Regularized Deep Signed Distance Fields for Reactive Motion Generation

Puze Liu, Kuo Zhang, Davide Tateo, Snehal Jauhri, Jan Peters and Georgia Chalvatzaki

***Abstract*— Autonomous robots should operate in real-world dynamic environments and collaborate with humans in tight spaces. A key component for allowing robots to leave structured lab and manufacturing settings is their ability to evaluate online and real-time collisions with the world around them. Distance-based constraints are fundamental for enabling robots to plan their actions and act safely, protecting both humans and their hardware. However, different applications require different distance resolutions, leading to various heuristic approaches for measuring distance fields w.r.t. obstacles, which are computationally expensive and hinder their application in dynamic obstacle avoidance use-cases. We propose Regularized Deep Signed Distance Fields (ReDSDF), a single neural implicit function that can compute smooth distance fields at any scale, with fine-grained resolution over high-dimensional manifolds and articulated bodies like humans, thanks to our effective data generation and a simple *inductive bias* during training. We demonstrate the effectiveness of our approach in representative simulated tasks for whole-body control (WBC) and safe Human-Robot Interaction (HRI) in shared workspaces. Finally, we provide proof of concept of a real-world application in a HRI handover task with a mobile manipulator robot.**

I. Introduction

Safety is an essential prerequisite for the real-world deployment of robots and their interaction with the dynamic world and the humans in it. The study of safety constraints in robotics that can filter dangerous actions is as long as the field itself. A common way of approaching the problem is to define constraint functions that apply to specific applications and tasks. The imposed constraints can ensure safety by excluding unsafe parts of the state and action space to avoid collisions and self-collision, harming humans in shared workspaces, or for specifying task-specific workspaces, e.g., in human-robot collaboration, robot-assisted feeding, dressing, etc. Each of these tasks may demand a different set of constraints to be specified and satisfied during deployment.

Despite their importance, constraint functions may be difficult to hand-design, while ill-specified constraints may often make it challenging for robots to plan [1], optimize [2], learn [3], or act reactively [4]. One way of specifying a constraint is to define its boundaries as a level curve k of an implicit function $f(X) = k$, where X can be any query point in space. This formulation is particularly convenient because it defines the constraint over a surface intuitively. Second, it is easy to reason about constraint violations/slack, allowing for easier optimization. Furthermore, if the constraint is also differentiable, it is also possible to use gradient-based projection algorithms to enforce them.

(a) Robot view (b) ReDSDF of the human

Fig. 1: Real-world evaluation of our learned ReDSDF model in a HRI scenario. The mobile manipulator robot TIAGo++ has to approach the human and deliver him a water bottle without intervening in the safety boundary. (a) The perceived instance in the robot's field of view. (b) The perceived human manifold, along with the 20cm level set (red point-cloud), i.e. the set of points at 20cm distance to the human.

It is common in robotics to formulate constraints in terms of distance w.r.t. a given object. Measuring distances w.r.t. to manifolds is necessary for most robot applications for collision avoidance [5], but also for HRI, where the robot should position itself at a certain distance from the surrounding humans to interact with them safely while respecting social norms [6]–[8]. Signed Distance Function (SDF) is a prominent representation for expressing distance w.r.t. a given surface, by defining a function that precomputes the distance of an arbitrary point in the space w.r.t. a fixed surface. Every point on the surface has zero distance, while points outside it have a signed distance. The sign is chosen

Computer Science Department, Technische Universität Darmstadt, Germany puze@robot-learning.de, kuo.zhang@stud.tu-darmstadt.de, {davide.tateo, snehal.jauhri, jan.peters, georgia.chalvatzaki}@tu-darmstadt.de

This research received funding by the DFG Emmy Noether Programme (#448644653), the RoboTrust project of the Centre Responsible Digitality Hessen, Germany and the China Scholarship Council (No. 201908080039). Research presented in this paper has been supported by the German Federal Ministry of Education and Research (BMBF) within a subproject "Modeling and exploration of the operational area, design of the AI assistance as well as legal aspects of the use of technology" of the collaborative KIARA project (grant no. 13N16274).

by convention to be positive in the line of sight w.r.t. the sensor detecting the surface. If a surface represents a closed region of space, e.g., an object or a human, we can consider every point inside this region as a negative distance. Different variants of SDF were introduced in particular for robotics to counteract the high computational complexity and the non-smooth representations that arise from querying multiple points from numerous static or even dynamic obstacles [9].

In this paper, we present ReDSDF, an approach that extends the concept of SDFs for arbitrary articulated objects such as robotic manipulators and humans (Fig. 1). ReDSDF allows us to define complex distance functions to be used as constraints or energy functions to avoid collisions and achieve safe interactions. Our method provides a single deep model that not only preserves structural information in the proximity of articulated bodies, but also effectively provides a signed distance field *at any scale*. Notably, we define a simple yet effective *inductive bias*, i.e. L2-norm, for regularizing our deep signed distance model. This regularization allows us to obtain expressive deep SDFs that can provide precise distance computation over the object's manifold when operating close to it and meaningful level curves when the robot is far from it.

Our deep implicit distance fields give us three significant benefits: first, we can learn and generalize the distance function from data, making it possible to reliably approximate complex manifolds like humans. Second, we can generate smooth and differentiable distance fields that are particularly well-suited for reactive control or trajectory optimization. Finally, deep neural networks allow fast queries of many points in parallel, allowing precise collision avoidance. Our method can be applied to a wide range of robotic applications, from reactive WBC and HRI, to collision-free navigation of aerial robots, as ReDSDF can satisfy any-scale demands depending on the task at hand.

We demonstrate the capabilities of ReDSDF when employed in reactive motion generation tasks. In particular, we provide a qualitative comparison of the produced distance fields of ReDSDF against representative deep-based baselines, showcasing the advantages of our method. Next, we integrate ReDSDF in reactive motion generation. Namely, we design controllers for the WBC of the bimanual mobile manipulator TIAGo++, and for safe HRI in a shared workspace scenario in a novel simulated task that integrates real-human demonstrations. Finally, thanks to the advances in real-time human skeleton tracking and 3D shape detection, we demonstrate the real-world performance of ReDSDF for computing safety distances when interacting with a human. Our results show the potential of ReDSDF to become a major component of various robotic applications, where precise safety constraints are needed.

