi/4$).

We handle this problem using an approach in the category (ii): we propose a new "distance" metric between objects that is guaranteed to be always smooth. This metric is parameterized by four positive numbers (h, g, R, S) (two for each object) and we show that when $h \to 0$, $g \to 0$, we recover the usual Euclidean distance between the objects. When $h, g > 0$ the distance function is guaranteed to be smooth, that is, infinitely differentiable, as long as the two objects are convex (no strict convexity is required). The parameters (h, g, R, S) provide a trade-off between smoothness of the derivative/faster convergence rates and being close to the real distance. We say "distance" metric because it is not truly a distance, for example, it is not guaranteed to vanish when the objects intersect. Thus, it does not fall in the aforementioned category (i), because it is not computing the distance to a deformed version of the original objects. Even though it does not exactly

V. M. Gonçalves is with Center for Artificial Intelligence and Robotics (CAIR), New York University Abu Dhabi, United Arab Emirates
A. Tzes is with New York University Abu Dhabi, Electrical Engineering, Abu Dhabi 129188, United Arab Emirates
F. Khorrami is with New York University, Electrical & Computer Engineering Department, Brooklyn, NY 11201, USA.
P. Fraisse is with Laboratoire d'informatique, de robotique et de microelectronique de Montpellier (LIRMM) Montpellier, France.

vanish when the two objects intersect, it is still useful for obstacle avoidance because we can use safety margins.

With this new metric, we can compute the "(h, g, R, S) witness points" with an iterative algorithm based on a modified form of *von Neumann's projection algorithm*. One of the benefits of the approach is that no specialized solver is needed: the algorithm only relies on the computation of an operator that we call (h, R)-*projection*. Another benefit is that we can easily control the trade-off between being smooth and close to the traditional distance and being smooth/having faster convergence by tuning the parameters h, R. This (h, R)-projection is computed by solving integrals, exploiting a connection between computing "extreme" integrals over a region and optimization in this same region. This integral often cannot be done analytically, but we provide very good approximations for common objects (spheres, boxes and cylinders) and a workaround for more general objects (using sampled points). The algorithm is fast and easy to implement. We provide several formal guarantees for the proposed "distance" metrics and the modified von Neumann's projection algorithm, including convergence guarantees for the algorithm, which is iterative.

Once we define this point-to-set distance and the associated witness points, we define a smoothed set-to-set distance function. If these sets move as a function of a configuration q, this set-to-set function is ultimately a function of the configuration q. In the important case in which the objects move according to rigid motions, we show how to compute the gradient of this function on the variable q analytically, and further derivatives of this function exist and can be computed numerically. Furthermore, the trade-off between how accurate is this distance computation in comparison to the original Euclidean distance and how large are the derivatives can be controlled using the parameters h and R. It is important to mention that, as far as the goal of smoothing the set-to-set distance is concerned, we have other approaches as well. For example, the aforementioned work [11] proposes a sampling-based approach to smooth a function f and its gradient, the *bundled objective/gradient*. Thus, an alternative to what we propose in this paper is to use this smoothing approach into the original (non-smoothed) set-to-set function. However, the bundled objective/gradient approach can be inefficient in this particular case, because, in order to compute the smoothed function/gradient at a single configuration q, it requires the evaluation of the original set-to-set distance/gradient on many different configurations around q. Although one such evaluation can be relatively fast, many of them, for generating a single evaluation of the smooth set-to-set distance/gradient, can be prohibitive. In comparison, our computation for the function/gradient requires only the computation of the (smoothed) witness points at the desired configuration q.

We also provide a second order kinematic robotic control framework in which the joints' accelerations are computed by solving a convex quadratic programming (QP) problem. The framework allows the insertion of inequality constraints that can be used to enforce safety constraints to account for obstacle avoidance. QP formulations are a popular choice for computing controllers with safety constraints (see [15], [16], [17] to cite a few that work with robotics manipulators), mainly because they are versatile enough while being relatively easy to solve. The obstacle avoidance in this framework requires that the distance function is not only twice differentiable, but the second derivative should not be too large, because high values can cause disturbances on the control loop or even render the QP infeasible. How large this second derivative may be is related to many factors such as the maximum actuation capabilities of the robot, controller parameters, the geometry of the environment and the current state of the system. Regardless, these characteristics make the proposed smooth distance very suitable for the proposed control framework, since we can explicitly control the smoothing parameters. We also provide Lyapunov stability results for this framework and an experiment of its application on a 7 Degree-of-Freedom (DoF) manipulator performing motions while considering obstacles.

While the proposed control framework depends on the proposed smooth distance, it offers independent value. For instance, many path-planning methods are formulated as optimization problems, and many optimization techniques (e.g., BFGS) benefit when all functions involved in the optimization problem (such as distance) are twice differentiable. Such formulations could benefit of the proposed distance.

II. Mathematical Notation and Basic Results

We denote by $\mathbb{R}^+$ the set of nonnegative real numbers. All vectors are column vectors. $I_{n\times n}$ denotes the $n \times n$ identity matrix; 1_n denotes the column vector with n entries in which all entries are 1 and $0_{n\times m}$ the $n\times m$ matrix of zeros. $S : \mathbb{R}^3 \mapsto \mathbb{R}^{3\times 3}$ is the map that implements the cross product, that is, for two vectors $u, v \in \mathbb{R}^3$, we have $u \times v = S(u)v$. If $v \in \mathbb{R}^n$ is a vector, $\|v\|$ is its Euclidean norm. If $f(t)$ is a function of time, then $\dot{f}(t) \triangleq \frac{d}{dt}f(t)$. The (Euclidean) induced norm of a matrix M, denoted by $\|M\|$, is the square root of the largest eigenvalue of M^TM. In particular, if M is symmetric and positive semi-definite, this is just the largest eigenvalue of M.

If $f(u) : \mathbb{R}^n \mapsto \mathbb{R}^m$ is a function, $\frac{\partial f}{\partial u}$ is its *Jacobian*, an $n \times m$ matrix in which the entry in ith-row and jth-column is $\frac{\partial f_i}{\partial u_j}(u)$. If $f : \mathbb{R}^n \mapsto \mathbb{R}$, $\frac{\partial^2 f}{\partial u^2}$ is its *Hessian*, an $n \times n$ matrix in which the (i, j)th entry is $\frac{\partial^2 f}{\partial u_i \partial u_j}(u)$. The function $v = \log(u)$ is the *natural logarithm* of u (the inverse of $u = e^v$). For $z \in \mathbb{R}$, *Erf*$(z) \triangleq \frac{2}{\sqrt{\pi}} \int_0^z e^{-t^2} dt$ is the *error function* [18].

If $\mathcal{R} \subseteq \mathbb{R}^n$ is a set, and β is a scalar, we define the set $\beta\mathcal{R} = \{\beta r \mid r \in \mathcal{R}\}$. ∂R is the $(n-1)$-dimensional boundary of the set. *chull*$(\mathcal{R})$ is the *convex hull* of the set $\mathcal{R}$. If $g \in \mathbb{R}^n$ is an integration variable, we define as dg the respective volume element in $\mathbb{R}^n$. If $f : \mathbb{R}^n \mapsto \mathbb{R}^n$ and $\mathcal{R} \subseteq \mathbb{R}^n$, f is extended to sets as $f(\mathcal{R}) = \{f(p) \mid p \in \mathcal{R}\}$.

A function $f : \mathbb{R}^n \mapsto \mathbb{R}$ is *convex* if for all $p', p'' \in \mathbb{R}^n$ and $\lambda \in [0, 1]$, $f(\lambda p' + (1 - \lambda)p'') \leq \lambda f(p') + (1 - \lambda)f(p'')$. A function is *concave* if $-f$ is convex. A nonnegative function is *log-concave* if $\log(f)$ is concave, or, equivalently, if $f(\lambda p' + (1 - \lambda)p'') \geq f(p')^\lambda f(p'')^{1-\lambda}$. *Strict convexity, strict concavity* or *strict log-concavity* is obtained when the respective equality is achieved if and only if $\lambda = 0$ or $\lambda = 1$. Furthermore, a function is σ *strongly convex* if for a finite

$\sigma > 0$ we have that $f(p) - \|p\|^2/(2\sigma)$ is convex, σ *strongly concave* if $-f$ is strongly convex and σ *strongly log-concave* if $\log(f)$ is σ strongly concave. We will sometimes drop the term σ and just say strongly convex, strongly concave or strongly log-concave

If $A \in \mathcal{A} \subseteq \mathbb{R}^n$ is a random vector with probability density $\rho : \mathcal{A} \mapsto \mathbb{R}^+$, then $\mathbb{E}[A]$ is its *expected value*, $\mathbb{E}[A] = \int_{\mathcal{A}} \rho(a) a da$. We also define its *covariance matrix*, $Cov[A] = \mathbb{E}[AA^\top] - \mathbb{E}[A]\mathbb{E}[A]^\top$, in which $\mathbb{E}[AA^\top] = \int_{\mathcal{A}} \rho(a) aa^\top da$.

A subset $\mathcal{A}$ of $\mathbb{R}^n$ is called *standard* if it is: a) compact (bounded and closed in $\mathbb{R}^n$), b) its n-dimensional volume is non-zero, and c) has a piecewise smooth boundary. So, a flat bidimensional disk in $\mathbb{R}^3$ is not standard, because its 3D volume is 0. The set $\mathcal{A} = (0, 1)$ in $\mathbb{R}$ is also not standard, because it is not closed, hence not compact.

The following results will be used thorough this paper:

Result 1. ***(Properties of eingevalues)*** *[19] If $M \in \mathbb{R}^{n\times n}$ is a square matrix with eigenvalues λ_i and γ is a scalar, then the matrix $M - \gamma I_{n\times n}$ has eigenvalues $\lambda_i - \gamma$. Similarly, the eigenvalues of $\gamma I_{n\times n} - M$ are $\gamma - \lambda_i$.*

Result 2. ***(A consequence of an extension of Prékopa–Leindler's inequality)*** *[20] Let $f, g : \mathbb{R}^n \times \mathbb{R}^+$ be two σ_f and σ_g strongly log-concave functions, respectively. Then their convolution $H(u) = \int_{\mathbb{R}^n} f(v)g(u - v)dv$ is $(\sigma_f + \sigma_g)$ strongly log-concave on u.*

Result 3. ***(Laplace's principle of large deviations, simplified version)*** *[21] Let $\mathcal{R} \subseteq \mathbb{R}^n$ be a standard set , $W : \mathcal{R} \mapsto \mathbb{R}$ a continuous positive function and $f : \mathcal{R} \mapsto \mathbb{R}$ a continuous function. Then: $\lim_{t\to 0+} -t \log\left(\int_{r\in\mathcal{R}} W(r)e^{-f(r)/t}dr\right) = \min_{r\in\mathcal{R}} f(r)$.*

Result 4. ***(Banach's fixed point theorem)*** *[22] Let $f : \mathbb{R}^w \mapsto \mathbb{R}^w$ be a* contractible *operation, i.e., there exists a scalar $0 < \alpha < 1$ such that $\forall x', x'' \in \mathbb{R}^w$ we have $\|f(x') - f(x'')\| \leq \alpha\|x' - x''\|$. Then the fixed point equation $x = f(x)$ has a unique solution, and can be computed by the sequence $x[k + 1] = f(x[k])$ for any $x[0] \in \mathbb{R}^w$.*

Result 5. ***(Properties of the S matrix)***

- *(i) If $u \in \mathbb{R}^3$, $S(u)^\top = -S(u)$;*
- *(ii) If $u, v \in \mathbb{R}^3$, $S(u)v = -S(v)u$;*
- *(iii) If $Q(t)$ is a time-variant rotation matrix from frame $\mathcal{F}_A$ to $\mathcal{F}_B$ with associated angular velocity vector $\omega(t) \in \mathbb{R}^3$ (measured in the frame $\mathcal{F}_A$), then $\dot{Q}(t) = S(\omega(t))Q(t)$.*

III. Main Result

A. The alternating projection algorithm

Definition 1. ***(Projection function into a set)*** *Let $\mathcal{A} \subseteq \mathbb{R}^n$ be a standard convex set. Define the* projection function into $\mathcal{A}$, $\Pi^{\mathcal{A}} : \mathbb{R}^n \mapsto \mathcal{A}$ *as*

$$\Pi^{\mathcal{A}}(p) \triangleq \arg\min_{a\in\mathcal{A}} \|p - a\|^2. \tag{1}$$

This function simply maps a point p into its closest point in $\mathcal{A}$. Since we assume that the set is convex, the minimizer is unique [4] and thus $\Pi^{\mathcal{A}}(p)$ is well defined (i.e., there isn't more than a single point in $\mathcal{A}$ that can be "the closest" to a point p).

Definition 2. ***(Half-squared distance between two sets)*** *Let $\mathcal{A}, \mathcal{B} \subseteq \mathbb{R}^n$ be two standard sets. Define the* half-squared distance between the sets *as*

$$\mathbb{D}^{\mathcal{A},\mathcal{B}} \triangleq \min_{a\in\mathcal{A}} \min_{b\in\mathcal{B}} \frac{1}{2}\|a - b\|^2. \tag{2}$$

Half-squared distances instead of distances will be used throughout his paper; the conventional distance can be easily recovered by computing $\sqrt{2\mathbb{D}^{\mathcal{A},\mathcal{B}}}$.

Let $\mathcal{A}, \mathcal{B} \subseteq \mathbb{R}^n$ be two standard convex sets, then the sequence

$$a[k + 1] = \Pi^{\mathcal{A}}(b[k]);\ b[k + 1] = \Pi^{\mathcal{B}}(a[k + 1]) \tag{3}$$

is such that , $\forall b[0]$, converges to one of the possible pair of points a^*, b^* such that [23] $\mathbb{D}^{\mathcal{A},\mathcal{B}} = \frac{1}{2}\|a^* - b^*\|^2$.

The algorithm in (3) is called *(von Neumann's) alternating projection algorithm* [24] to compute the closest points between two standard convex sets. The algorithm may not converge to a unique pair of points (a^*, b^*) for given sets $\mathcal{A}, \mathcal{B}$ since such points are not always unique.

The non-uniqueness of the points (a^*, b^*), also called *witness points* [4], can cause several problems. In robotics, the sets $\mathcal{A}$ and/or $\mathcal{B}$ can vary continuously with a parameter q. For example, consider that both sets are different links of one robotic manipulator and q is its configuration. In this case, to avoid collision between the links, we can use the Jacobian of the distance function between the sets/links, but this Jacobian may not be continuous if the witness points are not unique. This can cause several problems in the control schemes that use these Jacobians to avoid collisions between the links [13].

The witness points are unique as long as at least one of the sets is not only standard and convex, but standard and strictly convex [4]. To alleviate this restriction, we will search for a different version of the "distance" that is guaranteed to be smooth for all sets. This is the first objective of this article.

B. (h,R)-distances

Definition 3. ***(Half-squared distance from a point to a set)*** *Let $\mathcal{A} \subseteq \mathbb{R}^n$ be a standard (not necessarily convex) set. Define the function $D^{\mathcal{A}} : \mathbb{R}^n \mapsto \mathbb{R}^+$ that computes the half-squared closest distance between a point $p \in \mathbb{R}^n$ and the set: $D^{\mathcal{A}}(p) \triangleq \min_{a\in\mathcal{A}} \frac{1}{2}\|p - a\|^2$.*

Unfortunately, it is well known that this function is not differentiable in p if there is more than one point a^* such that $D^{\mathcal{A}}(p) = \frac{1}{2}\|p - a^*\|^2$ [4]. If there is a single point $a^*(p)$, then $\frac{\partial}{\partial p} D^{\mathcal{A}}(p) = p - a^*(p) = p - \Pi^{\mathcal{A}}(p)$ [25]. Aiming to solve this differentiability issue, we will soon define a smooth version of $D^{\mathcal{A}}(p)$. But first, we define the following:

Definition 4. ***(Weight function)*** *Let $\mathcal{A}$ be a standard set. Let Cen($\mathcal{A}$) denote the* centroid *of $\mathcal{A}$, that is, Cen($\mathcal{A}$) $\triangleq \int_{\mathcal{A}} ada / \int_{\mathcal{A}} da$. Furthermore, let R be a positive number. We*

define the weight function $W_R^{\mathcal{A}}(a) \triangleq e^{-\|a-Cen(\mathcal{A})\|^2/(2R^2)}$. *Finally, we define the* R-volume *as* $Vol_R(\mathcal{A}) \triangleq \int_{\mathcal{A}} W_R^{\mathcal{A}}(a)da$.