II. Related Work

SDFs have been studied extensively in the field of computer graphics [10] to reconstruct meshes of surrounding objects given some range information from lasers or cameras. The benefits of SDFs are very well related to problems of robot motion planning and control, collision-checking, and obstacle avoidance [11]. However, while SDFs are locally accurate due to truncation effects, they are challenging to construct from partial observations, i.e., when the whole shape of an object cannot be determined from a single viewpoint. Curless et al. [10] proposed a volumetric integration method that sacrifices the full coverage of the space for improving the local updates based on partial observations, preserving the representation of positions and orientations on the surface of objects. Truncating the field at small negative and positive values produces the Truncated Signed Distance Function (TSDF), in which a point outside the truncated region is located in a narrowband that embeds the surface of the object [12], [13], which allows for better modeling of sensor noise [14]. TSDFs are used overwhelmingly in simultaneous localization and mapping applications [15], [16]. Differently from TSDFs that limit the representation capabilities on points close to the surface, the Euclidean Signed Distance Function (ESDF) provides a method for assessing the free space rather than a fine obstacle area, which is needed for aerial robot mapping and planning [17]. The most popular trajectory optimization methods rely on ESDF to represent the environment [2], [18], [19], as they are effective in static environments, but suffers high computation time for real-time deployment in dynamic environments.

The deep learning revolution in computer graphics and vision led to the emergence of novel methods for object instance segmentation and reconstruction [20], [21] with immediate applications to robot mapping [22]–[24]. Deep implicit functions provide a novel way of training and learning to approximate meshes from point clouds and voxelized data from large simulated datasets [25]. DeepSDF refers to a shape representation method that employs a SDF for training a surface reconstruction network by classifying points as belonging to the surface of a mesh or not [26]. The Equality Constraint Manifold Neural Network (ECoMaNN) approach [27] extends the application of implicit functions to approximate generic equality constraints for robot motion planning. However, these results only apply to simple scenes, and the distance field estimates are accurate only locally. Other deep learning advances allow for 3D human shape reconstruction from point-clouds [28], [29] and RGB frames [30], opening up new possibilities for more precise pose estimation of highly articulated meshes, like humans [31]. While implicit representations have been studied in reconstructing the geometry of articulated objects [32]–[35], distances for query points that are not on the object surface may be ill-defined for robot manipulation.

In the context of reactive motion control, the use of SDFs is challenging due to high computation times that hinder the necessary high control frequency for operating in dynamic environments. To this end, composite SDFs were proposed in [36] for propagating SDFs of moving objects by tracking the occupancy box of those objects in the environment. GPU-accelerated voxels combined with whole-body motion planning were explored in [9]. SDFs are specified across every link of a robot in [37] for obstacle avoidance during

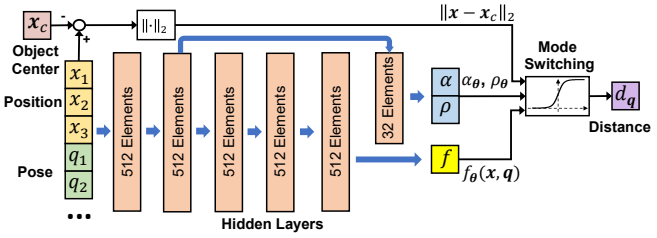


Fig. 2: The Network Structure of ReDSDF

manipulation, while a specific distance function is presented in [38] for self-collision avoidance in WBC.

For HRI, detecting human poses and understanding the effective *safe* workspace of the human is essential. Parting from computer graphics, the modeling of highly articulated meshes and the simulation of natural biomechanical properties of human motion [39] became increasingly interesting for robotics. Human-centric knowledge was naturally transferred to human-robot simulated tasks, where the need for specifying the effective workspace of a human partner when interacting with a robot can be considered as a manipulability area around the human skeleton [40]. A common way of representing safety constraints in humans for HRI tasks is through learning the contour around a human pose projected in a low-dimensional space [41], [42] with primary applications in social human-aware navigation. For closer interactions, many methods represent human links with bounding spheres, from which proximity queries can be made for adapting robot strategies [43], [44].

III. Learning ReDSDF and Robot Control

In what follows, we present our unified vision of distance fields through our method ReDSDF. We show how to learn deep-distance fields for complex articulated objects, such as robotic manipulators and humans. Next, we describe the integration of the learned distance fields in reactive motion generation for robot WBC and reactive HRI. For our method, we assume a target-centric coordinate system centered at the center of mass of the object or on any relevant fixed point, e.g. the pelvis position of a human.

A. Regularized Deep Signed Distance Fields

We exploit two key assumptions to derive a unified vision of *any-scale* deep signed distance fields. First, we assume that the scale of an arbitrary object can be defined using one of its characteristic dimensions, e.g., the radius of the bounding sphere r. Thus, for any point $\boldsymbol{x}$ where the distance to the center of object $\boldsymbol{x}_c$ is $\|\boldsymbol{x} - \boldsymbol{x}_c\|_2 \gg r$, the distance is approximated by $d(\boldsymbol{x}) \approx \|\boldsymbol{x} - \boldsymbol{x}_c\|_2$. Exploiting this assumption makes it possible to build a distance function that unifies most definitions of "distance" to a given target. The second key assumption is the dependence of the distance function on the configuration $\boldsymbol{q}$ of the considered target. For example, $\boldsymbol{q}$ can be the joint positions of a manipulator or the human poses detected by a human tracking system.

We define ReDSDF w.r.t. a given (articulated) target with configuration $\boldsymbol{q}$ at any query point $\boldsymbol{x}$ as the distance

$$d_{\boldsymbol{q}}(\boldsymbol{x}) = (1 - \sigma_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{q}))\, f_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{q}) + \sigma_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{q}) \|\boldsymbol{x} - \boldsymbol{x}_c\|_2, \quad (1)$$

where $\boldsymbol{\theta}$ is the vector of learnable parameters, $\boldsymbol{q}$ is the target's state, $f_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{q})$ is a neural network approximator, and the mode switching function $\sigma_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{q})$ is defined as

$$\sigma_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{q}) = \text{sigmoid}\left(\alpha_{\boldsymbol{\theta}}\left(\|\boldsymbol{x} - \boldsymbol{x}_c\|_2 - \rho_{\boldsymbol{\theta}}\right)\right) \quad (2)$$

where $\alpha_{\boldsymbol{\theta}}$ and $\rho_{\boldsymbol{\theta}}$ are shaping functions, implemented as neural networks. Here, the $\rho_{\boldsymbol{\theta}}$ function defines a (soft) threshold for switching from a distance w.r.t. the closest point on the target's surface to the distance w.r.t. the center of the target. The $\alpha_{\boldsymbol{\theta}}$ function regulates the sharpness of the change between the two regimes.

The network of ReDSDF, depicted in Figure 2, is composed of 5 fully-connected hidden layers. All hidden layers are 512-dimensional. In addition, we connect the features from the second layer with a 32-dimensional layer to compute $\alpha_{\boldsymbol{\theta}}$ and $\rho_{\boldsymbol{\theta}}$. We use ReLU for all layers except for the output layers, which have their specific activation functions. We limit the range of $\rho_{\boldsymbol{\theta}} \in (0.5, 1.5)$ by applying a Sigmoid function and a bias on the output unit. $\alpha_{\boldsymbol{\theta}}$ is constrained to be positive using a Softplus activation. The distance output $f_{\boldsymbol{\theta}}$ is linear.

a) Data generation and augmentation: To train ReDSDF w.r.t. a given (articulated) target, we need to generate an appropriate dataset. We first obtain a point cloud of the object of interest from different viewpoints and at different configurations $\boldsymbol{q}^{[k]}$. Using the Open3D library [45], we estimate the normal direction $\bar{\boldsymbol{n}}_i^{[k]}$ for each point $\boldsymbol{x}_i^{[k]}$ in the point cloud. The normals are used to augment the original point cloud to different distance levels and to regularize the gradient direction of the ReDSDF as shown in the loss function (4). However, the estimation of normal directions may be wrong: training can be unstable if the estimated normals don't match the ones of the neighborhood. This issue can be solved by removing such points from the dataset. Using the estimated normals, we augment the data following the procedure of [27], i.e., we augment the data points to M different distance levels by $\boldsymbol{x}_{ij}^{[k]} = \boldsymbol{x}_i^{[k]} + \bar{\boldsymbol{n}}_i^{[k]} \bar{d}_j, j = 1, 2, \cdots, M$. For each augmented point, we find its closest point from the original point cloud. If the closest point is not the same as the one from which the augmentation starts $\boldsymbol{x}_i^{[k]}$, we reject this augmented point. Differently from [27], we do not use a fixed step size, but we heuristically select a set of step sizes, considering the scale of the object. Finally, we assign to every generated point $\boldsymbol{x}_{ij}^{[k]}$ a weight $\omega_i^{[k]} = M / M_{s,i}^{[k]}$, where $M_{s,i}^{[k]}$ is the number of the augmented points starting from $\boldsymbol{x}_i^{[k]}$ that are not rejected. The set of points generated by this method can be unnecessarily large to train a good distance field. To reduce the dataset without losing precision, we uniformly down-sample a subset of the augmented dataset. Finally, we obtain the dataset

$$\mathcal{D} = \{\boldsymbol{x}_{ij}^{[k]}, \boldsymbol{q}^{[k]}, \bar{d}_j, \bar{\boldsymbol{n}}_i^{[k]}, \omega_i^{[k]}\}. \quad (3)$$

b) Human-based data generation: An important factor, when considering distance to humans, is the high variability of body shapes. This is particularly important when we need to perform close quarter interaction, e.g., for handovers and

shared workspace interactions. Instead of using skeleton-based human reconstruction, we leverage the realistic 3D model of the human body, A Skinned Multi-Person Linear Model (SMPL) [31]. Human 3D meshes are obtained directly from the SMPL model, given the human poses. The points and normals are extracted from the triangular mesh-elements by computing the center of each triangle and the cross product of the edges. We generate the augmented data as mentioned in previous section. We use the AMASS [46] dataset to build up a comprehensive dataset that contains various human poses. In this work, we choose $10,000$ human configurations from the AMASS dataset.

c) Training: We train the model by optimizing the following loss function

$$\begin{aligned}\mathcal{L}(\mathcal{D}) = \sum_{\mathcal{D}} & \omega_{\boldsymbol{q}}(\boldsymbol{x}) \left(\bar{d}_{\boldsymbol{q}}(\boldsymbol{x}) - d_{\boldsymbol{q}}(\boldsymbol{x})\right)^2 \\ & + \left(\|D_{\boldsymbol{q}}(\boldsymbol{x})\bar{\boldsymbol{n}}_{\boldsymbol{q}}(\boldsymbol{x})\|_2^2 + \|N_{\boldsymbol{q}}(\boldsymbol{x})\nabla_{\boldsymbol{x}} d_{\boldsymbol{q}}(\boldsymbol{x})\|_2^2\right) \\ & + \gamma\rho_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{q})^2, \end{aligned} \tag{4}$$

where $D_{\boldsymbol{q}}(\boldsymbol{x}) = \text{null}\left(\nabla_{\boldsymbol{x}} d_{\boldsymbol{q}}(\boldsymbol{x})\right)$ and $N_{\boldsymbol{q}}(\boldsymbol{x}) = \text{null}\left(\bar{\boldsymbol{n}}_{\boldsymbol{q}}(\boldsymbol{x})\right)$ are, respectively, the null-space of the gradient of the distance field i.e., the space tangent to the isolines at every point $\boldsymbol{x}$, and the null space of the normals i.e., the tangent planes to the object. The first component of the loss is computing the squared error of the network distance prediction w.r.t. the target value. The second component of the loss is similar to the one proposed by [27] to align the estimated normals $\bar{\boldsymbol{n}}_{\boldsymbol{q}}(\boldsymbol{x})$ with the normals of the ReDSDF model. The last component of the loss is a regularization term, where γ is a regularization coefficient, set in our experiments to 0.02. This regularization is trying to impose the inductive bias of the distance to every point of the state-space. Reducing the output of the learnable function $\rho_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{q})$ has the effect of switching the distance regime of the network as soon as possible. This regularization is important in particular where the dataset is sparse. The dataset is split into training, validate and test datasets with the ratio $0.8, 0.1, 0.1$.