We can check, by inspection, that the function W has the following properties:

- $W_R^{\mathcal{A}}$ is R^2 strongly log-convex in a for any finite $R > 0$;
- It is consistent to scalings of the data of the problem, that is, for any $\beta > 0$, $W_{\beta R}^{\beta\mathcal{A}}(\beta a) = W_R^{\mathcal{A}}(a)$. This mean that if we scale everything $(\mathcal{A}, a, R)$ by β, nothing changes.
- It is consistent to rigid transformations applied on the data of the problem. Let $E : \mathbb{R}^n \mapsto \mathbb{R}^n$ be a rigid transformation on $\mathbb{R}^n$, then $W_R^{E(\mathcal{A})}(E(a)) = W_R^{\mathcal{A}}(a)$, that is, if we rotate/translate the set $\mathcal{A}$ but rotate/translate the point a accordingly, nothing changes.

With this definition, the following key definition can be presented:

Definition 5. ***((h,R)-half-squared distance from a point to a set)*** *Let $\mathcal{A} \subseteq \mathbb{R}^n$ be a standard set, a scalar $h > 0$, called* smoothing scale parameter*, and a scalar $R > 0$, called* regularization parameter*. Then we define the h, R-* (half-squared) distance to a standard set $\mathcal{A}$, $D_{h,R}^{\mathcal{A}} : \mathbb{R}^n \mapsto \mathbb{R}^+$ *as*

$$D_{h,R}^{\mathcal{A}}(p) \triangleq -h^2 \log\left(Vol_R(\mathcal{A})^{-1} \int_{\mathcal{A}} W_R^{\mathcal{A}}(a) e^{-\frac{\|p-a\|^2}{2h^2}} da\right). \tag{4}$$

It may not be obvious that this approximates the half-squared distance to a set, nor that it is always nonnegative, $D_{h,R}^{\mathcal{A}}(p) \geq 0$. Note also that if $Vol_R(\mathcal{A}) = 0$ we have an ill-defined $D_{h,R}^{\mathcal{A}}(p)$ (an indeterminate $0/0$). This is one of the reasons why we assume that the set is standard, since having a non-null volume in $\mathbb{R}^n$ implies $Vol_R(\mathcal{A}) \neq 0$. Sometimes, however, we may need to compute the h, R-distance between a point and a set with $Vol_R(\mathcal{A}) = 0$ in $\mathbb{R}^n$. For example, a distance from a point in $\mathbb{R}^3$ to a bidimensional semiplane in $\mathbb{R}^3$. In that case, we simply enlarge this set so that $Vol_R(\mathcal{A}) \neq 0$.

The division by $Vol_R(\mathcal{A})$ inside the integral is important because: 1) it allows dimensional consistency: the integral has a physical dimension of n-dimensional volume (say, cubic meters in $\mathbb{R}^3$) and the logarithm is a transcendental function, thus its argument must be dimensionless. We obtain this when we divide by $Vol_R(\mathcal{A})$ (we consider that the weight function is dimensionless). Note that h has a physical dimension of distance (say, meters), and thus $D_{h,R}^{\mathcal{A}}(p)$ has a physical dimension of distance squared (say, meters square); 2) this division by $Vol_R(\mathcal{A})$ also allows us scale consistency, that is, the property that, for any scalar $\beta > 0$, $D_{\beta h,\beta R}^{\beta\mathcal{A}}(\beta p) = \beta^2 D_{h,R}^{\mathcal{A}}(p)$ (i.e., when we scale everything: the set, the point and the smoothing scale parameter by β, the distance squared gets scaled by β^2 as well), whereas if the division by $Vol_R(\mathcal{A})$ was removed we would have $D_{\beta h,\beta R}^{\beta\mathcal{A}}(\beta p) = \beta^2 D_{h,R}^{\mathcal{A}}(p) - \beta^2 h^2 \log(\beta)$, and 3) it guarantees, as we will prove soon, that $D_{h,R}^{\mathcal{A}}(p) \geq 0$.

The term inside the logarithm can be interpreted as the *weighted average* of the function $e^{-\frac{\|p-a\|^2}{2h^2}}$ among the set $\mathcal{A}$ for a fixed p, with weighting function $W_R^{\mathcal{A}}$. This observation implies that we can approximate the integral as a finite sum, as explained in Subsection IV-B.

Example 1: Let us compute $D_{h,R}^{\mathcal{A}}(p)$ for the standard (in $\mathbb{R}$) and convex set $\mathcal{A} = [-1, 1]$. Even for this simple case, we cannot compute the integral in (4) analytically and we need to consider special functions or use numerical methods, as discussed in Subsection IV-A.

Special functions will be used in this case. Furthermore, we will consider the case in which the regularization factor R is a very large number, so $W_R^{\mathcal{A}}(a) \approx 1$. Indeed, as we will discuss soon, we will usually work with high regularization factors. Nevertheless, we need to use the special function *error function* $Erf(z)$ and after some changes of variables in the integral, we compute the following expression for $D_{h,R}^{\mathcal{A}}(p)$, using the properties of the error function presented in [18]:

$$-h^2 \log\left(h\frac{\sqrt{2\pi}}{4}\left(Erf\left(\frac{p+1}{\sqrt{2}h}\right) - Erf\left(\frac{p-1}{\sqrt{2}h}\right)\right)\right). \tag{5}$$

The real half-squared distance function in this case can be obtained analytically, $D^{\mathcal{A}}(p) = \frac{1}{2}\max(|p| - 1, 0)^2$. Figure 2 shows the graph of $D_{h,R}^{\mathcal{A}}(p)$ for many values of h. Note that, as $h \to 0$, $D_{h,R}^{\mathcal{A}}(p) \to D^{\mathcal{A}}(p)$.

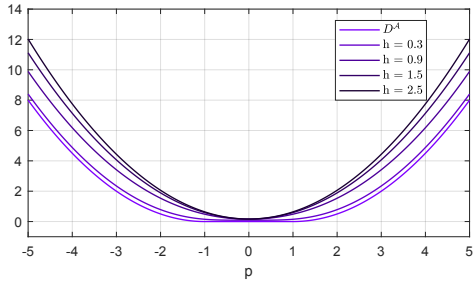


Fig. 2: Function $D_{h,R}^{\mathcal{A}}(p)$ for different values of h.

It is also important to mention that (5) exposes a numerical problem. The error function is practically 1 (with a difference of 10^{-12}) for $z > 5$ and -1 (with a difference of 10^{-12}) if $z < -5$. For $h = 0.1$, $p = 6$, for example, we have to compute $Erf(35.35)$ and $Erf(49.49)$ to compute $D_{h,R}^{\mathcal{A}}(p)$, which are indistinguishable from 1. Thus, we will have $\log(0)$ in the formula, which is an error. This is because $D_{h,R}^{\mathcal{A}}(p)$ often needs *extreme* numerical precision for computing correctly the values. This problem is addressed in Subsection IV-A. □

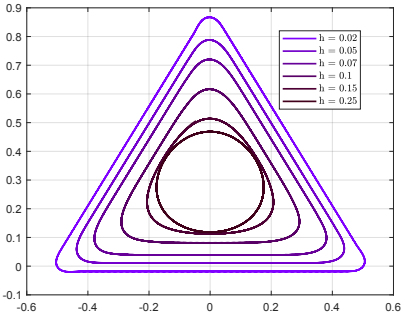


Fig. 3: Level sets for the function $D_{h,R}^{\mathcal{A}}(p)$ for different values of h.

For a two-dimensional example, Figure 3 shows level sets for $D_{h,R}^{\mathcal{A}}(p)$, that is, the sets $\mathcal{L}_h \triangleq \{p \in \mathbb{R}^2 \mid D_{h,R}^{\mathcal{A}}(p) =$

$c_h\}$, for $\mathcal{A}$ being an equilateral triangle with vertices $(-0.5, 0), (0, \sqrt{3}/2), (0.5, 0)$. In this case, different values of h were chosen, and R was fixed to a very large number. The values of c_h were chosen differently for each h, so the level sets do not overlap with each other, facilitating the visualization. Note that, as h increases, the level sets become smoother, i.e., the vertices become smoothed and for very large h, the shape approaches a circle. On the other hand, c_h for $h = 0.02$ was chosen very close to 0. Note then that the respective set $\mathcal{L}_h$ is very close to the original set $\mathcal{A}$. This illustrates the claim, that will be established soon, that as $h \to 0$, $D^{\mathcal{A}}_{h,R}(p) \to D^{\mathcal{A}}(p)$.

We will now establish some results about $D^{\mathcal{A}}_{h,R}(p)$:

Proposition 1. *Suppose $\mathcal{A}$ is a standard set. For any $h > 0$, $D^{\mathcal{A}}_{h,R}(p) \geq D^{\mathcal{A}}(p) \geq 0$.*

Proof. Let $a^*(p)$ be such that $D^{\mathcal{A}}(p) = \frac{1}{2}\|a^*(p) - p\|^2$. Factoring out $e^{-\|a^*(p)-p\|^2/(2h^2)}$ inside the logarithm in (4), we have $D^{\mathcal{A}}_{h,R}(p) - D^{\mathcal{A}}(p)$ as

$$-h^2 \log \left(\mathit{Vol}_R(\mathcal{A})^{-1} \int_{\mathcal{A}} W^{\mathcal{A}}_R(a) e^{-\frac{\|p-a\|^2 - \|p-a^*(p)\|^2}{2h^2}} da \right). \quad (6)$$

Note that $\|p-a\|^2 - \|p-a^*(p)\|^2 \geq 0$, $\forall a \in \mathcal{A}$. Multiplying both sides of the inequality by $-1/(2h^2)$, taking the exponential in both sides, multiplying by $W^{\mathcal{A}}_R(a)$ both sides, integrating in $\mathcal{A}$ in both sides, multiplying by $\mathit{Vol}_R(\mathcal{A})^{-1}$ in both sides and finally taking $-h^2 \log(\cdot)$ in both sides, we have that:

$$-h^2 \log \left(\mathit{Vol}_R(\mathcal{A})^{-1} \int_{\mathcal{A}} W^{\mathcal{A}}_R(a) e^{-\frac{\|p-a\|^2 - \|p-a^*(p)\|^2}{2h^2}} da \right) \quad (7)$$

is nonnegative, which concludes the proof. □

Proposition 2. *If $\mathcal{A}$ is a standard set, then* $\lim_{h\to 0} D^{\mathcal{A}}_{h,R}(p) = D^{\mathcal{A}}(p)$.

Proof. Since we can write $D^{\mathcal{A}}_{h,R}(p)$ as

$$h^2 \log\left(\mathit{Vol}_R(\mathcal{A})\right) - h^2 \log \left(\int_{\mathcal{A}} W^{\mathcal{A}}_R(a) e^{-\frac{\|p-a\|^2}{2h^2}} da \right). \quad (8)$$

The first term will obviously vanish when $h \to 0$. For the second one, we just need to apply Laplace's large deviation principle (Result 3) with $f(a) = \|a-p\|^2/2$, $t = h^2$ and $W(a) = W^{\mathcal{A}}_R(a)$, noting that, as $h \to 0$, $t \to 0+$. □

Since $\lim_{h\to 0} D^{\mathcal{A}}_{h,R}(p) = D^{\mathcal{A}}(p)$, $D^{\mathcal{A}}_{h,R}(p)$ is an approximation to the real (half-squared) distance between a point p and the set $\mathcal{A}$. Note that the result holds true for any R, but obviously R influences the approximation as well. Essentially, as we shall see, if we fix $h, \mathcal{A}$ and decrease R, we increase the convergence speed of the algorithm but makes $D^{\mathcal{A}}_{h,R}(p)$ a less faithful approximation of $D^{\mathcal{A}}(p)$.

Proposition 3. *Suppose $\mathcal{A}$ is a standard convex set. Then, the function $D^{\mathcal{A}}_{h,R}$ is $(1 + R^2/h^2)$ strongly convex.*

Proof. The function $f(u) \triangleq e^{-\|u\|^2/(2h^2)}$ is clearly h^2 strongly log concave. Consider the indicator function such that $I^{\mathcal{A}}(p) = 1$ if $p \in \mathcal{A}$ and 0 otherwise. We can show that, since $\mathcal{A}$ is convex, this function is log-concave. Also, $W^{\mathcal{A}}_R(v) = e^{-\|v - \mathit{Cen}(\mathcal{A})\|^2/(2R^2)}$ is R^2 strongly log-concave, and consequently the product of a log-concave function and a R^2 strongly log-concave function given by $g(v) \triangleq W^{\mathcal{A}}_R(v) I_{\mathcal{A}}(v)$ is also R^2 strongly log-concave [20]. Thus, applying Result 2, we have that the convolution $H(p) \triangleq \int_{\mathcal{A}} W^{\mathcal{A}}_R(a) e^{-\frac{\|p-a\|^2}{2h^2}} da$ (in which the integral from the entire $\mathbb{R}^n$ from the convolution was switched to the integral only on $\mathcal{A}$ thanks to the indicator function) is $(h^2 + R^2)$ strongly log-concave. This implies that $-h^2 H(p) = -h^2 \log\left(\mathit{Vol}_R(\mathcal{A})^{-1} H(p)\right) = D^{\mathcal{A}}_{h,R}(p)$ is $(h^2 + R^2)/h^2$ strongly convex in p. □

As said before, we will usually work with large values of R so, for the sake of convenience, $W^{\mathcal{A}}_R(a) \approx 1$. However, a finite value of R is needed to ensure σ strong convexity for a finite value of σ, which in turn will be used soon to guarantee the formal convergence of the algorithm.

We will now study the derivatives of $D^{\mathcal{A}}_{h,R}(p)$, we can see that

$$\frac{\partial}{\partial p} D^{\mathcal{A}}_{h,R}(p) = \frac{\int_{\mathcal{A}} W^{\mathcal{A}}_R(a) e^{-\frac{\|p-a\|^2}{2h^2}} (p-a) da}{\int_{\mathcal{A}} W^{\mathcal{A}}_R(a') e^{-\frac{\|p-a'\|^2}{2h^2}} da'}. \quad (9)$$

in which a' is just a renaming of the variable a in the integral of the denominator, which is convenient for defining the following:

$$\theta^{\mathcal{A}}_{h,R}(p,a) \triangleq \left(\int_{\mathcal{A}} W^{\mathcal{A}}_R(a') e^{-\frac{\|p-a'\|^2}{2h^2}} da' \right)^{-1} W^{\mathcal{A}}_R(a) e^{-\frac{\|p-a\|^2}{2h^2}}. \quad (10)$$

Then,

$$\frac{\partial}{\partial p} D^{\mathcal{A}}_{h,R}(p) = p - \int_{\mathcal{A}} \theta^{\mathcal{A}}_{h,R}(p,a) a da. \quad (11)$$

When $D^{\mathcal{A}}(p)$ is differentiable at a point p, we have that $\frac{\partial}{\partial p} D^{\mathcal{A}}(p) = p - \Pi^{\mathcal{A}}(p)$ [25]. Then, by analogy:

Definition 6. ***(h,R)- projection function into a set)*** *Let $\mathcal{A} \subseteq \mathbb{R}^n$ be a standard set and $h, R > 0$. Define the (h,R)-*projection function into $\mathcal{A}$, $\Pi^{\mathcal{A}}_{h,R} : \mathbb{R}^n \mapsto \mathit{chull}(\mathcal{A})$ *as*

$$\Pi^{\mathcal{A}}_{h,R}(p) \triangleq \int_{\mathcal{A}} \theta^{\mathcal{A}}_{h,R}(p,a) a da. \quad (12)$$

So $\frac{\partial}{\partial p} D^{\mathcal{A}}_{h,R}(p) = p - \Pi^{\mathcal{A}}_{h,R}(p)$. Note that $\forall h > 0$, $\Pi^{\mathcal{A}}_{h,R}(p)$ is well defined and thus $D^{\mathcal{A}}_{h,R}(p)$ *is differentiable* $\forall p$. Furthermore, note that $\Pi^{\mathcal{A}}_{h,R}$ maps p into the convex hull of $\mathcal{A}$, since the θ's are nonnegative and integrate to 1 (it is a parametrized density distribution in $\mathcal{A}$).