B. Robot Motion Control with ReDSDF

ReDSDF can be readily employed within any control and planning framework to provide real-time constraints' inference. Reactive motion generation provides a nice framework to showcase the benefit of our deep-distance field for robot control. Namely, we will describe the integration of ReDSDF in a framework for whole-body motion control based on Artificial Potential Fields (APF) [47]. Note that ReDSDF can be integrated to any other type of reactive motion generation, such as Riemannian Motion Policies (RMP) [4], CEP [48]. ReDSDF is not restricted to reactive motion generation, but can be employed by any constrained planner, both as a differentiable constraint and as an energy function. For the sake of simplicity, we focus on APF-based reactive control, as it directly allows showing the benefits of ReDSDF.

We consider the settings where the APFs compute a velocity field for the robot's end-effector. We define a simple PID controller for tracking the end-effector's velocity, but more sophisticated options can be used to generate the task-oriented velocity signal. On top of this, we add an obstacle avoidance velocity signal. Since our method exploits deep neural networks, it is possible to rapidly compute the whole set of collision points in a batch. To perform precise obstacle avoidance on a bimanual mobile manipulator robot of Fig. 5.a, we evaluate the ReDSDF over multiple points sampled on the shell of the arms w.r.t. the obstacle. It is noteworthy that the target obstacle can as well be the other arm or any other part of the articulated body of the robot, allowing us to perform self-collision avoidance. For avoiding self-collisions with the other arm of a bimanual robot, we learn a ReDSDF model of the robot with only a single arm. The learned ReDSDF can be used to avoid obstacle for the other arm. We leverage the symmetry of the robot to construct the ReDSDF for the untrained arm.

For each target obstacle o, we compute the obstacle avoidance energy field as

$$E_o(\boldsymbol{x}) = \begin{cases} 0 & d_{\boldsymbol{q}}(x) > \kappa \\ \frac{\bar{v}}{2\kappa}\left(d_{\boldsymbol{q}}(\boldsymbol{x}) - \kappa\right)^2 & 0 \leq d_{\boldsymbol{q}}(x) \leq \kappa \end{cases}, \tag{5}$$

where κ is the maximum obstacle avoidance distance, and $\bar{v}$ is the repulsive velocity coefficient. For $0 \leq d_{\boldsymbol{q}}(x) \leq \kappa$, we obtain the obstacle avoidance velocity as follows

$$\dot{\boldsymbol{x}}_o = -\nabla_{\boldsymbol{x}} E_o(\boldsymbol{x}) = -\frac{\bar{v}}{\kappa}\left(d_{\boldsymbol{q}}(x) - \kappa\right)\nabla_{\boldsymbol{x}} d_{\boldsymbol{q}}(\boldsymbol{x}). \tag{6}$$

While we could also define the energy for negative distance, we choose not to do it as we are considering the rigid obstacle avoidance task, as a negative distance implies a collision with the obstacle. In case of collision, we stop the robot from taking any other action.

The velocity at each Points of Interest (PoI) $\boldsymbol{x}_i$ is computed by combining the repulsive velocity contribution from the set of obstacles $\mathcal{O}$ in the environment:

$$\dot{\boldsymbol{x}}_i = \frac{1}{|\mathcal{O}|}\sum_{o \in \mathcal{O}} \dot{\boldsymbol{x}}_{i,o}. \tag{7}$$

To transform this velocity from task-space velocity to the robot configuration space velocity $\boldsymbol{q}_r$, we can use the Jacobian matrix $J_i(\boldsymbol{q}_r)$ computed in the coordinate system defined in $\boldsymbol{x}_i$. While the most common solution is to project the velocity into the configuration space using the Jacobian pseudoinverse $J^{\dagger}(\boldsymbol{q}_r)$, in this work we use the Jacobian transpose$\dot{\boldsymbol{q}}_{r,i} = J_i^{\intercal}(\boldsymbol{q}_r)\dot{\boldsymbol{x}}_i$. This choice is well-known in the literature [49], [50], and it is often used to avoid issues with singularities without losing the convergence guarantee. Finally, we compute the joint velocity by combining the obstacle avoidance velocity with the task velocity $\dot{\boldsymbol{x}}_t$ as

$$\dot{\boldsymbol{q}}_r = J_t^{\intercal}(\boldsymbol{q}_r)\dot{\boldsymbol{x}}_t + \sum_{i}^{N} \dot{\boldsymbol{q}}_{r,i}, \tag{8}$$

where N is the number of interest points and $J_t(\boldsymbol{q}_r)$ is the Jacobian computed in the end-effector frame. Instead of using clipping to enforce the joint velocity limits, we rescale all joints' velocities with an appropriate constant, maintaining the same task-space direction while keeping every joint inside its limit.

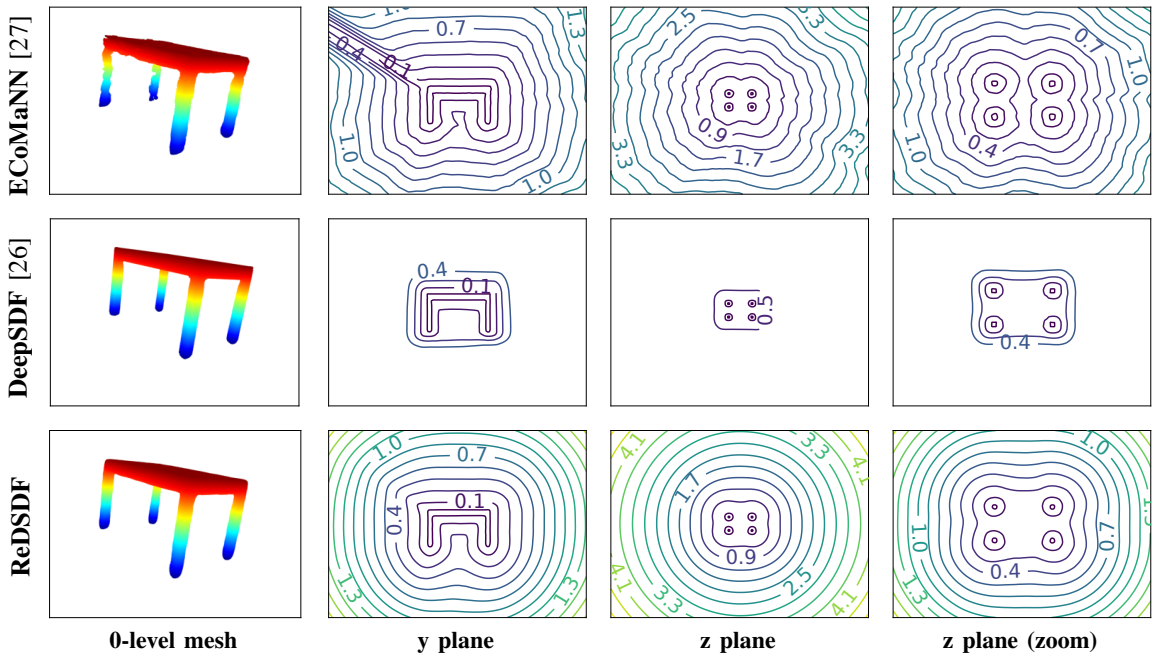


Fig. 3: Reconstruction of the table distance function

IV. Experiments

To evaluate the effectiveness and applicability of ReDSDF to produce smooth distance fields for robot control, we perform qualitative and quantitative comparisons with various baseline methods in a set of simulated control tasks. First, we evaluate the quality of the produced distance fields of ReDSDF against state-of-the-art methods. Next, we test our method as a component of reactive motion generation for WBC and robot control in a shared human-robot workspace. Last, we empirically evaluate the performance of ReDSDF in a realistic HRI scenario, as seen in Fig. 1, whose results can be found in the attached video and in our project site `https://irosalab.com/2022/02/28/redsdf/`.

A. *Quality of the distance field*

We compare ReDSDF against the state-of-the-art methods ECoMaNN [27] and DeepSDF [26]. As both baselines were introduced for approximating the manifold for static objects or sampled points cloud, we compare against them with a static object, e.g., a table in Fig. 3 and a human mesh with fixed pose Fig. 4. For the baseline approaches, we directly apply the available source code provided by the authors. We use the same databases for all methods, but the training data are generated using the approach-specific data augmentation technique. As the computation of an accurate metric for this task is challenging, we will only provide a qualitative evaluation. However, we argue that the performance difference w.r.t. other state-of-the-art methods is clear, and thus this evaluation is already sufficiently convincing that ReDSDF can provide more rational distance fields for robotics environments.

Fig. 3 depicts the results for learning the table manifold. As we can see, the ECoMaNN reconstruction contains artifacts in the estimation of the 0-level curve of the constraint, while our approach reconstructs the mesh without issues. The distance field estimation of ECoMaNN is problematic due to wrong normal estimations on the manifold. Instead, DeepSDF reconstructs the manifold well. However, DeepSDF normalizes every mesh file individually and restricts the network's output by a tanh, limiting the distance field's output range. Contrarily, we can observe the superiority of ReDSDF in generating not only a clear geometric manifold of the object, but also smoother and cleaner distance fields at any scale. Next, we compare the distance field reconstruction from a human mesh file in Fig. 4. All methods reconstruct roughly the human shape. However, ReDSDF provides more precise details like the human fingers and ears. The weighting technique significantly improves the reconstruction of the points in cluttered areas. Furthermore, when looking at the shape of the distance field for points far from the center of mass, we can see that ECoMaNN unnaturally deforms the space, e.g., stretching the distance along the hands. DeepSDF, on the other hand, suffers the same scaling problem, while our approach exploits the inductive bias to enforce a well-behaved, even if approximate, distance field.

For ReDSDF, we generate augmented fine-grained data points in the proximity of the target-surface and more sparse data points as we go away from it. This augmentation allows us to have points at any scale, improving the smoothness of the function and obtaining a precise reconstruction of the 0-level constraint. The inductive bias we use to regularize the distance field plays a major role in the quality of the reconstruction of the distance function at any scale. Indeed, this is one of the reasons why we do not need to generate augmentation points in the whole state space, and we can make them sparser going away from the manifold without losing the smoothness of the approximated field.