Note that, as opposed to the true projection $\Pi^{\mathcal{A}}$, we did not assume that the set $\mathcal{A}$ is convex in the definition of $\Pi^{\mathcal{A}}_{h,R}$. This is because $\Pi^{\mathcal{A}}_{h,R}$ is well defined regardless if $\mathcal{A}$ is convex or not. For convex sets, $\Pi^{\mathcal{A}}_{h,R}$ maps $\mathbb{R}^n$ into $\mathcal{A}$ because $\mathit{chull}(\mathcal{A}) = \mathcal{A}$.

Example 2: Let the aforementioned set $\mathcal{A} = [-1, 1]$, then using the error function and considering a large R so $W^{\mathcal{A}}_R \approx 1$:

$$\Pi^{\mathcal{A}}_{h,R}(p) = h \left(\frac{e^{-(1-p)^2/(2h^2)} - e^{-(1+p)^2/(2h^2)}}{\mathit{Erf}\left(\frac{p+1}{\sqrt{2}h}\right) - \mathit{Erf}\left(\frac{p-1}{\sqrt{2}h}\right)} \right). \quad (13)$$

Figure 4 shows different values of h, along with the real projection function, which is computed analytically:

$$\Pi^{\mathcal{A}}(p) = \begin{cases} -1 & \text{if } p \leq -1 \\ p & \text{if } |p| \leq 1 \\ 1 & \text{if } p \geq 1 \end{cases} \tag{14}$$

As mentioned in Example 1, there are also numerical problems when computing $\Pi^{\mathcal{A}}_{h,R}(p)$ because both the numerator and denominator can go to 0 when p are too far from the set and/or h is small. This numerical issue is addressed in Subsection IV-A. □

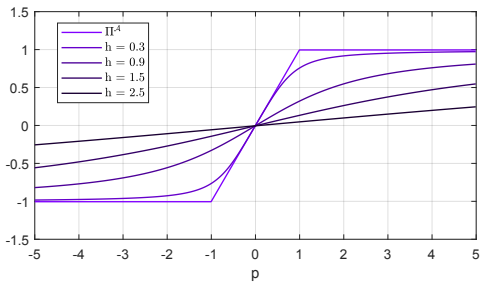


Fig. 4: Function $\Pi^{\mathcal{A}}_{h,R}(p)$ for different values of h.

An interesting probabilistic interpretation of $\Pi^{\mathcal{A}}_{h,R}(p)$ is as follows: Let $A_{h,R}(p)$ be a random vector taking values in $\mathcal{A}$ with probability density (parametrized by p and h) given by $\rho(a) = \theta^{\mathcal{A}}_{h,R}(p,a)$. Note that, for any fixed p and h, $\theta^{\mathcal{A}}_{h,R}(p,a)$ is indeed a probability density in $\mathcal{A}$, since it is nonnegative and $\int_{\mathcal{A}} \theta^{\mathcal{A}}_{h,R}(p,a)da = 1$, for any p, h. Then $\Pi^{\mathcal{A}}_{h,R}(p) = \mathbb{E}[A_{h,R}(p)]$.

Let us compute the Hessian of $D^{\mathcal{A}}_{h,R}(p)$, or $\frac{\partial^2}{\partial p^2} D^{\mathcal{A}}_{h,R}(p) = I_{n\times n} - \frac{\partial}{\partial p}\Pi^{\mathcal{A}}_{h,R}(p)$, so what we need to compute is the Jacobian of the projection map. After some algebra, we deduce that this Jacobian has also an interesting probabilistic interpretation using the previously-defined random variable $A_{h,R}(p)$:

$$\frac{\partial}{\partial p}\Pi^{\mathcal{A}}_{h,R}(p) = \frac{1}{h^2}\text{Cov}[A_{h,R}(p)]. \tag{15}$$

Equations (11) and (15) allows us to conclude by inspection that $D^{\mathcal{A}}_{h,R}$ is at least differentiable for a nonzero h. In fact:

Proposition 4. *Suppose $\mathcal{A}$ is a standard set and $h > 0$. Then $D^{\mathcal{A}}_{h,R}(p)$ (and thus $\Pi^{\mathcal{A}}_{h,R}(p)$) is infinitely differentiable (smooth).*

Proof. The integral inside $D^{\mathcal{A}}_{h,R}(p)$ is an integral of a smooth function in both variables p and a. Furthermore, the integral is always positive and finite (since $\mathcal{A}$ is compact) for a finite h, and since $\log(u)$ is infinitely differentiable for any $u > 0$, this implies that $D^{\mathcal{A}}_{h,R}(p)$ is infinitely differentiable. □

Note that Proposition 4 can "fail" when $h \to 0$. For example, in Equation (15) $\frac{\partial}{\partial p}\Pi^{\mathcal{A}}_{h,R}(p)$ can "blow up" in this case due to the division by $1/h^2$. This mirrors the fact that the real projection $\Pi^{\mathcal{A}}(p)$ may not be differentiable, even when $\mathcal{A}$ is convex.

The interpretation of the Jacobian of $\Pi^{\mathcal{A}}_{h,R}$ as a covariance, in (15), can be used to prove the following:

Proposition 5. *Suppose $\mathcal{A}$ is a standard set and $h > 0$. Then, $\forall p$, the eigenvalues of the Hessian of $D^{\mathcal{A}}_{h,R}(p)$ are less than 1.*

Proof. Proposition 4 guarantees that the Hessian is well defined. Now, it is a well known fact that covariance matrices are positive semi-definite, having null eigenvalues if and only if the random vector always lies in an $(n-1)$ dimensional affine space in $\mathbb{R}^n$ [26]. If $Vol_R(\mathcal{A}) \neq 0$, this is impossible. Then the covariance matrix is strictly positive definite and has eigenvalues strictly greater than 0. From this, using $\frac{\partial^2}{\partial p^2} D^{\mathcal{A}}_{h,R}(p) = I_{n\times n} - \frac{\partial}{\partial p}\Pi^{\mathcal{A}}_{h,R}(p)$ and Result 1, we can see that the eigenvalues of the Hessian of $D^{\mathcal{A}}_{h,R}(p)$ are less than 1. □

C. The (h,g,R,S)-alternating projection algorithm

We will now use the smoothed projections $\Pi^{\mathcal{A}}_{h,R}$ and $\Pi^{\mathcal{B}}_{g,S}$ to study an algorithm similar to the one in (3), but with smoothed projections instead of the usual projections. The sequence is then analogous to (3):

$$a[k+1] = \Pi^{\mathcal{A}}_{h,R}(b[k]); \quad b[k+1] = \Pi^{\mathcal{B}}_{g,S}(a[k+1]). \tag{16}$$

Note that we have (h,R)-projections and (g,S)-projections so not necessarily we need to have $h = g$ or $R = S$. We will now establish convergence for the sequence.

Proposition 6. *If the sets $\mathcal{A}$ and $\mathcal{B}$ are standard and convex and $h, g, R, S > 0$, (16) has a fixed point, and it is unique. Furthermore, for any initial condition $b[0]$, when $k \to \infty$ the sequence $(a[k], b[k])$ generated from it converges to this unique fixed point $(a^*_{h,R}, b^*_{g,S})$.*

Proof. From Proposition 5, the Hessian of $D^{\mathcal{A}}_{h,R}(p)$, which is $I_{n\times n} - \frac{\partial}{\partial p}\Pi^{\mathcal{A}}_{h,R}(p)$, has real eigenvalues that are less than 1. Thus, using Result 1, the eigenvalues of $\frac{\partial}{\partial p}\Pi^{\mathcal{A}}_{h,R}(p)$ are greater than 0 $\forall p$.

Now, we use the fact that $\mathcal{A}$ is convex, which together with Proposition 3 implies that $D^{\mathcal{A}}_{h,R}$ is a σ strongly convex function with $\sigma = 1+R^2/h^2$. Since additionally $D^{\mathcal{A}}_{h,R}$ is twice differentiable (Proposition 4), we have that the eigenvalues of the Hessian $D^{\mathcal{A}}_{h,R}(p)$ are at least $1/\sigma$ $\forall p$ [27]. Consequently applying Result 1, we have that the eigenvalues of $\frac{\partial}{\partial p}\Pi^{\mathcal{A}}_{h,R}(p)$ are less than or equal $1 - 1/\sigma$ $\forall p$, which implies that they are strictly less than one.

Since the Jacobian of the map $\Pi^{\mathcal{A}}_{h,R}(p)$ is a symmetric positive definite matrix, its induced norm $\|\Pi^{\mathcal{A}}_{h,R}(p)\|$ is just the maximum eigenvalue of $\Pi^{\mathcal{A}}_{h,R}(p)$, and thus is less than 1 $\forall p$. This implies that the map $p \mapsto \Pi^{\mathcal{A}}_{h,R}(p)$ is locally contractible $\forall p$, and thus a globally contractible operation [28].

The same results holds for $\Pi^{\mathcal{B}}_{g,S}$. Now, note that the sequence in (16) can be rewritten as $b[k+1] = \Pi^{\mathcal{B}}_{g,S}(\Pi^{\mathcal{A}}_{h,R}(b[k]))$. Since each map, for $\mathcal{A}$ and $\mathcal{B}$, are globally contractible, this implies that the composition map $\Pi^{\mathcal{B}}_{g,S}(\Pi^{\mathcal{A}}_{h,R}(p))$ is also globally contractible. *Banach's fixed point theorem* (Result 4) then implies that the sequence $b[k+1] = \Pi^{\mathcal{B}}_{g,S}(\Pi^{\mathcal{A}}_{h,R}(b[k]))$ converges to the unique fixed point of $\Pi^{\mathcal{B}}_{g,S}(\Pi^{\mathcal{A}}_{h,R}(p))$, $b^*_{g,S} = \Pi^{\mathcal{B}}_{g,S}(\Pi^{\mathcal{A}}_{h,R}(b^*_{g,S}))$. We can compute $a^*_{h,R}$ by $a^*_{h,R} = \Pi^{\mathcal{A}}_{h,R}(b^*_{g,S})$. □

The proposed (modified) alternating projection algorithm is a fixed point iteration. Thus, acceleration methods as Anderson acceleration [29] can be used to improve the convergence. However, for the sake of simplicity, in this paper we will keep the algorithm as close as possible to the original projection algorithm.

D. The smooth distance between sets

The functions $D^{\mathcal{A}}_{h,R}(p)$ computes h,R-distances between a *point* and *a set*. However, we are mainly interested in computing this approximated distance between *two sets*. One idea is to simply use $\|a^*_{h,R}-b^*_{g,S}\|^2/2$, but this function is not convenient for computing derivatives on a parameter q when $a^*_{h,R}, b^*_{g,S}$ depends on q, something that will be useful for our robotics application. As a preliminary step for defining this distance for sets, we define the following:

Definition 7. ***((h,g,R,S)-half-squared cross distance)*** *Let $\mathcal{A},\mathcal{B}$ be two standard sets in $\mathbb{R}^n$ and $h,g,R,S>0$. We define the (h,g,R,S)-* half-squared cross distance *as the function $\mathbb{d}^{\mathcal{A},\mathcal{B}}_{h,g,R,S}:\mathbb{R}^n\times\mathbb{R}^n\mapsto\mathbb{R}$:*

$$\mathbb{d}^{\mathcal{A},\mathcal{B}}_{h,g,R,S}(a,b)\triangleq D^{\mathcal{A}}_{h,R}(b)+D^{\mathcal{B}}_{g,S}(a)-\frac{1}{2}\|a-b\|^2. \quad (17)$$

This function has the following property, that will be useful soon:

Proposition 7. *Suppose $\mathcal{A}$ and $\mathcal{B}$ are standard sets and $h,g,R,S>0$. Then, the fixed point $(a^*_{h,R},b^*_{g,S})$ for the sequence (16) is a stationary point (i.e., the gradient vanishes) for the function $\mathbb{d}^{\mathcal{A},\mathcal{B}}_{h,g,R,S}(a,b)$.*

Proof. It is easy to see, using the fact that $\frac{\partial}{\partial p}D^{\mathcal{A}}_{h,R}(p)=p-\Pi^{\mathcal{A}}_{h,R}(p)$ and $\frac{\partial}{\partial p}D^{\mathcal{B}}_{g,S}(p)=p-\Pi^{\mathcal{B}}_{g,S}(p)$ that $\frac{\partial}{\partial a}\mathbb{d}^{\mathcal{A},\mathcal{B}}_{h,g,R,S}(a,b)=b-\Pi^{\mathcal{B}}_{g,S}(a)$ and $\frac{\partial}{\partial b}\mathbb{d}^{\mathcal{A},\mathcal{B}}_{h,g,R,S}(a,b)=a-\Pi^{\mathcal{A}}_{h,R}(b)$. Setting these two to zero we obtain exactly the equations for the fixed point of the sequence: $a=\Pi^{\mathcal{A}}_{h,R}(b)$ and $b=\Pi^{\mathcal{B}}_{g,S}(a)$. □

We define the following:

Definition 8. ***((h, g, R,S)-half-squared distance between sets)*** *Let $\mathcal{A}$, $\mathcal{B}$ be two standard convex sets and $h,g,R,S>0$. We define the (h,g,R,S)-* half-squared distance between the sets $\mathcal{A}$ and $\mathcal{B}$ *as*

$$\mathbb{D}^{\mathcal{A},\mathcal{B}}_{h,g,R,S}\triangleq\mathbb{d}^{\mathcal{A},\mathcal{B}}_{h,g,R,S}(a^*_{h,R},b^*_{g,S}). \quad (18)$$

Note that Proposition 6 allow us to define $\mathbb{D}^{\mathcal{A},\mathcal{B}}_{h,g,R,S}$, because it guarantees that $a^*_{h,R}$ and $b^*_{g,S}$ exist and are unique.

We can prove the following result that connects $\mathbb{D}^{\mathcal{A},\mathcal{B}}_{h,g,R,S}$ to the real half-squared distance between the sets.

Proposition 8. *Let $\mathcal{A},\mathcal{B}$ be two standard convex sets. Then $\lim_{(h,g)\to(0,0)}\mathbb{D}^{\mathcal{A},\mathcal{B}}_{h,g,R,S}=\mathbb{D}^{\mathcal{A},\mathcal{B}}$.*

Proof. Using Proposition 2, $\lim_{(h,g)\to(0,0)}D^{\mathcal{A}}_{h,R}(b^*_{g,S})=\|a^*-b^*\|^2/2$. Similarly, $\lim_{(h,g)\to(0,0)}D^{\mathcal{B}}_{g,S}(a^*_{h,R})=\|a^*-b^*\|^2/2$. Thus, we can conclude that $\lim_{(h,g)\to(0,0)}\mathbb{D}^{\mathcal{A},\mathcal{B}}_{h.g,R,S}$ is equal to $\frac{\|a^*-b^*\|^2}{2}+\frac{\|a^*-b^*\|^2}{2}-\frac{\|a^*-b^*\|^2}{2}=\mathbb{D}^{\mathcal{A},\mathcal{B}}$. □

So $\mathbb{D}^{\mathcal{A},\mathcal{B}}_{h,g,R,S}$ approximates $\mathbb{D}^{\mathcal{A},\mathcal{B}}$. However, the same could be said of $\|a^*_{h,R}-b^*_{g,S}\|^2/2$. We will see soon that Proposition 7, a property unique to $\mathbb{D}^{\mathcal{A},\mathcal{B}}$ and not to $\|a^*_{h,R}-b^*_{g,S}\|^2/2$, will be useful for simplifying derivatives.