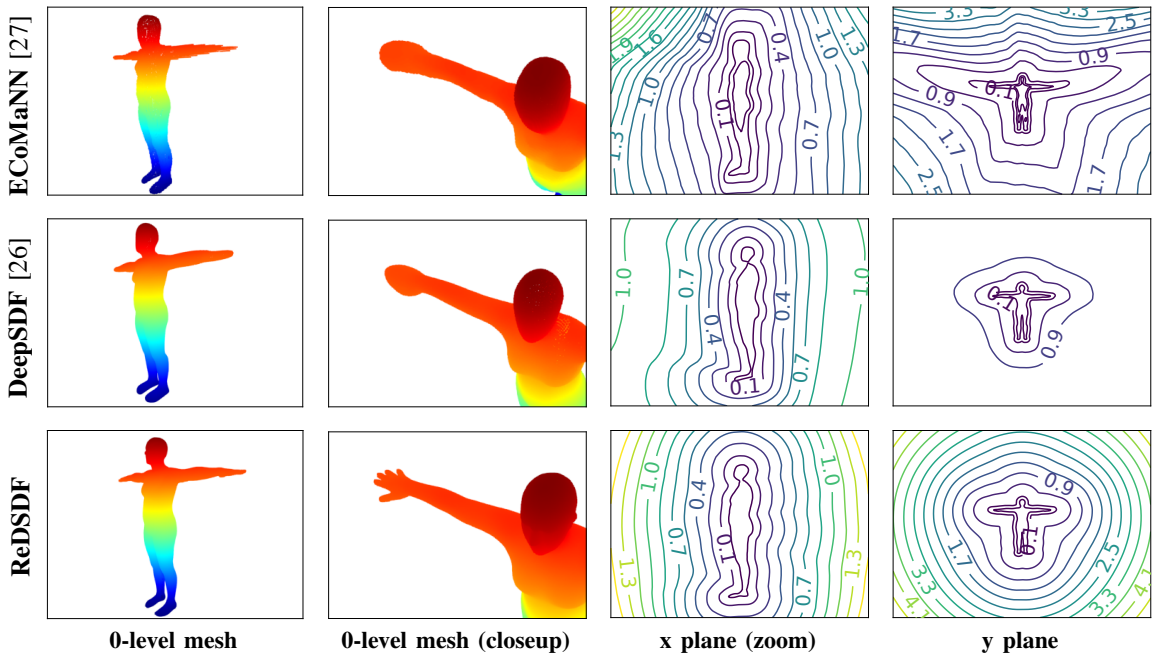


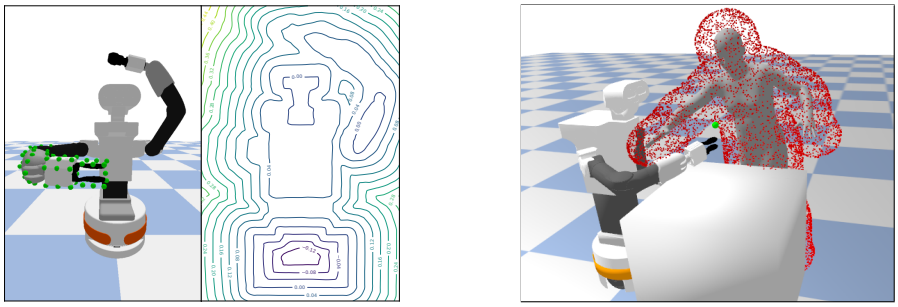
Fig. 4: Reconstruction of the human distance function

B. *Whole-body Control*

For evaluating the applicability of ReDSDF for robot control, we devise a WBC for the bimanual mobile manipulator TIAGo++ robot (Fig. 5a). For our simulated experiment, we focus on the control of the two arms and torso (15 degrees of freedom) but assume a stationary base. However, our approach can be trivially extended to generate a base velocity. We generate the augmented dataset of $10,000$ configurations that only contains one arm (left in our experiment). The distance field of the other arm can be computed by mirroring the PoI through the symmetric plane. We defined 66 PoI distributed on the surface of each arm and computed the distance using ReDSDF w.r.t. the other arm and the robot body. Note that the model is learned by excluding each controlled arm from the robot itself; otherwise, every point would be considered in collision. The learned ReDSDF and PoI of the other arms is illustrated in Fig. 5a. We use a PID controller to provide the control velocity of the end-effector to reach the target. The repulsive velocity that avoids the collision is computed as described in (6). We compare our WBC to one without obstacle avoidance and to another one that uses APFs but computes the distance using spheres to approximate the robot model, a common approach in the literature. We group the spheres into 3 subgroups, i.e., body (18 spheres), left and right arm (6 spheres each). The distances are computed between the spheres from different groups. By adjusting the distance threshold of κ and the velocity coefficient $\bar{v}$, we can implement a task-oriented controller that cares more about the end-task, or a cautious controller that places more emphasis on collision avoidance.

Our experimental results are summarized in the first part of Table I. We conducted $1,000$ experiments with different reaching targets for each arm. To generate the target points, we first sample a random point in space for the first arm, and then we generate a second point for the second arm in the vicinity of the first point. The target points that are too close to the robot are rejected. Each task is simulated for 30 s with a control frequency of 60 Hz. The episode terminates once the simulator detects a collision based on the meshes. We regard the task as successful when the end-distance of the end-effector to the target is less than 3 cm. We report the success rate percentage, the absolute number of collisions, and the mean and confidence interval of the final position error, the reach time, smoothness of the generated trajectory τ as: $s(\tau) = \sum_{\boldsymbol{q}_r(i)\in\tau} \|\ddot{\boldsymbol{q}}_r(i)\|^2$, and of the time spent for computing collisions (C. time).

Due to our design choices, both the sphere-based distance computation and ReDSDF are comparable in terms of success rate when the controller is task-oriented, but ReDSDF only has seven collisions. For the cautious controller, ReDSDF provides zero collisions and satisfactory performance, while the heuristic sphere-based method requires strong repulsive forces to eliminate collisions, hurting the task performance. For the smoothness metric, we consider both changes in the direction of the velocity and changes in magnitude, and for a fair comparison, we only compute the smoothness for the collision-free episodes. ReDSDF generates a continuous distance field, which results in a much smoother controller. For the heuristic, the distance and normal directions are chosen based on the closest sphere of each PoI. The normals may change drastically when the closest sphere varies, which results in jittery movements. The distance computation time is evaluated on an AMD Ryzen 7 3700X 8-Core Processor with a GeForce RTX 2080 SUPER. We compute the distance of the 132 PoI (66 PoI per arm) with ReDSDF in batch on the GPU using PyTorch. For the sphere-based method, 288 distance queries are requested at each time in batch on the GPU to speed up its performance.



(a) (b)

Fig. 5: (a) ReDSDF for WBC. **Left:** PoI of the controlled arm that we check to avoid collisions with the rest of the robot. **Right:** The distance field of the robot, excluding the controlled arm. (b) Human-robot shared-workspace simulated task: human and robot execute sequential pick-and-place-like actions in a tight workspace, where the robot should avoid collisions with the human and the table. The red point cloud represents the 0.1-level ReDSDF of the human and is used for collision-avoidance by the robot controller.

WBC results

	No avoidance	**Sphere-based (task-oriented)**	**ReDSDF (task-oriented)**	**Sphere-based (cautious)**	**ReDSDF (cautious)**
Success rate	44.8%	80.9%	82.6%	38.3%	73.2%
# collisions	548/1000	49/1000	7/1000	0/1000	0/1000
Final err. (cm)	10.42 ± 0.59	1.42 ± 0.14	1.25 ± 0.11	7.21 ± 0.42	1.95 ± 0.15
Reach time (s)	6.17 ± 0.42	12.58 ± 0.58	12.74 ± 0.58	23.98 ± 0.55	15.56 ± 0.65
Smoothness	10.50 ± 0.06	811.88 ± 183.40	11.62 ± 0.11	3557.51 ± 284.15	12.34 ± 0.20
C. time (ms)	0.10 ± 5.49e-5	5.79 ± 6.28e-3	3.27 ± 5.61e-3	5.97 ± 5.68e-3	3.23 ± 3.18e-3

Shared Workspace results

	No avoidance	**Sphere-based (task-oriented)**	**ReDSDF (task-oriented)**	**Sphere-based (cautious)**	**ReDSDF (cautious)**
# collisions	935/1000	171/1000	95/1000	86/1000	27/1000
# targets	2.172 ± 0.19	6.35 ± 0.18	5.78 ± 0.16	4.44 ± 0.14	4.73 ± 0.16
Smoothness	71.95 ± 0.85	2120.00 ± 198.35	135.56 ± 3.67	2286.69 ± 174.82	624.14 ± 43.50

TABLE I: Results of WBC and shared workspace experiments in simulation

The computation time for ReDSDF is 50% less than the heuristic. We want to point out that the computation load of ReDSDF grows linearly w.r.t. the number of PoI, while for the sphere-based method it grows quadratically. This advantage allows us to achieve more query points and finer computations of the distance field.

C. Shared Human-Robot workspace

To test the applicability of ReDSDF for reactive motion generation for challenging HRI tasks, we built a simulation task where a human and a robot are performing sequential pick-and-place-like actions in a shared workspace. To simulate realistic movements, we record a set of trajectories from an actual human performing the task, and we infer the SMPL trajectories with the help of VIBE [30]. We replay these trajectories in our PyBullet simulator, constituting a novel way of building simulation tasks for HRI with the robot dynamically interacting with the human (Fig. 5b). Instead of reaching the fixed targets as in WBC task, we generate 9 targets for each task to mimic the pick-and-place scenario, a new target will be activated once the old target has been reached. We assume that the human is not responsive to the robot in the current setting. While this assumption is unrealistic, we want to test the reactiveness of the robot controller while interacting in a tight workspace with a human. If the human actively avoids the robot, it would be impossible to test if our controller successfully avoids collisions. Indeed, we can see the proposed scenario as an approximation of a human not paying attention to the robot while performing a shared-workspace task.

Table I contains the results of this experiment. We compared the collision avoidance behavior based on ReDSDF with two simple baselines. The first, trivial baseline is to plan without considering the human at all. The second baseline is a prevalent approach in the literature that considers only the human hand's position and reacts to hand movements. We developed similar task-oriented and cautious controller as in Sec. IV-B. For all methods, we also include a potential field to prevent collision with the table. Inspecting the results, we can observe that our controllers with ReDSDF can provide smoother distance fields that result in smoother robot movements, with significantly fewer collisions compared to the baseline methods. Even though in the target-oriented case the heuristic achieves more targets, it suffers from a higher number of collisions. Overall, our results validate our assumption that ReDSDF is effective and smooth when used in real-time as a constrained function for safe and reactive HRI. As proof of concept, Fig. 1 and the accompanying video provides a real-world experiment in a realistic HRI handover scenario, where we can inspect the any-scale smooth performance of ReDSDF.

V. Conclusions

Safety is a critical property for autonomous robots, both for planning actions that respect their hardware and the surrounding world and when robots interact with humans in shared workspaces. In this work, we presented the Regularized Deep Signed Distance Fields (ReDSDF) framework, which generalizes the concept of SDF to arbitrary articulated objects. We regularize the training of the distance function

with an inductive bias to ensure smooth distance fields that can be used at any scale. Using the proposed technique, we obtain high-quality, smooth distance fields for any point in the space. Our method can be easily deployed to generate reactive motions both in the context of a whole-body control (WBC) and in a human-robot shared workspace scenario, using real human data. The reactive motion can reliably avoid obstacles while reducing the interference considerably with the main task, allowing us to outperform baselines methods by a significant margin in many metrics. The ReDSDF structure can be easily deployed in real robot applications, as it can provide accurate distance fields in different resolutions, from rough distance estimates for obstacle avoidance during navigation to fine-grained resolutions in close interactions with humans. Though we demonstrated the applicability of ReDSDF to reactive motion generation, thanks to its smooth differentiable nature, it can be easily integrated into different pipelines, such as constrained task and motion planning or constrained reinforcement learning.

REFERENCES

[1] M. da Silva Arantes, C. F. M. Toledo, B. C. Williams, and M. Ono, "Collision-free encoding for chance-constrained nonconvex path planning," *IEEE TRO*, 2019.

[2] N. Ratliff, M. Zucker, J. A. Bagnell, and S. Srinivasa, "Chomp: Gradient optimization techniques for efficient motion planning," in *ICRA*, 2009.

[3] P. Liu, D. Tateo, H. B. Ammar, and J. Peters, "Robot reinforcement learning on the constraint manifold," in *CoRL*, 2022.

[4] N. D. Ratliff, J. Issac, D. Kappler, S. Birchfield, and D. Fox, "Riemannian motion policies," *arXiv:1801.02854*, 2018.

[5] F. S. Barbosa, B. Lacerda, P. Duckworth, J. Tumova, and N. Hawes, "Risk-aware motion planning in partially known environments," *IEEE CDC*, 2021.

[6] C.-P. Lam, C.-T. Chou, K.-H. Chiang, and L.-C. Fu, "Human-centered robot navigation—towards a harmoniously human–robot coexisting environment," *IEEE TRO*, 2010.

[7] P. T. Singamaneni, A. Favier, and R. Alami, "Human-aware navigation planner for diverse human-robot interaction contexts," in *IROS*, 2021.

[8] F. Ferraguti, C. T. Landi, S. Costi, M. Bonfè, S. Farsoni, C. Secchi, and C. Fantuzzi, "Safety barrier functions and multi-camera tracking for human–robot shared environment," *Robotics and Autonomous Systems*, vol. 124, p. 103388, 2020.

[9] M. N. Finean, W. Merkt, and I. Havoutis, "Simultaneous scene reconstruction and whole-body motion planning for safe operation in dynamic environments," in *IROS*, 2021.

[10] B. Curless and M. Levoy, "A volumetric method for building complex models from range images," in *SIGGRAPH*, 1996.

[11] N. Ratliff, M. Toussaint, and S. Schaal, "Understanding the geometry of workspace obstacles in motion optimization," in *ICRA*, 2015.

[12] D. R. Canelhas, "Truncated signed distance fields applied to robotics," Ph.D. dissertation, Örebro University, 2017.

[13] H. Oleynikova, A. Millane, and Z. Taylor et al., "Signed distance fields: A natural representation for both mapping and planning," in *RSS Workshops*, 2016.

[14] K. Saulnier, N. Atanasov, G. J. Pappas, and V. Kumar, "Information theoretic active exploration in signed distance fields," in *ICRA*, 2020.