E. Moving sets

Suppose the sets $\mathcal{A}(q)$ and $\mathcal{B}(q)$ vary smoothly with a parameter $q\in\mathbb{R}^s$. There is interest in computing the derivative, with respect to $q\in\mathbb{R}^s$, of the following function:

$$\Lambda(q)\triangleq\mathbb{D}^{\mathcal{A}(q),\mathcal{B}(q)}_{h,g,R,S}=\mathbb{d}^{\mathcal{A}(q),\mathcal{B}(q)}_{h,g,R,S}\Big(a^*_{h,R}(q),b^*_{g,S}(q)\Big). \quad (19)$$

The function $\Lambda(q)$ is the "smoothed" (half-squared) distance between the moving sets $\mathcal{A}(q)$ and $\mathcal{B}(q)$. We now seek to compute its derivative on the parameter q. Using the multidimensional chain rule:

$$\begin{aligned}\frac{\partial}{\partial q}\Lambda(q)=&\frac{\partial}{\partial q}\mathbb{d}^{\mathcal{A}(q),\mathcal{B}(q)}_{h,g,R,S}(a,b)\Big|_{a=a^*_{h,R}(q),b=b^*_{g,S}(q)}+\\&\frac{\partial a^*_{h,R}}{\partial q}(q)\frac{\partial}{\partial a}\mathbb{d}^{\mathcal{A}(q),\mathcal{B}(q)}_{h,g,R,S}\big(a^*_{h,R}(q),b^*_{g,S}(q)\big)+\\&\frac{\partial b^*_{g,S}}{\partial q}(q)\frac{\partial}{\partial b}\mathbb{d}^{\mathcal{A}(q),\mathcal{B}(q)}_{h,g,R,S}\big(a^*_{h,R}(q),b^*_{g,S}(q)\big)\end{aligned} \quad (20)$$

in which the first term means that first we evaluate the derivative of $\mathbb{d}^{\mathcal{A}(q),\mathcal{B}(q)}_{h,g,R,S}(a,b)$ only in q, keeping a,b constant, then we plug the optimal values $a=a^*_{h,R}(q)$, $b=b^*_{g,S}(q)$.

However, using Proposition 7, we can cancel the two last summands in (20), which greatly simplifies the expression. Therefore, computing $\frac{\partial}{\partial q}\mathbb{d}^{\mathcal{A}(q),\mathcal{B}(q)}_{h,g,R,S}(a,b)$ entails to computing $\frac{\partial}{\partial q}D^{\mathcal{A}(q)}_{h,R}(p)$ (and similarly for the set $\mathcal{B}$). Explicitly:

$$\frac{\partial}{\partial q}\Lambda(q)=\frac{\partial}{\partial q}D^{\mathcal{A}(q)}_{h,R}(b)\Big|_{b=b^*_{g,S}(q)}+\frac{\partial}{\partial q}D^{\mathcal{B}(q)}_{g,S}(a)\Big|_{a=a^*_{h,R}(q)}. \quad (21)$$

We will consider a very important special case of moving sets $\mathcal{A}(q)$ and $\mathcal{B}(q)$. Consider two fixed sets $\mathcal{U}$ and $\mathcal{V}$ that are inside $\mathbb{R}^3$. Let $E(\cdot,q):\mathbb{R}^3\mapsto\mathbb{R}^3$ and $F(\cdot,q):\mathbb{R}^3\mapsto\mathbb{R}^3$ denote *rigid transformations* (rotations and translations parametrized by q). Then we will consider the case in which $\mathcal{A}(q)=E(\mathcal{U},q)$ and $\mathcal{B}(q)=F(\mathcal{V},q)$, that is, the movement of the sets are due to a rigid transformation of the original sets. These can be, for example, two different links of the same robotic manipulator that move as the joint values, the system configuration q, change. This can also be a link of the manipulator and a *fixed* external object, in the particular case in which F is a fixed transformation.

Note that it was assumed at the beginning of this section that the sets $\mathcal{A}(q)$ and $\mathcal{B}(q)$ "vary smoothly" with q. This is related to the smoothness of the rigid transformations $E(\cdot,q)$ and $F(\cdot,q)$ on q. This holds true when we use as the configuration q joint angles (rotative joints) and/or joint positions (prismatic joints) of a serial kinematic chain. In this case, $E(\cdot,q)$ and $F(\cdot,q)$ are finite combination of sums, subtractions and products of smooth functions as sines and cosines.

Consider a fixed reference frame. In relation to this reference frame, for each one of the sets $\mathcal{U}$ and $\mathcal{V}$, we can compute the *linear velocity (v) Jacobian* and *angular velocity (ω) Jacobian*. We will write $J_v^{\mathcal{U}}(q)$, $J_\omega^{\mathcal{U}}(q)$ for the linear and angular velocity Jacobians, respectively. Similarly for the set $\mathcal{V}$, in which we will write $J_v^{\mathcal{V}}(q)$, $J_\omega^{\mathcal{V}}(q)$. We will also assume the position of the origin of the frame attached to these sets, in the reference frame, denoted by $p^{\mathcal{U}}(q)$ and $p^{\mathcal{V}}(q)$.

Before computing $\frac{\partial \Lambda}{\partial q}$ for the special case of rigid transformations, we need a preliminary result:

Proposition 9. *Let $T(\cdot, q) : \mathbb{R}^3 \mapsto \mathbb{R}^3$ be a rigid transformation that is differentiable on q such that $T(p, q) = Q(q)p + p_c(q)$ for a rotation matrix $Q \in \mathbb{R}^{3\times 3}$ and vector $p_c \in \mathbb{R}^3$. Let $T^{-1}(p, q) = Q(q)^\top (p - p_c(q))$ denote the inverse transformation. Let $J_v(q)$ and $J_\omega(q)$ be the respective linear and angular velocity Jacobians. Then:*

$$\frac{\partial T^{-1}}{\partial q}(p, q) = -\Big(J_v(q) + S\big(p_c(q) - p\big)J_\omega(q)\Big)^\top Q(q). \quad (22)$$

Proof. We use the derivative computation in one variable to infer the derivative in many variables. Suppose p is fixed and $q = q(t)$ for time t. Then, from the chain rule, $\frac{d}{dt}T^{-1}(p, q(t)) = \dot{q}(t)^\top \frac{\partial T^{-1}}{\partial q}(p, q(t))$.

Hence using the fact that $T^{-1}(p, q) = Q(q)^\top (p - p_c(q))$, that $\dot{p}_c = v = J_v(q)\dot{q}$, $\omega = J_\omega(q)\dot{q}$ and the properties of the matrix S in Result 5, we obtain that $\dot{q}(t)^\top \frac{\partial T^{-1}}{\partial q}(p, q(t))$ is equal to:

$$-\dot{q}^T \Big(J_v(q) + S\big(p_c(q) - p\big)J_\omega(q)\Big)^\top Q(q). \quad (23)$$

Since this is valid $\forall \dot{q}$, the desired result follows. □

We need the following useful fact:

Proposition 10. *Let $T(\cdot, q) : \mathbb{R}^3 \mapsto \mathbb{R}^3$ be a rigid transformation which is differentiable on q such that $T(p, q) = Q(q)p + p_c(q)$ for a rotation matrix Q and vector p_c. Let $T^{-1}(p, q) = Q(q)^\top (p - p_c(q))$ denote the inverse transformation. Let $J_v(q)$ and $J_\omega(q)$ be the respective linear and angular velocity Jacobians.*

Consider a fixed standard set $\mathcal{W} \subseteq \mathbb{R}^3$. Consider $h > 0$, then the following formulae hold:

$$D_{h,R}^{T(\mathcal{W},q)}(p) = D_{h,R}^{\mathcal{W}}\big(T^{-1}(p, q)\big). \quad (24)$$

$$\Pi_{h,R}^{T(\mathcal{W},q)}(p) = T\Big(\Pi_{h,R}^{\mathcal{W}}\big(T^{-1}(p, q)\big), q\Big). \quad (25)$$

$$\frac{\partial}{\partial q} D_{h,R}^{T(\mathcal{W},q)}(p) = \Big(J_v(q) + S\big(p_c(q) - p\big)J_\omega(q)\Big)^\top \Big(\Pi_{h,R}^{T(\mathcal{W},q)}(p) - p\Big). \quad (26)$$

Proof. **Proving** (24): First, we note that, since $T(\cdot, q)$ is a rigid transformation for all q, using the consistency to rigid transformations property for $W_R^{\mathcal{A}}$, we have that, $Vol_R(T(\mathcal{W}, q)) = Vol_R(\mathcal{W})$. Now, we use the expression for $D_{h,R}^{T(\mathcal{W},q)}(p)$ in (4) and make the change of variable $a = T(a', q)$ in the integral. The Jacobian determinant of the transformation will be 1 for all q, a' (since it is a rigid transformation), and thus $da = da'$. Furthermore, the integrated region will be transformed from $a \in T(\mathcal{W}, q)$ into $a' \in \mathcal{W}$. We then use the fact that, for a rigid transformation $T(\cdot, q)$, we have $\|p - T(a', q)\| = \|T^{-1}(p, q) - a'\|$ which together with the consistency to rigid transformations property for $W_R^{\mathcal{A}}$ allows us to conclude (24).

Proving (25): We use the fact that $\Pi_{h,R}^{\mathcal{A}}(p') = p' - \frac{\partial}{\partial p'} D_{h,R}^{\mathcal{A}}(p')$. Differentiating (24) on p, using the chain rule and the fact that $\frac{\partial}{\partial p} T^{-1}(p, q) = Q(q)$, we obtain our desired result.

Proving (26): We now differentiate on q in both sides of (24), use the chain rule and the fact that $\frac{\partial}{\partial p'} D_{h,R}^{\mathcal{A}}(p') = p' - \Pi_{h,R}^{\mathcal{A}}(p')$ to conclude that $\frac{\partial}{\partial q} D_{h,R}^{T(\mathcal{W},q)}(p)$ is

$$\frac{\partial T^{-1}}{\partial q}(p, q)\left(T^{-1}(p, q) - \Pi_{h,R}^{\mathcal{W}}\big(T^{-1}(p, q)\big)\right). \quad (27)$$

Then, we can use the expression $T^{-1}(p, q) = Q(q)^\top (p - p_c(q))$ to rewrite (27) as [1]:

$$\frac{\partial T^{-1}}{\partial q}(p, q)Q(q)\left(p - T\Big(\Pi_{h,R}^{\mathcal{W}}\big(T^{-1}(p, q)\big), q\Big)\right). \quad (28)$$

Now, using Proposition 9 and (25), this simplifies to:

$$\Big(J_v(q) + S\big(p_c(q) - p\big)J_\omega(q)\Big)^T \left(\Pi_h^{T(\mathcal{W},q)}(p) - p\right). \quad (29)$$

□

Finally, we use (21), together with the expressions for $\frac{\partial}{\partial q} D_{h,R}^{\mathcal{A}(q)}(b)$ and $\frac{\partial}{\partial q} D_{g,S}^{\mathcal{B}(q)}(a)$ obtained from (26) and the fact that $\Pi_{h,R}^{E(\mathcal{U},q)}(b_{g,S}^*) = a_{h,R}^*$ and $\Pi_{g,S}^{F(\mathcal{V},q)}(a_{h,R}^*) = b_{g,S}^*$ (the convergence criteria for the sequence in (16)), to obtain our final formula, after factoring out $(b_{g,S}^* - a_{h,R}^*)$:

$$\frac{\partial \Lambda}{\partial q} = \big[J_v^{\mathcal{U}} + S\big(p^{\mathcal{U}} - a_{h,R}^*\big)J_\omega^{\mathcal{U}} - J_v^{\mathcal{V}} - S\big(p^{\mathcal{V}} - b_{g,S}^*\big)J_\omega^{\mathcal{V}}\big]^\top (b_{g,S}^* - a_{h,R}^*) \quad (30)$$

in which we omitted the dependency on q. Note that it is very simple to compute the derivative once we have the velocity Jacobians (readily available in robotics from differential kinematics), the origins $p^{\mathcal{U}}$, $p^{\mathcal{V}}$ (readily available from forward kinematics) and $a_{h,R}^*$, $b_{g,S}^*$ (obtained from the sequence (16)). The formula in (30) is also notable in how little it depends on the approximation parameters h and g: just in the computation of $a_{h,R}^*$, $b_{g,S}^*$. So, the formula for different h, g are structurally similar, just changing the approximation for the closest points, $a_{h,R}^*$, $b_{g,S}^*$.

IV. Practical Implementation

A. Analytic computation

As mentioned in Examples 1 and 2, computing the volume integral for $D_{h,R}^{\mathcal{A}}(p)$ analytically is seldom possible, even for very simple sets. However, very good analytical approximations can be obtained for simple sets as spheres, boxes and cylinders in $\mathbb{R}^3$. We refer to our supplementary material [30].

[1] Using the following fact: if $T(u) = Qu + p_c$ is a rigid transformation and v, w are vectors, then $T^{-1}(v) - w = Q^T(v - p_c) - w = Q^T((v - p_c) - Qw) = Q^T(v - (Qw + p_c)) = Q^T(v - T(w))$.

There are two key aspects about these approximations. The first is that, as also mentioned in Example 1 and 2, special care must be taken due to the possibility of numerical underflow when computing numerically the integral in (4) for small values of h. Note that the integral inside the logarithm goes to 0 when $h \to 0$, and so we will have an indeterminate form $0 \cdot \log(0)$ for computing $D^{\mathcal{A}}_{h,R}$. In order to have this limit converge to the real value when $h \to 0+$, extreme numerical precision (often impossible) is needed *unless* a clever trick is used to rewrite the integral in a more amenable form. In the aforementioned supplementary material [30], these tricks are discussed.

The second keypoint is that when the time-varying set is a result of application of time-varying rigid transformation in a fixed set, which is often the case in robotics, Equation (25) implies that we need to calculate/approximate the projection for the object only in an static ("canonical") form, $\Pi^{\mathcal{W}}_{h,R}$. The projection for the translated/rotated body, $\Pi^{T(\mathcal{W},q)}_{h,R}$, can be obtained from $\Pi^{\mathcal{W}}_{h,R}$ using Equation (25).

B. Using cloud of points

For other convex shapes that are not discussed in the supplementary material [30], we need to think about suitable numerical schemes for computing $D^{\mathcal{A}}_{h,R}$ and $\Pi^{\mathcal{A}}_{h,R}$. Traditional numerical integration schemes may also be too slow, since in robotic applications, we need to compute the projections quickly to compute the points $a^*_{h,R}$ and $b^*_{g,S}$.

Inhere, *samples* of the set can be used. Let a_i be $m_{\mathcal{A}}$ points sampled from $\mathcal{A}$. We denote this sampled set by $\tilde{\mathcal{A}}$. Then, we can approximate $D^{\mathcal{A}}_{h,R}(p)$ (Definition 5) and $\Pi^{\mathcal{A}}_{h,R}(p)$ (Definition 6) by

$$\tilde{D}^{\tilde{\mathcal{A}}}_{h,R}(p) \triangleq -h^2 \log\left(\frac{1}{\tilde{V}^{\mathcal{A}}_R}\sum_{i=1}^{m_{\mathcal{A}}} \tilde{W}^{\tilde{\mathcal{A}}}_R(a_i) e^{-\frac{\|p-a_i\|^2}{2h^2}}\right);$$
$$\tilde{\Pi}^{\tilde{\mathcal{A}}}_{h,R}(p) \triangleq \frac{\sum_{i=1}^{m_{\mathcal{A}}} \tilde{W}^{\tilde{\mathcal{A}}}_R(a_i) e^{-\frac{\|p-a_i\|^2}{2h^2}} a_i}{\sum_{i=1}^{m_{\mathcal{A}}} \tilde{W}^{\tilde{\mathcal{A}}}_R(a_i) e^{-\frac{\|p-a_i\|^2}{2h^2}}} \tag{31}$$

in which $\tilde{W}^{\tilde{\mathcal{A}}}_R(a) \triangleq \exp(-\|a - \widetilde{Cen}(\tilde{\mathcal{A}})\|^2/(2R^2))$, $\widetilde{Cen}(\tilde{\mathcal{A}}) \triangleq \frac{1}{m_{\mathcal{A}}}\sum_{i=1}^{m_{\mathcal{A}}} a_i$ are approximations for $W^{\mathcal{A}}_R$ and $Cen(\mathcal{A})$ respectively and $\tilde{V}^{\mathcal{A}}_R \triangleq \sum_{i=1}^{m_{\mathcal{A}}} \tilde{W}^{\tilde{\mathcal{A}}}_R(a_i)$ is the discrete counterpart of $Vol_R(\mathcal{A})$. Note that the approximation for $\Pi^{\mathcal{A}}_{h,R}(p)$ is the derivative of the proposed approximation for $D^{\mathcal{A}}_{h,R}(p)$. Furthermore, note that we swapped the average, made by an integral in (4), by another (discrete) average in $\tilde{D}^{\tilde{\mathcal{A}}}_{h,R}(p)$. Finally, note that the computational complexity of computing $\tilde{D}^{\tilde{\mathcal{A}}}_{h,R}(p)$ and $\tilde{\Pi}^{\tilde{\mathcal{A}}}_{h,R}(p)$ is $\mathcal{O}(m_{\mathcal{A}})$, i.e., linear in the number of samples.

The expression for $\tilde{D}^{\tilde{\mathcal{A}}}_{h,R}(p)$ can be seen as the application of the function $LogSumExp_\lambda(u_1, u_2, .., u_m) = \frac{1}{\lambda}\log\left(\sum_{i=1}^m e^{\lambda u_i}\right)$ with $u_i = \|p - a_i\|^2/2 + h^2\|a_i - \widetilde{Cen}(\tilde{\mathcal{A}})\|^2/(2R^2)$ and $\lambda = -1/h^2$, plus a constant term $h^2\log(m_{\mathcal{A}})$. The *LogSumExp* function appears often in Machine Learning and Artificial Intelligence [31]. Interestingly, if we use $\tilde{D}^{\tilde{\mathcal{A}}}_{h,R}(p)$ instead of $D^{\mathcal{A}}_{h,R}(p)$, then all of the propositions in this paper hold (often with simpler proofs), except for two key Propositions: 3 and 6. The latter proposition is specially important since it guarantees the convergence of the sequence in (16). Thus, although convergence proofs for the algorithm in this discrete case were not provided, it is intuitive that as the point clouds become denser - meaning the discrete sum approaches the integral - we get closer to achieving the theoretical guarantees established in the continuous case.

C. Moving sets

We will also consider moving sets $\mathcal{A}(q)$, in which the movement is given by rigid transformations (as in Subsection III-E), see Figure 5. In this case, it advisable to sample the points of $\mathcal{A}(q)$ in the initial q_0 and perform rigid transformations in each one of them to obtain the sampled set at another q. More precisely, suppose $\mathcal{A}(q) = E(\mathcal{U}, q)$ for a fixed set $\mathcal{U}$ and a rigid transformation $E(\cdot, q)$, parametrized by q. Then $\tilde{\mathcal{A}}(q) = E(\tilde{\mathcal{U}}, q)$, that is, the sampled points of $\mathcal{A}(q) = E(\mathcal{U}, q)$ are obtained by sampling the fixed set $\mathcal{U}$, obtaining $\tilde{\mathcal{U}}$, and then applying the rigid transformation $E(\cdot, q)$ in every point of $\tilde{\mathcal{U}}$.

This comment is important, because we could, at each q, resample the set at the current configuration q in a different way. This can cause unnecessary noise in the computation of $\tilde{D}^{\tilde{\mathcal{A}}}_{h,R}(p)$. Furthermore, with this property, Equation (30) holds, but with $a^*_{h,R}$ and $b^*_{g,S}$ computed from $\tilde{\Pi}^{\tilde{\mathcal{A}}}_{h,R}$ and $\tilde{\Pi}^{\tilde{\mathcal{B}}}_{g,S}$ instead of $\Pi^{\mathcal{A}}_{h,R}$ and $\Pi^{\mathcal{B}}_{g,S}$.

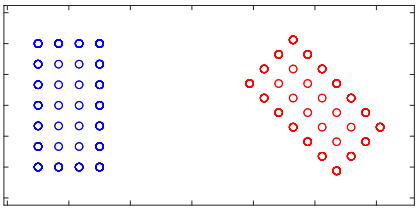


Fig. 5: An initial set, a rectangle, is sampled, obtaining the blue points. Then this set undergoes a translation and a rotation, and instead of resampling the points in a different way, we simply apply the same transformation to all the initial points, obtaining the points in red.

Finally, this property allows us to greatly simplify the computation of the (h, R)-witness points. Indeed, it turns out that it is not *necessary* to *explicitly* make the transformations for all the points in both sets at every time step. Let $E(\cdot, q)$ and $F(\cdot, q)$ be two rigid transformations parametrized by q. Let $\mathcal{A}(q) = E(\mathcal{U}, q)$ and $\mathcal{B}(q) = F(\mathcal{V}, q)$ for two static sets $\mathcal{U}$ and $\mathcal{V}$.

At first glance, to implement the distance computation we would need at every instant t to apply the transformations $E(\cdot, q(t))$ and $F(\cdot, q(t))$ for all sampled points of $\mathcal{U}$ and $\mathcal{V}$ respectively. If there are $m_{\mathcal{U}}$ and $m_{\mathcal{V}}$ points sampled from $\mathcal{U}$ and $\mathcal{V}$ respectively, this implies computing $m_{\mathcal{U}} + m_{\mathcal{V}}$ rigid transformations on points. Then, we use the (h, g, R, S) alternating algorithm in equation (16).

However, for the original set $\mathcal{U}$ (not the sampled set), Equation (25) shows this is not necessary. If $\tilde{\mathcal{A}}(q) = E(\tilde{\mathcal{U}}, q)$, then an analogous result can be proven for the sampled case:

$$\tilde{\Pi}_{h,R}^{E(\tilde{\mathcal{U}},q)}(p) = E\Big(\tilde{\Pi}_{h,R}^{\tilde{\mathcal{U}}}\big(E^{-1}(p,q)\big), q\Big) \tag{32}$$

and similarly for $\tilde{\mathcal{B}}(q) = E(\tilde{\mathcal{V}}, q)$. With this, the alternating algorithm becomes:

$$\begin{aligned} a[k] &= E\Big(\tilde{\Pi}_{h,R}^{\tilde{\mathcal{U}}}\big(E^{-1}(b[k],q)\big), q\Big); \\ b[k+1] &= F\Big(\tilde{\Pi}_{h,S}^{\tilde{\mathcal{V}}}\big(F^{-1}(a[k],q)\big), q\Big) \end{aligned} \tag{33}$$

in which $q = q(t)$. Hence we avoid, at every t, computing $E(\tilde{\mathcal{U}}, q(t))$ and $F(\tilde{\mathcal{V}}, q(t))$, which entails computing $m_{\mathcal{U}}+m_{\mathcal{V}}$ transformations on points. Instead, we apply four additional transformations on points (E, E^{-1}, F, F^{-1}) at every step of the iterative algorithm. Since the number of steps for convergence is usually very small compared with $m_{\mathcal{U}} + m_{\mathcal{V}}$, this trade pays off.

D. Numerical issues

As mentioned in Examples 1 and 2 and also in Subsection IV-A, there are numerical issues when computing $D_{h,R}^{\mathcal{A}}(p)$ and $\Pi_{h,R}^{\mathcal{A}}(p)$. These also appear when computing $\tilde{D}_{h,R}^{\mathcal{A}}(p)$ and $\tilde{\Pi}_{h,R}^{\mathcal{A}}(p)$, and the reason why is even clearer in the discrete case: when the terms $\|p-a_i\|^2/(2h^2)$ are all very large (i.e: p is very far from the points a_i and/or h is too small), the terms $e^{-\|p-a_i\|^2/(2h^2)}$ can be *very small* and the computation fails.

This issue is apparent in other applications in which $\mathit{LogSumExp}_\lambda(u_1, u_2, .., u_m)$ appears. To partially alleviate this, for each point p in which we want to compute $\tilde{D}_{h,R}^{\mathcal{A}}(p)$, we compute $i^*(p)$ as the index of one of the possible points that are closest to p. Then we factor out the term $e^{-\|p-a_{i^*(p)}\|^2/(2h^2)}$ in the logarithm and write the following *equivalent* expression for $\tilde{D}_{h,R}^{\mathcal{A}}(p)$ (in (31))

$$-h^2 \log\left(\frac{1}{\tilde{V}_R^{\mathcal{A}}}\sum_{i=1}^{m_{\mathcal{A}}} \tilde{W}_R^{\tilde{\mathcal{A}}}(a_i) e^{-\left(\frac{\|p-a_i\|^2-\|p-a_{i^*(p)}\|^2}{2h^2}\right)}\right) + \frac{\|p-a_{i^*(p)}\|^2}{2}. \tag{34}$$

There are two different sets of i: a) those in which $\|p - a_i\|^2 - \|p - a_{i^*(p)}\|^2 > 0$, and b) those in which $\|p - a_i\|^2 = \|p - a_{i^*(p)}\|^2 = 0$, the latter including at least $i = i^*(p)$. For the former set, we can still have that the exponential vanishes when p is too far from the a_i and/or h is small, *but*, for the latter set we have the value 1. Thus, the term inside the logarithm is *at least* $\tilde{W}_R^{\tilde{\mathcal{A}}}(a_{i^*(p)})/\tilde{V}_R^{\mathcal{A}}$ at all times, avoiding the numerical issue.

A similar trick can be applied for $\tilde{\Pi}_{h,R}^{\mathcal{A}}(p)$ (in (31)), by factoring out the term $e^{-\|p-a_{i^*(p)}\|^2/(2h^2)}$ in both the numerator and denominator:

$$\tilde{\Pi}_{h,R}^{\mathcal{A}}(p) = \frac{\sum_{i=1}^{m_{\mathcal{A}}} \tilde{W}_R^{\tilde{\mathcal{A}}}(a_i) e^{-\left(\frac{\|p-a_i\|^2-\|p-a_{i^*(p)}\|^2}{2h^2}\right)} a_i}{\sum_{i=1}^{m_{\mathcal{A}}} \tilde{W}_R^{\tilde{\mathcal{A}}}(a_i) e^{-\left(\frac{\|p-a_i\|^2-\|p-a_{i^*(p)}\|^2}{2h^2}\right)}}. \tag{35}$$

E. Initialization

The sequence in (16) (or in (33)) needs an initial point $b[0]$. The closer this initial point is to the final value of $b_{g,S}^*$, the better. When we have a set that moves smoothly, $b_{g,S}^*$ changes only a little from time step to time step. Thus, it is very beneficial to use the previous value of $b_{g,S}^*$ as the starting point in the next iteration.

F. Convergence of the iterative algorithm

Note that Proposition 6 allows us to bound the convergence rate of the algorithm. We start from the fixed point equation $b[k+1] = \Pi_{g,S}^{\mathcal{B}}(\Pi_{h,R}^{\mathcal{A}}(b[k]))$. From this, since the Jacobians of $\Pi_{g,S}^{\mathcal{B}}$ and $\Pi_{h,R}^{\mathcal{A}}$ are norm bounded by $1-1/(1+S^2/g^2) = S^2/(S^2+g^2)$ and $1-1/(1+R^2/h^2) = R^2/(R^2+h^2)$ respectively (see Proposition 6), the norm of the Jacobian of the composition of both is upper bounded by the product of both bounds: $c_{h,g,R,S} \triangleq \frac{S^2R^2}{(S^2+g^2)(R^2+h^2)} < 1$. Thus, we can bound the convergence of $b[k]$ towards the fixed point $b_{g,S}^*$ by the inequality:

$$\|b[k+1] - b_{g,S}^*\| \leq c_{h,g,R,S}\|b[k] - b_{g,S}^*\| \tag{36}$$

which implies linear convergence with a worst-case rate $c_{h,g,R,S}$. However, this convergence bound can be conservative, since the algorithm can actually converge much faster than that. To test this experimentally, we generated 36000 test cases with random objects (boxes, cylinders and spheres) placed on random orientations, with different values of h and R. In this experiment, for computing $\mathbb{D}_{h,g,R,S}^{\mathcal{A},\mathcal{B}}$, we assumed $h = g$ and $R = S$. We then, for each experiment, estimated the convergence factor of the run as $C = \lim_{k\to\infty} \|b[k] - b_{h,R}^*\|^{1/k}$. We then grouped these factors by the quantity $\Omega \triangleq R^2/(R^2+h^2)$, and computed the *maximum* value of C and the *mean* value of C for each group. The result is in Figure 6, in which we can see the following: as the ratio Ω increases, the worst-case convergence rate becomes worse, since C increases, but the average convergence rate stays mostly constant, with a value of about 0.17. Note that when $h = g$ and $R = s$ our worst case prediction for the linear rate $c_{h,g,R,S}$ is $c_{h,h,R,R} = \Omega^2$. In the experiments, the worst case C (maximum) was very close to this prediction, being approximately $0.9287\Omega^2$.

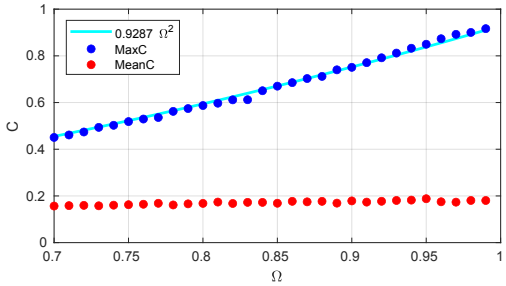


Fig. 6: Mean and maximum convergence factor C for different values of Ω, computed experimentally.

G. Computational time

The previous subsection discussed the *convergence rate* of the algorithm for computing the smooth distance between objects, which is related to the number of iterations the iterative algorithm $b[k+1] = \Pi^{\mathcal{B}}_{g,S}(\Pi^{\mathcal{A}}_{h,R}(b[k]))$ needs to converge (given a tolerance). This is a more computer-agnostic analysis of the complexity of the algorithm. However, the time taken for each iteration—and consequently the overall running time of the algorithm—is also important. This is, however, more computer-dependent since it depends on the hardware and implementation details of the algorithm.

Note that a single iteration of $b[k+1] = \Pi^{\mathcal{B}}_{g,S}(\Pi^{\mathcal{A}}_{h,R}(b[k]))$ requires the computation of two projections. The number of iterations primarily depends on the accepted tolerance—how close the current iteration is to the true value of the smooth projection. It is very rare, occurring in less than 3% of half a million test instances, for the number of iterations to exceed 5 when the tolerance is set to 0.001 meters. Therefore, we can confidently state that the computational time for the algorithm is generally less than $2 \times 5 \times t$, where t represents the average time needed to compute each projection. Thus, this subsection will focus on the computational cost of computing projections.