[15] S. Izadi, D. Kim, O. Hilliges, and D. Molyneaux et al., "Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera," in *ACM Symp. on User interface software and tech.*, 2011.

[16] E. Vespa, N. Nikolov, and M. Grimm et al., "Efficient octree-based volumetric slam supporting signed-distance and occupancy mapping," *IEEE RA-L*, 2018.

[17] L. Han, F. Gao, B. Zhou, and S. Shen, "Fiesta: Fast incremental euclidean distance fields for online motion planning of aerial robots," in *IROS*, 2019.

[18] M. Mukadam, J. Dong, X. Yan, F. Dellaert, and B. Boots, "Continuous-time gaussian process motion planning via probabilistic inference," *IJRR*, 2018.

[19] J. Schulman, Y. Duan, and J. Ho et al., "Motion planning with sequential convex optimization and convex collision checking," *IJRR*, 2014.

[20] L. Han, T. Zheng, L. Xu, and L. Fang, "Occuseg: Occupancy-aware 3d instance segmentation," in *CVPR*, 2020.

[21] W. Xu, H. Wang, F. Qi, and C. Lu, "Explicit shape encoding for real-time instance segmentation," in *ICCV*, 2019.

[22] B. Xu, W. Li, and D. Tzoumanikas et al., "Mid-fusion: Octree-based object-level multi-instance dynamic slam," in *ICRA*, 2019.

[23] M. Grinvald, F. Furrer, and T. Novkovic et al., "Volumetric instance-aware semantic mapping and 3d object discovery," *IEEE RA-L*, 2019.

[24] J. McCormac, R. Clark, M. Bloesch, A. Davison, and S. Leutenegger, "Fusion++: Volumetric object-level slam," in *3DV*, 2018.

[25] S. Peng, M. Niemeyer, L. Mescheder, M. Pollefeys, and A. Geiger, "Convolutional occupancy networks," in *ECCV*, 2020.

[26] J. J. Park and P. Florence et al., "Deepsdf: Learning continuous signed distance functions for shape representation," in *CVPR*, 2019.

[27] G. Sutanto, I. M. R. Fernández, and P. Englert et al., "Learning equality constraints for motion planning on manifolds," in *CoRL*, 2020.

[28] H. Jiang, J. Cai, and J. Zheng, "Skeleton-aware 3d human shape reconstruction from point clouds," in *CVPR*, 2019.

[29] B. L. Bhatnagar, C. Sminchisescu, C. Theobalt, and G. Pons-Moll, "Combining implicit function learning and parametric models for 3d human reconstruction," in *ECCV*, 2020.

[30] M. Kocabas, N. Athanasiou, and M. J. Black, "Vibe: Video inference for human body pose and shape estimation," in *CVPR*, 2020.

[31] M. Loper, N. Mahmood, J. Romero, G. Pons-Moll, and M. J. Black, "Smpl: A skinned multi-person linear model," *ACM TOG*, 2015.

[32] B. Deng, J. P. Lewis, T. Jeruzalski, G. Pons-Moll, G. Hinton, M. Norouzi, and A. Tagliasacchi, "Nasa neural articulated shape approximation," in *ECCV*. Springer, 2020.

[33] J. Mu, W. Qiu, A. Kortylewski, A. Yuille, N. Vasconcelos, and X. Wang, "A-sdf: Learning disentangled signed distance functions for articulated shape representation," in *ICCV*, 2021.

[34] A. Gropp, L. Yariv, N. Haim, M. Atzmon, and Y. Lipman, "Implicit geometric regularization for learning shapes," in *ICML*. PMLR, 2020.

[35] M. Atzmon and Y. Lipman, "Sald: Sign agnostic learning with derivatives," in *ICLR*, 2020.

[36] M. N. Finean, W. Merkt, and I. Havoutis, "Predicted composite signed-distance fields for real-time motion planning in dynamic environments," in *ICAPS*, 2021.

[37] S. Xu, G. Li, and J. Liu, "Obstacle avoidance for manipulator with arbitrary arm shape using signed distance function," in *ROBIO*, 2018.

[38] J. J. Quiroz-Omaña and B. V. Adorno, "Whole-body control with (self) collision avoidance using vector field inequalities," *IEEE RA-L*, 2019.

[39] N. I. Badler, C. B. Phillips, and B. L. Webber, *Simulating humans: computer graphics animation and control*. Oxford Uni. Press, 1993.

[40] N. Vahrenkamp, H. Arnst, and M. Wächter et al., "Workspace analysis for planning human-robot interaction tasks," in *Humanoids*, 2016.

[41] P. Papadakis, A. Spalanzani, and C. Laugier, "Social mapping of human-populated environments by implicit function learning," in *IROS*, 2013.

[42] B. Lewandowski, T. Wengefeld, and S. Müller et al., "Socially compliant human-robot interaction for autonomous scanning tasks in supermarket environments," in *RO-MAN*, 2020.

[43] J. Corrales, F. Candelas, and F. Torres, "Safe human–robot interaction based on dynamic sphere-swept line bounding volumes," *Robotics and Computer-Integrated Manufacturing*, 2011.

[44] P. Svarny, M. Tesar, J. K. Behrens, and M. Hoffmann, "Safe physical hri: Toward a unified treatment of speed and separation monitoring together with power and force limiting," in *IROS*, 2019.

[45] Q.-Y. Zhou, J. Park, and V. Koltun, "Open3D: A modern library for 3D data processing," *arXiv:1801.09847*, 2018.

[46] N. Mahmood, N. Ghorbani, N. F. Troje, and et al., "AMASS: Archive of motion capture as surface shapes," in *ICCV*, 2019.

[47] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," in *ICRA*, 1985.

[48] J. Urain, A. Li, P. Liu, C. D'eramo, and J. Peters, "Composable energy policies for reactive motion generation and reinforcement learning," in *Robotics: Science and Systems (RSS)*, 2021.

[49] S. Chiaverini, L. Sciavicco, and B. Siciliano, "Control of robotic systems through singularities," in *Advanced Robot Control*, 1991.

[50] S. R. Buss, "Introduction to inverse kinematics with jacobian transpose, pseudoinverse and damped least squares methods," *IEEE Journal of Robotics and Automation*, 2004.