There are two distinct situations that need to be considered: when the object is represented using primitives (the original formulation proposed in Section III), and the extension using point clouds (presented in Subsection IV-B). The extension using point clouds, although more generic, is expected to be more computationally costly than the representation using primitives.

The following numbers were obtained using an 11th Gen Intel Core i7 with 2.30 GHz, in a C++ code. For primitive objects, the average computational time per smooth projection $\Pi^{\mathcal{A}}_{h,R}(p)$ is $0.3\mu s$, utilizing the approximations for the integral explained in [30]. The time taken for computing each projection does not significantly vary with the type of object (whether boxes, cylinders, or spheres) nor with the parameters h, R, provided that $h > 0$. However, it is worth noting that when $h = 0$ (corresponding to the traditional Euclidean distance), the projection can be computed within an average of $0.02\mu s$. Therefore, the additional time is a trade-off for achieving smoothness.

For the point cloud case, Figure 7 shows the average computational time for $\tilde{\Pi}^{\tilde{\mathcal{A}}}_{h,R}(p)$ considering different number of samples, $m_{\mathcal{A}}$. As expected, the time grows linearly with the number of points. From this plot, we can infer that the computational time seems to be roughly $9.0\ ns$ per sample.

This computation can be improved by parallelization: the terms $\exp(-\|p - a_j\|^2/(2h^2))$ can be computed in parallel for different values of j, significantly benefiting from the use of GPUs.

V. Application to Robotics

We will now present a second-order control framework that benefits from the properties of the present smooth distance. Theoretical guarantees and an experiment are presented as well.

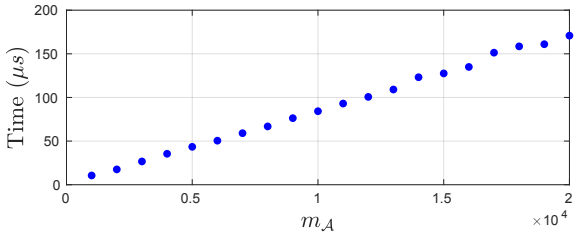


Fig. 7: Computational time for $\tilde{\Pi}^{\tilde{\mathcal{A}}}_{h,R}(p)$ for different number of samples ($m_{\mathcal{A}}$).

A. Second order control framework

Let our robotic system correspond to a double integrator, $\ddot{q} = u$. Let $r(q)$ be a twice differentiable task function such that $r(q) = 0$ if and only if the task is achieved (e.g., a task function to achieve a given pose). We consider obstacles that need to be avoided. These are either controlled by the configuration (e.g., links of the manipulator, in which we want to avoid inter-link collision) or static obstacles. We also consider joint limits, joint velocity limits and joint acceleration limits. Let $J_r(q) \triangleq \frac{\partial r}{\partial q}(q) = J_r$. We obtain our control input u by solving the following strictly convex quadratic optimization problem:

$$\begin{aligned} u = \arg\min_{a} \quad & \left\| J_r a + \dot{J}_r \dot{q} + 2\alpha\dot{r} + \alpha^2 r \right\|^2 + \beta \|a + \zeta\dot{q}\|^2 \\ \text{subject to:} \quad & Aa \leq b \end{aligned} \tag{37}$$

in which α, β, ζ are positive scalar constants and $A(q,\dot{q})$ and $b(q,\dot{q})$ are constraint matrices that represent the constraints (such as obstacle avoidance) that are considered in the motion. We also have that $\dot{r} = \frac{d}{dt} r(q) = J_r \dot{q}$ and $\dot{J}_r = \frac{d}{dt} J_r(q) \approx \frac{J_r(q+\dot{q}\epsilon) - J_r(q-\dot{q}\epsilon)}{2\epsilon}$ in which ϵ is a small number. Note that, since $\beta > 0$, the quadratic optimization problem is strictly convex (the matrix $J_r^\top J_r + \beta I_{n\times n}$ is always positive definite) and therefore there is always a single solution as long as the set $\{a \mid A(q,\dot{q})a \leq b(q,\dot{q})\}$ is always nonempty during the trajectory of $q(t)$.

The first term in the objective function in (37) exists to enforce that the desired task will be achieved. The second term exists to guarantee that there will be no spurious movements in the null space of the task Jacobian. If the robot has more degrees of freedom than the task demands, without this term the controller can induce permanent movements in the robot, that do not disturb the task, when the task is already achieved. This behavior is, of course, undesired. If there is no freedom left for the robot in the task, we can set $\beta = 0$ safely. Note that in this case the optimization problem is still strictly convex even with $\beta = 0$, because the task Jacobian J_r will be full column rank and thus $J_r^\top J_r$ will be a positive definite matrix.

Lyapunov stability can be guaranteed under certain conditions. The following proof borrows the same idea of the one of our previous work [32], but adapted for second order dynamics.

Proposition 11. *Let $\alpha, \beta, \zeta > 0$. Suppose that for all $q, \dot{q}$ there exists at least one $\hat{a}$ such that:*

$$\begin{aligned} &(i) \quad A\hat{a} \leq b; \\ &(ii) \quad J_r\hat{a} + \dot{J}_r\dot{q} + \alpha\dot{r} = 0; \\ &(iii) \quad \|J_r u + \dot{J}_r\dot{q} + \alpha\dot{r}\|^2 + \beta\|u\|^2 \geq \beta\left(2\zeta\dot{q}^\top\hat{a} + \|\hat{a}\|^2\right) \end{aligned} \tag{38}$$

in which $u = u(q, \dot{q})$ is computed according to (37). Then, the control loop $\ddot{q} = u(q, \dot{q})$ with u obtained by solving (37) is Lyapunov stable for the set $\{(q, \dot{q}) \mid r(q) = 0 \ , \ \dot{q} = 0\}$.

Proof. Consider the Lyapunov function:

$$V(q, \dot{q}) \triangleq \alpha\|\dot{r} + \alpha r\|^2 + \zeta\beta\|\dot{q}\|^2. \tag{39}$$

This Lyapunov function is nonnegative and zero only if $\dot{r} + \alpha r = 0$, and $\dot{q} = 0$, which, together, implies $r = 0$.

Since $\dot{V} = 2(\alpha\dot{r} + \alpha^2 r)^T(J_r u + \dot{J}_r\dot{q} + \alpha\dot{r}) + 2\beta\zeta\dot{q}^\top u$, we will show that $\dot{V} \leq 0$. For this, let $F(z) \triangleq \|J_r z + \dot{J}_r\dot{q} + 2\alpha\dot{r} + \alpha^2 r\|^2 + \beta\|z + \zeta\dot{q}\|^2$ describe the objective function of the optimization problem in (37). Let u be the optimal solution and $\hat{a}$ any vector that satisfies (38). Solving this minimization problem with $\hat{a}$ being in the feasible set (guaranteed due to (38)-(i)), we must have $F(u) \leq F(\hat{a})$. After expanding the two norms squared in the objective function, this inequality can be rewritten as:

$$\begin{aligned} &\|\alpha\dot{r} + \alpha^2 r\|^2 + \|J_r u + \dot{J}_r\dot{q} + \alpha\dot{r}\|^2 + \beta\|u\|^2 + 2\beta\zeta\dot{q}^T u \\ &+\beta\zeta^2\|\dot{q}\|^2 + 2(\alpha\dot{r} + \alpha^2 r)^T(J_r u + \dot{J}_r\dot{q} + \alpha\dot{r}) \leq \\ &\|\alpha\dot{r} + \alpha^2 r\|^2 + \|J_r\hat{a} + \dot{J}_r\dot{q} + \alpha\dot{r}\|^2 + \beta\|\hat{a}\|^2 + 2\beta\zeta\dot{q}^T\hat{a} \\ &+\beta\zeta^2\|\dot{q}\|^2 + 2(\alpha\dot{r} + \alpha^2 r)^T(J_r\hat{a} + \dot{J}_r\dot{q} + \alpha\dot{r}). \end{aligned} \tag{40}$$

Inequality (40) can be further simplified since $J_r\hat{a} + \dot{J}_r\dot{q} + \alpha\dot{r} = 0$ (due to (38)-(ii)). Furthermore, the term $\|\alpha\dot{r} + \alpha^2 r\|^2 + \beta\zeta^2\|\dot{q}\|^2$ can be canceled in both sides. Finally, from (38)-(iii) $\|J_r u + \dot{J}_r\dot{q} + \alpha\dot{r}\|^2 + \beta\|u\|^2 - \beta(2\zeta\dot{q}^T\hat{a} + \|\hat{a}\|^2) \geq 0$. Thus, we can rewrite (40) as: $2(\alpha\dot{r} + \alpha^2 r)^T(J_r u + \dot{J}_r\dot{q} + \alpha\dot{r}) + 2\beta\zeta\dot{q}^\top u \leq 0$ which corresponds to $\dot{V} \leq 0$. □

Note that Lyapunov stability (i.e., $\dot{V} \leq 0$) was established, but not *asymptotic Lyapunov stability* (i.e., $V \to 0$). This implies that the controller will prevent the system from becoming unstable and will continue to approach task completion, but it does not necessarily guarantee that the task will indeed be completed. There exists a possibility that the system reaches a *spurious equilibrium point*, that is, it may stop at a configuration q where the task remains unachieved ($r(q) \neq 0$).

Guaranteeing that $\hat{a}$ satisfies (38) seems to be a non-trivial assumption, while the condition (38)-(iii) requires particular attention. Note, however, that since $\|J_r u + \dot{J}_r\dot{q} + \alpha\dot{r}\|^2 \geq 0$, we can always force (38)-(iii) to hold if we make β a sufficiently small positive number. In fact, when $\beta \to 0+$ it holds trivially. As discussed before, we can set $\beta = 0$ when the task requires the same number of degrees of freedom that the robot has. When this is not the case (the task requires less), we can pad out the task with artificial sub tasks to ensure this.

Thus, the real challenge lies in enforcing (38)-(i) and (38)-(ii). The intuitive reasoning of this condition is the following: this set of constraints exists to guarantee that there is always a feasible ((38)-(i)) control action $u = \hat{a}$ that can make the robot decelerate and stop the task. Indeed, if $u = \hat{a}$, (38)-(ii) can be written as $\ddot{r} + \alpha\dot{r} = 0$, which implies that $\dot{r} \to 0$. The requirement that such action is always feasible mirrors our first order kinematic control results in [32], that has a requirement that $u = \dot{q} = 0_{n\times 1}$, is always feasible. This, of course, implies that $\dot{r} = 0$ as well. Thus, the conditions in (38) are the second-order version of the constraint that the robot always has the option to stop (either immediately, in the first order case, or in steady state, in the second order case).

B. Obstacle avoidance constraints

The work [13] proposes first order (single integrator system $\dot{q} = u$) constraints to ensure obstacle avoidance. These will be adapted here for the second order dynamics $\ddot{q} = u$.

Given a function $c(q)$, $c : \mathbb{R}^n \mapsto \mathbb{R}$, in which we want to guarantee that $c(q(t)) \geq 0, \ \forall t$, we can impose the following constraint:

$$\ddot{c} + 2\gamma\dot{c} + \gamma^2 c \geq 0 \tag{41}$$

for $\gamma > 0$. This is essentially a second order Control Barrier Function [33]. The Strong Comparison Lemma [34] allows us to compare the solution of this inequality with the solution of the corresponding equality $\ddot{c} + 2\gamma\dot{c} + \gamma^2 c = 0$. Therefore, we can conclude that this inequality implies:

$$c(q(t)) \geq \Big(c_0 + (\dot{c}_0 + \gamma c_0)t\Big)e^{-\gamma t} \ \forall t \tag{42}$$

Thus, if $c_0 \geq 0$ and $\dot{c}_0 + \gamma c_0 \geq 0$, then $c(q(t)) \geq 0$ for all $t \geq 0$.

Let $J_c = \frac{\partial c}{\partial q}$. Using the system dynamics $\ddot{q} = u$ in (41) and rewriting, we have the following constraint of the form $A(q, \dot{q})u \leq b(q, \dot{q})$:

$$-J_c u \leq \dot{J}_c\dot{q} + 2\gamma\dot{c} + \gamma^2 c \tag{43}$$

in which $\dot{c} = \frac{d}{dt}c(q) = J_c\dot{q}$ and $\dot{J}_c = \frac{d}{dt}J_c(q) \approx \frac{J_c(q+\dot{q}\epsilon) - J_c(q-\dot{q}\epsilon)}{2\epsilon}$ for a small scalar ϵ. To guarantee obstacle avoidance between objects i and j, we can use $c_{ij}(q) = \Lambda_{ij}(q) - \delta_{ij}$, in which $\Lambda_{ij}(q)$ is the (h, g, R, S)-half squared distance between the objects i and j (see Subsection III-E) and δ_{ij} is a positive margin. This positive margin is needed for $h, g > 0$ because the smoothed distance can be greater than zero when the two objects are colliding.

It is very important to note that, to implement this constraint, the function c must be *at least twice differentiable* on the variable q. Otherwise, not only $\dot{J}_c$ would be formally undefined, but if we try to approximate $\dot{J}_c$ using any numerical differentiation scheme, we may obtain a very large value. This can render the inequality (43) impossible to be met. Since, from the chain rule, $\dot{J}_c = \dot{q}^\top\frac{\partial^2 c}{\partial q^2}(q)$, there are two potential sources of this problem: $\dot{q}$ being too large and/or $\frac{\partial c}{\partial q}(q) = J_c(q)$ not being continuous on q (that would imply an "infinite" derivative $\frac{\partial}{\partial q}(\frac{\partial c}{\partial q}(q))$). The former issue can be eliminated since, as it will be shown in the next subsection, we can explicitly enforce $\dot{q}$ to be bounded. The second one can be guaranteed by ensuring that $J_c(q)$ is differentiable on q, which is a stronger condition than continuity. Equivalently, this means that c is twice differentiable on q. This justifies the

use of the smoothed distance, which is guaranteed to be twice differentiable, as opposed to the traditional distance, which is not always twice differentiable.

C. Joint position, speed and acceleration constraints

We will also consider constrained joint position $q_{\min} \leq q \leq q_{\max}$, joint velocity $\dot{q}_{\min} \leq \dot{q} \leq \dot{q}_{\max}$ and joint acceleration $\ddot{q}_{\min} \leq \ddot{q} \leq \ddot{q}_{\max}$. These limits are described as vectors containing limits for each joint.

The acceleration constraints are easy to implement:

$$u \geq \ddot{q}_{\min} \ , \ u \leq \ddot{q}_{\max}. \tag{44}$$

The velocity constraints are

$$u \geq -\xi(\dot{q} - \dot{q}_{min}) \ , \ u \leq -\xi(\dot{q} - \dot{q}_{max}) \tag{45}$$

for a positive scalar ξ. Since $\ddot{q} = u$, the aforementioned Strong Comparison Lemma implies that $\dot{q}(t) \geq (\dot{q}(0) - \dot{q}_{\min})e^{-\xi t} + \dot{q}_{\min}$ and $\dot{q}(t) \leq (\dot{q}(0) - \dot{q}_{\max})e^{-\xi t} + \dot{q}_{\max}$. Thus, if $\dot{q}_{\min} \leq \dot{q}(0) \leq \dot{q}_{\max}$, these bounds are valid $\forall t$.

The joint limits can be implemented using (41) with $c(q) = q - q_{\min}$ and $c(q) = q_{\max} - q$. We have, therefore:

$$u \geq -2\chi\dot{q} - \chi^2(q - q_{\min}) \ , \ u \leq -2\chi\dot{q} - \chi^2(q - q_{\max}) \tag{46}$$

in which γ in (41) was renamed to χ (γ will be used exclusively for the obstacle avoidance constraints). Combining (44), (45) and (46) we have:

$$\begin{aligned} u &\geq \max\Big(\ddot{q}_{\min}, -\xi(\dot{q} - \dot{q}_{\min}), -2\chi\dot{q} - \chi^2(q - q_{\min})\Big) \\ u &\leq \min\Big(\ddot{q}_{\max}, -\xi(\dot{q} - \dot{q}_{\max}), -2\chi\dot{q} - \chi^2(q - q_{\max})\Big) \end{aligned} \tag{47}$$

in which the $\min$ and $\max$ are taken componentwise.

D. Choosing the parameters

In order to implement the controller, we need to choose the parameters $\alpha, \beta, \zeta, \gamma, \xi$ and χ. We will now describe their impact on the system, therefore guiding the tuning process.

- The larger the α, the faster the convergence, but more acceleration is required from the robot;
- β is the weight of the regularization term $\|a + \zeta\dot{q}\|^2$ in the objective function (37): the larger it is, the smoother is the control input $\ddot{q}$, but the system is more prone to reach spurious stable equilibrium points (i.e., to fail to reach the target and to stop somewhere else). So we know that β should be relatively small when compared to the weight of the other term, which is 1. We suggest that it should be at least two orders of magnitude less.
- The clue on how to tune ζ comes from the Lyapunov candidate in eq. (39). There, we see that the product $\zeta\beta$ is the weight of another regularization term for the Lyapunov function, weighting the term $\|\dot{q}\|^2$. So $\zeta\beta$ should be relatively small when compared to the weight of the other term, α. At least two orders of magnitude less, from our experience;
- The larger are γ, ξ, χ, the more the robot is willing to approach obstacles and violate velocity/joint limits, respectively, to complete the task. This is specially important in tasks as the one we propose in Section VI (going through a hole), because the robot has to be close to collision to the edges of the hole in order to complete the task. As we increase these parameters too much, we increase the risk of having collisions and violating velocity/joint limits, i.e., the safety is reduced. This is important because our dynamical model $\ddot{q} = u$ is not perfect, and non-modeled dynamics together with reduced safety margins can lead the system to a collision.

VI. Experimental Studies

In the experimental setup, we consider a 7 DoF robot Kuka LBR iiWA R820[2] that must achieve a pose that is behind a styrofoam-wall with a hole. Thus, the manipulator should go through the wall using the hole and achieve the desired pose without colliding with the wall. Figure 8 shows the 3D starting (left) and final (right) configuration of the arm.

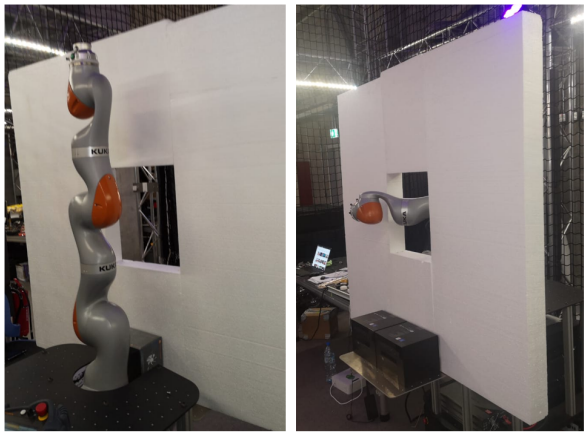


Fig. 8: Snapshots of the experiment.

The task can be represented by the desired end-effector position vector p_{ef}^d and desired orientation, represented as the desired orthonormal axis vectors $x_{ef}^d, y_{ef}^d, z_{ef}^d$, all expressed in the *world frame*. Let $p_{ef}(q), x_{ef}(q), y_{ef}(q), z_{ef}(q)$ be the respective vectors (current position and current orthonormal axis vectors) for the robot at the current configuration q, also expressed in the world frame, which can be obtained with standard forward kinematics. We use as the task function $r(q)$ the 6-dimensional vector

$$r(q) = \begin{bmatrix} p_{ef}(q) - p_{ef}^d \\ 1 - (x_{ef}^d)^\top x_{ef}(q) \\ 1 - (y_{ef}^d)^\top y_{ef}(q) \\ 1 - (z_{ef}^d)^\top z_{ef}(q) \end{bmatrix}. \tag{48}$$

It can be shown that $r(q) = 0$ if and only if $p_{ef}(q) = p_{ef}^d$, $x_{ef}(q) = x_{ef}^d$, $y_{ef}(q) = y_{ef}^{des}$ and $z_{ef}(q) = z_{ef}^d$. For the last three components, we use the fact that if a, b are two normalized vectors, $1 - a^T b = 0$ iff $a = b$. This task function is also related to the task in the "decoupled controller" in [35], but with a homogeneous transformation matrix approach instead of a dual quaternion approach.

[2]https://www.kuka.com/en-de/products/robot-systems/industrial-robots/lbr-iiwa.

The robot collision model is composed of 11 primitive objects: 4 spheres (S_i), 1 box (B_i) and 6 cylinders (C_i), as shown in Figure 9. The obstacle is a 10 cm thick styrofoam wall with a 50 cm square hole in its middle, composed of 4 boxes. Thus, we will have $11 \times 4 = 44$ distances to be computed, $\Lambda_{ij}(q)$, since the robot's primitives $i \in 1, \ldots, 11$ and the obstacle's primitives $j \in 1, \ldots, 4$. This implies 44 constraints in the form of (43). Furthermore, we have $2 \times 7 = 14$ maximum and minimum joint limit constraints as in (45) resulting in $44 + 14 = 58$ constraints for the quadratic program.

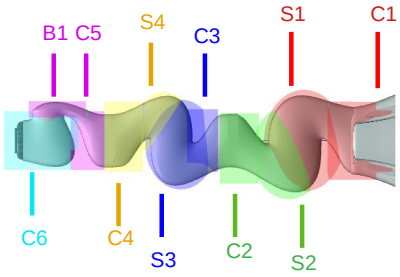


Fig. 9: Collision model of the robot, showing the Spheres (S), Cylinders (C) and Boxes (B).

In the control framework that we used, the joint velocity is computed numerically (in combination with a low pass filter) from the joint position measurement. We then compute the target acceleration, integrate it twice numerically and, finally, send to the robot the next joint position. In a 11th Gen Intel Core I7 with 2.90 Ghz, the average computational time for the whole control loop was around 5ms. Everything was implemented in `C++` using the `iiwa_stack` package [36] for ROS communication. The controller runs at 100 Hz.

The parameters used are:

- Equation (37): $\alpha = 2\ s^{-1}$, $\beta = 0.005$, $\zeta = 0.5\ s^{-1}$;
- Equation (43): $\gamma = 0.5\ s^{-1}$, $h = g = 0.05\ m$, $R = S = 10\ m$ (for all 44 pairs of objects) and $\delta_{ij} = 0.2\ m^2$ (for all 44 pairs of objects);
- Equation (47): $\chi = 0.5 s^{-1}$, $q_{max} = -q_{min} = [165^o\ 115^o\ 165^o\ 115^o\ 163^o\ 115^o\ 170^o]^\top$ (in degrees), No explicit limits for joint velocity and joint acceleration were implemented.

Figure 10 shows , for this experiment, the joint angles q_i and joint angular velocities $\dot{q}_i$, $i = 1, \ldots, 7$. These are expressed in degrees and degrees/s, respectively and the motion spans a wide range of motion for the angles. Note that the joint limits were respected and, although no explicit joint speed limits were enforced, the values fall within the robot's capabilities (75 degrees/second).

Figure 11 shows the control inputs u_i, $i = 1, \ldots, 7$, which are the accelerations used in each joint expressed in degrees/second^2. For some joints, there is a fast spike of acceleration (around ± 40 degrees/second^2) at the beginning, not displayed in order to not distort the view for the remainder of the graph. This is expected because we are applying a "step error input" (i.e., from zero error to the maximum error

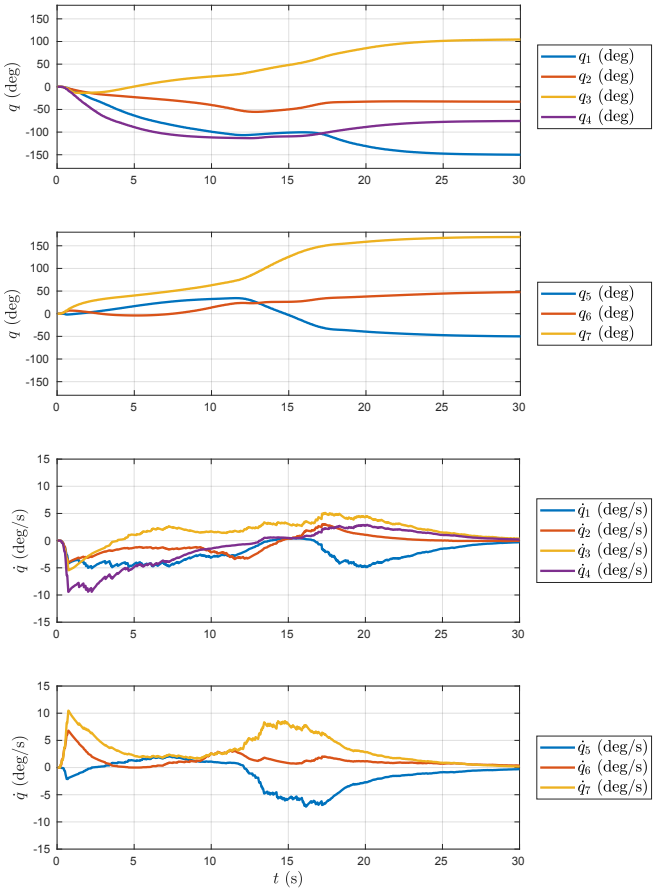


Fig. 10: Experimental joint angles and angular joint velocities

without any smooth transition between the two states) for the system, so from the very beginning there is a relatively large error $r(q_0)$ that causes a spike of the control input. Overall, there is significant motion during the first second while the robot adjusts its pose so that it can enter the square cavity followed by similar action around 15 seconds where after the robot's end effector has passed through the cavity it then adjusts its body to move around the cavity.

The time evolution of the *Lyapunov function* $V(q, \dot{q})$ defined in (39) is shown in Figure 12. The Lyapunov function decreases steadily towards zero and, after 30 seconds, the error is 0.2 mm in position and 4° in orientation.

The *smallest distance* (i.e., smooth distance with $h = g = 0$) between the robot's primitive objects and the styrofoam-wall (obstacle) is shown in Figure 13. The minimum (positive) observed value was 13 mm at $t \simeq 20\ s$ implying that no collision occurred.

Figure 14 shows a snapshot of the 3D-visualization of the experiment in which we can see some of the (h, R)-witness points between the robot's primitives and the obstacles. A pair of witness points $(a^*_{h,R}, b^*_{g,S})$ computed from the (h, g, R, S) projection algorithm in (16) (one on the obstacle and other on the robot) shares the same color, and the color also matches the color of the respective primitive. These points can also be seen in the interactive replay [37] of the experiment.

For comparison purposes, we also implemented the algo-

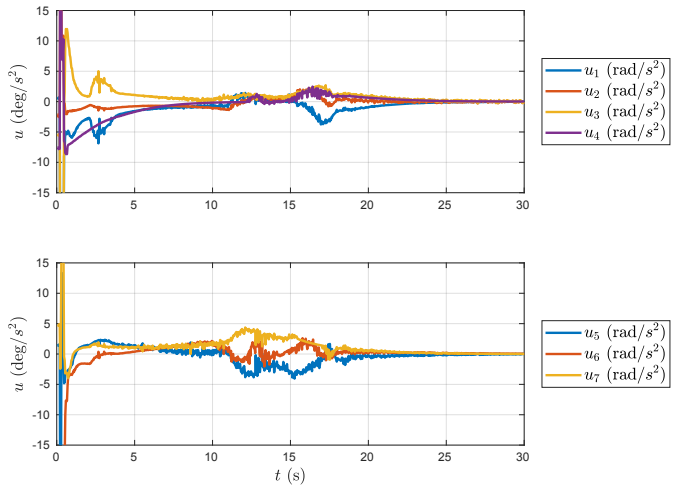


Fig. 11: Experimental control inputs $u_i,\ i = 1, \ldots, 7$.

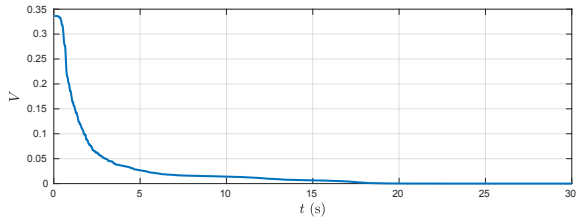


Fig. 12: Experimental evolution of the Lyapunov function $V(q, \dot{q})$.

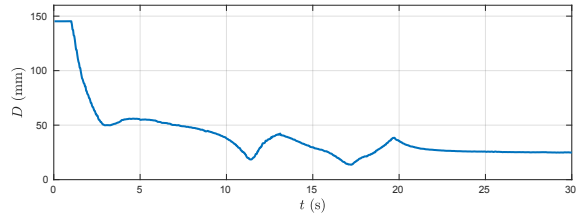


Fig. 13: Experimental evolution of obstacle avoidance minimum distance.

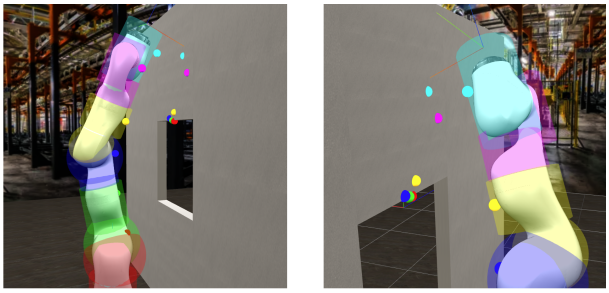


Fig. 14: Witness points between the robot's primitives and the obstacles.

rithm with the same utilized parameters having $h = g = 0$ (i.e., using the traditional Euclidean distance) and $\delta_{ij} = 0.02$ m^2. The QP became unfeasible, due to the term $\dot{J}_c\dot{q}$ in (43) being a very large negative number for several pairs of obstacles and primitives, thus rendering the feasible space of the QP empty. This was anticipated since at the initial configuration, several robot primitive-cylinders are parallel to the obstacle (see Figure 8-left and also Figure 15), a situation in which the pair of witness points for the traditional Euclidean distance are non-unique. Thus, a small movement changes the witness points drastically, resulting in large $\dot{J}_c\dot{q}$. The same situation happens when using other approaches that provide non-differentiable distance functions in this case. For example, using the scaling factor in [10], the distance function between the robot primitive-cylinders and the wall at the starting configuration is non differentiable. Furthermore, the same would happen if we used the pills proposed in [5] instead of cylinders. However, this situation is avoided when using smooth distances.

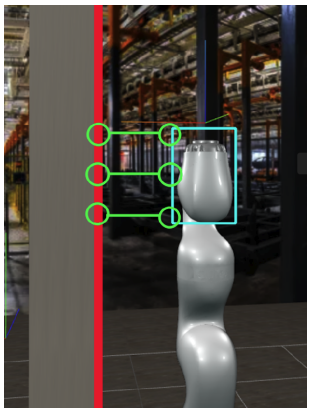


Fig. 15: 3D reproduction of the initial configuration of the manipulator. At this starting configuration, the shortest distance between the robot and the wall are achieved with more than one point, with some of them represented in green. In this case, the traditional Euclidean distance is non-differentiable.

As mentioned in the introduction, [11] proposes an alternative method, the bundled objective/gradient, to smooth any function/gradient. In particular, it can be used to smooth the distance function. In this case, the true Euclidean robot-to-environment distance function $\Lambda_0(q)$, which is defined in (19) with $h = g = 0$, is calculated at N configurations q_i and then averaged (with weights) to obtain a single smoothed instance of the distance function. This contrasts with our approach, which involves a single computation of $\Lambda(q)$ in (19) with $h, g \neq 0$. The noteworthy aspect is that a single computation of the true distance $\Lambda_0(q)$ is less expensive than a single computation of $\Lambda(q)$ with $h, g \neq 0$. However, the averaging process requires many computations of $\Lambda_0(q)$. Consequently, there is a threshold number of samples N beyond which, from a computational time perspective, the method no longer offers a benefit. This threshold depends on several factors, but most notably on the number of primitives composing the robot. For instance, in the case of the Kuka robot used in this experiment, which consists of 11 primitives, a single computation of $\Lambda_0(q)$

with the respect to an obstacle composed of one primitive (e.g., one of the boxes composing the wall-with-a-hole environment) takes an average of 6 μs on an 11th Gen Intel Core i7 with 2.90 GHz, whereas $\Lambda(q)$ takes about 24 μs. Since $24/6 = 4$, this means that if the number of samples N exceeds 4 the bundled objective/gradient approach ceases to be more efficient. Only four samples is a very low threshold: to illustrate, [11] presents results with $N = 100$.

An interactive replay of this experiment, using real data collected from the robot, can be seen in [37]. It also shows the robot's primitives in movement, along with the (h, R)-witness points at the robot and at the obstacles for the last three links; a video highlighting the robot's trajectory can be seen here `https://youtu.be/j4QkzO3n978` and snapshots of the experiment can be seen in Figure 16.

VII. Conclusions

In this paper, we proposed a modified point-to-set distance (Definition 5) that is guaranteed to be smooth (i.e., infinitely differentiable). We then proved that a modified form of Von Neumann's projection algorithm for convex sets, obtained simply by swapping the real distance by the modified distance, can be used to compute the counterparts of the witness points in this modified distance.

Using the modified witness points, we defined a set-to-set distance (Definition 8). We then used this set-to-set distance to define a distance function in the configuration space, between two convex shapes when one of the shapes is attached to a kinematic chain of a robot manipulator (Equation (19)). Then, thanks to the definition of the set-to-set distance that we propose, we show that the Jacobian of this distance is structurally the same as the one we have when we use the real distance instead of the modified one, in the sense that we just have to change the witness points used in the original formula by their modified counterparts.

We then proposed an acceleration-based control framework that leverages the properties of the proposed modified distance. This framework allows us to have collision avoidance with obstacles. This is done by implementing as a constraint a differential inequality, for which we use the Jacobian of the distance in the configuration space and also for which it is necessary to have at least the second derivative of the distance function. This controller was implemented into a 7-DoF Kuka LBR iiwa R820 robot and we highlighted the benefits of the proposed smooth distance in this setting.

Acknowledgements

The authors wish to thank NYUAD's Core Technologies Lab Kinesis for sharing its facilities for the experiment presented in Section VI. This work was partially supported by the NYUAD Center for Artificial Intelligence and Robotics (CAIR), funded by Tamkeen under the NYUAD Research Institute Award CG010.

References

[1] J.-C. Latombe, *Robot Motion Planning*. Springer, 1990.

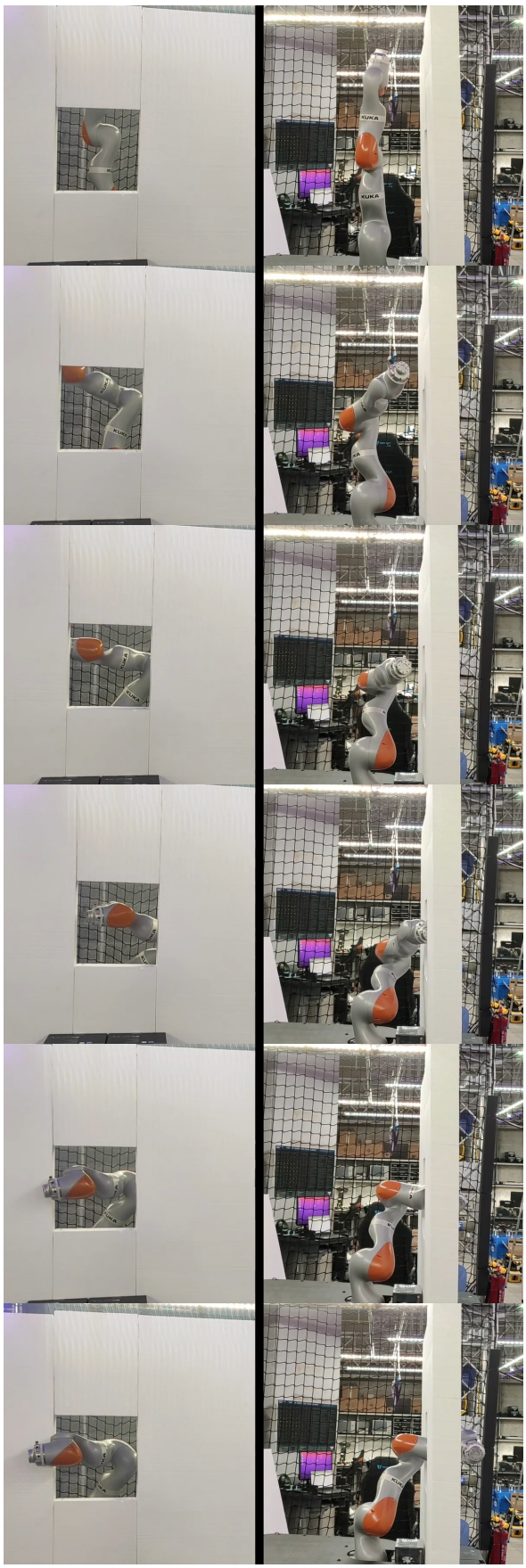


Fig. 16: Experimental snapshots (front and left side) at $t \in \{0, 5, 10, 15, 20, 25\}$ sec.

[2] A. Escande, S. Miossec, and A. Kheddar, "Continuous gradient proximity distance for humanoids free-collision optimized-postures," in *2007 7th IEEE-RAS International Conference on Humanoid Robots*, 2007, pp. 188–195.

[3] A. Dietrich, T. Wimböck, A. Albu-Schäeffer, and G. Hirzinger, "Integration of reactive, torque-based self-collision avoidance into a task hierarchy," *IEEE Transactions on Robotics*, vol. 22, pp. 1278–1293, 11 2012.

[4] A. Escande, S. Miossec, M. Benallegue, and A. Kheddar, "A strictly convex hull for computing proximity distances with continuous gradients," *IEEE Transactions on Robotics*, vol. 30, no. 3, pp. 666–678, 2014.

[5] K. Tracy, T. A. Howell, and Z. Manchester, "Diffpills: Differentiable collision detection for capsules and padded polygons," https://arxiv.org/abs/2207.00202, 2022.

[6] X. Zhu, H. Ding, and S. Tso, "A pseudodistance function and its applications," *IEEE Transactions on Robotics and Automation*, vol. 20, no. 2, pp. 344–352, 2004.

[7] J. Xu, Z. Liu, C. Yang, L. Li, and Y. Pei, "A pseudo-distance algorithm for collision detection of manipulators using convex-plane-polygons-based representation," *Robotics and Computer-Integrated Manufacturing*, vol. 66, p. 101993, 12 2020.

[8] A. Schmeißer, R. Wegener, D. Hietel, and H. Hagen, "Smooth convolution-based distance functions," *Graphical Models*, vol. 82, pp. 67–76, 2015.

[9] M. Sanchez, O. Fryazinov, P. Fayolle, and A. Pasko, "Convolution filtering of continuous signed distance fields for polygonal meshes," *Computer Graphics Forum*, vol. 34, p. 277–288, 09 2015.

[10] K. Tracy, T. A. Howell, and Z. Manchester, "Differentiable collision detection for a set of convex primitives," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023.

[11] H. J. T. Suh, T. Pang, and R. Tedrake, "Bundled gradients through contact via randomized smoothing," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4000–4007, 2022.

[12] Y. Zheng and K. Hang, "Calculating the support function of complex continuous surfaces with applications to minimum distance computation and optimal grasp planning," *IEEE Transactions on Robotics*, vol. 36, no. 4, pp. 1004–1021, 2020.

[13] O. Kanoun, F. Lamiraux, and P.-B. Wieber, "Kinematic control of redundant manipulators: Generalizing the task-priority framework to inequality task," *IEEE Transactions on Robotics*, vol. 27, pp. 785 – 792, 09 2011.

[14] E. Gilbert and D. Johnson, "Distance functions and their application to robot path planning in the presence of obstacles," *IEEE Journal on Robotics and Automation*, vol. 1, no. 1, pp. 21–30, 1985.

[15] W. Shaw Cortez and D. V. Dimarogonas, "Safe-by-design control for Euler–Lagrange systems," *Automatica*, vol. 146, p. 110620, 2022.

[16] M. Rauscher, M. Kimmel, and S. Hirche, "Constrained robot control using control barrier functions," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 279–285.

[17] M. A. Murtaza, S. Aguilera, V. Azimi, and S. Hutchinson, "Real-time safety and control of robotic manipulators with torque saturation in operational space," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021, pp. 702–708.

[18] J. Glaisher, "XXXII. On a class of definite integrals," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 42, no. 280, pp. 294–302, 1871.

[19] G. Strang and B. Coonley, *Linear Algebra and Its Applications*. Harcourt Brace Jovanovich, 1998. [Online]. Available: https://books.google.ae/books?id=GtF1PwAACAAJ

[20] A. Saumard and J. A. Wellner, "Log-concavity and strong log-concavity: a review," *Statistics surveys*, vol. 8, p. 45, 2014.

[21] A. Dembo and O. Zeitouni, *Large Deviations Techniques and Applications*, ser. Stochastic Modelling and Applied Probability. Springer Berlin Heidelberg, 2009.

[22] K. Ciesielski, "On Stefan Banach and some of his results," *Banach Journal of Mathematical Analysis*, vol. 1, no. 1, pp. 1 – 10, 2007.

[23] W. Cheney and A. A. Goldstein, "Proximity maps for convex sets," *Proceedings of the American Mathematical Society*, vol. 10, no. 3, pp. 448–450, 1959.

[24] H. Bauschke and J. J. Borwein, "On the convergence of von neumann's alternating projection algorithm for two sets," *Set-Valued Analysis*, vol. 1, pp. 185–212, 06 1993.

[25] J. J. Borwein and A. Lewis, *Convex Analysis and Nonlinear Optimization : Theory and Examples*. Springer, 2000.

[26] M. L. Eaton, *Multivariate statistics: A vector space approach*. Wiley, 2007.

[27] S. Boyd, S. P. Boyd, and L. Vandenberghe, *Convex optimization*. Cambridge University Press, 2004.

[28] T. V. Petersdorff, "Notes on contractible maps," https://terpconnect.umd.edu/~petersd/666/fixedpoint.pdf.

[29] D. G. Anderson, "Iterative procedures for nonlinear integral equations," *J. ACM*, vol. 12, no. 4, p. 547–560, oct 1965. [Online]. Available: https://doi.org/10.1145/321296.321305

[30] V. M. Gonçalves, A. Tzes, F. Khorrami, and P. Fraisse, "Supplementary material for "Smooth Distances for Second Order Kinematic Robot Control"," https://raw.githubusercontent.com/viniciusmgn/uaibot_content/master/contents/PDFs/TRO_Smooth_Supplementary.pdf, 2023.

[31] B. Gao and L. Pavel, "On the properties of the softmax function with application in game theory and reinforcement learning," *ArXiv*, vol. abs/1704.00805, 2017.

[32] V. M. Gonçalves, P. Fraisse, A. Crosnier, and B. V. Adorno, "Parsimonious kinematic control of highly redundant robots," *IEEE Robotics and Automation Letters*, vol. 1, no. 1, pp. 65–72, 2016.

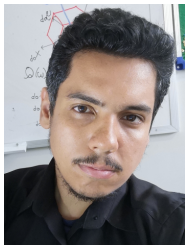
[33] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, "Control barrier functions: Theory and applications," in *2019 18th European Control Conference*, 2019, pp. 3420–3431.

[34] A. McNabb, "Comparison theorems for differential equations," *Journal of Mathematical Analysis and Applications*, vol. 119, no. 1, pp. 417 – 428, 1986.

[35] L. F. da Cruz Figueredo, B. V. Adorno, and J. Y. Ishihara, "Robust h kinematic control of manipulator robots using dual quaternion algebra," *Automatica*, vol. 132, p. 109817, 2021.

[36] C. Hennersperger, B. Fuerst, S. Virga, O. Zettinig, B. Frisch, T. Neff, and N. Navab, "Towards MRI-based autonomous robotic us acquisitions: a first feasibility study," *IEEE Transactions on Medical Imaging*, vol. 36, no. 2, pp. 538–548, 2017.

[37] V. M. Gonçalves, A. Tzes, F. Khorrami, and P. Fraisse, "Interactive replay for "Smooth Distances for Second Order Kinematic Robot Control"," https://raw.githack.com/viniciusmgn/uaibot_content/master/contents/Experiments/realexperiment_kuka_smooth.html, 2023.



Vinicius M. Gonçalves received his B.S. degree in control and automation engineering and his Ph.D. degree in control in dioid algebras and discrete event systems from the Universidade Federal de Minas Gerais (UFMG), Belo Horizonte, Brazil, in 2011 and 2014, respectively. From 2017 to 2022, he was an Assistant Professor with the Department of Electrical Engineering at UFMG. He is currently a research associate at the Center for Artificial Intelligence and Robotics (CAIR) at New York University Abu Dhabi. His current research interests include control, robotics, optimization, and discrete-event systems.

Anthony Tzes is a graduate of Electrical Computer Engineering (ECE) Department (1985) at University of Patras (UPAT), Greece. He received his doctorate from the Ohio State University in 1990. From 1990 until 1999 he was a tenured associate professor with Polytechnic University, now New York University (NYU) Tandon School of Engineering. He has been a Professor and ECE Department Head at UPAT since 1999. He has been the Chairman of IEEE's Control Systems Society Greek Chapter, a member of the national (Greek) committee of the European Control Association (EUCA), and member of several committees. Professor Tzes joined NYU Abu Dhabi (NYUAD) in 2017, as a tenured Professor at its Engineering Division. He is the Program Head of the EE and the Lead Investigator of NYUAD's Center for AI and Robotics. He has served in various positions, and as IPC-member at several international conferences. Prof. Tzes has received research funding from various organizations. He has authored more than 100 papers published in international journals. He has received various awards as co-author of published articles. He has also served in the editorial board of several journals. His research interests include distributed collaborative control of autonomous mobile agents, cooperative control of networked systems, and other control engineering applications.

Farshad Khorrami received Bachelors degrees in mathematics (1982) and electrical engineering (1984) from The Ohio State University (OSU). He also received his Master's degree in mathematics (1984) and Ph.D. in electrical engineering (1988) from OSU. He is currently a professor of ECE at NYU where he joined in 1988. His research interests include adaptive and nonlinear controls, robotics and unmanned vehicles, cyber security for cyber-physical systems, embedded systems security, machine learning, and large-scale systems. He has published about 350 refereed journal and conference papers and has 15 U.S. patents. His book "Modeling and Adaptive Nonlinear Control of Electric Motors" was published by Springer Verlag in 2003. He has developed the Control/Robotics Research Laboratory and is a co-director of the Center in AI and Robotics. His research has been supported by the ARO, NSF, ONR, DARPA, ARL, AFRL, NASA, and several corporations. He has served as conference organizing committee member of several international conferences.

Philippe Fraisse received the master's degree in electrical engineering from the Ecole Normale Supérieure de Cachan, Cachan, France, in 1988, and the Ph.D. degree in automatic control from the University of Montpellier, Montpellier, France, in 1994. He is currently a Professor with the Robotics Department, University of Montpellier. He has authored or co-authored over 150 papers in international peer-reviewed conferences and journals. He was Associate-Editor with IEEE/RAS Transactions on Robotics (2016-2019), Senior Editor with IEEE/RSJ IROS (2018,2019). His research interests include physical human–robot interaction, humanoid robotics, robotics for rehabilitation, and mobile manipulators.