2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Generating Force Vectors from Projective Truncated Signed Distance Fields for Collision Avoidance and Haptic Feedback

Seongjin Bien[1*], Abdeldjallil Naceri[1], Luis Figueredo[2] and Sami Haddadin[1]

***Abstract*— Signed Distance Fields are a common surface representation method widely used for both 3D mapping and obstacle avoidance. While the former traditionally uses projective Truncated Signed Distance Fields (TSDF), the latter often requires a complete Euclidean Signed Distance Field (ESDF) representation of the environment. In this paper, we propose a unified system by combining both methods to generate force vectors to nearby obstacles from a TSDF-based 3D reconstruction. We introduce a new merging scheme to better capture the geometry of the object, with no post-processing requirements, and a way to increase the effective range of the system. Validation experiments demonstrate the accuracy of the force vector calculation by comparing it against an ideal simulated environment. The flexibility of the system is demonstrated by implementing a haptic feedback teleoperation setup, which is validated through a user study in a teleoperation task. Through this, it is shown that the proposed method provides a statistically significant improvement to the task. Finally, a brief description on future improvements to the system is presented.**

I. Introduction

As technology advances, robots are expected to operate in larger, more unstructured environments containing complex objects that cannot be represented by simple shapes. Since their properties are often unknown, processing them offline is impractical. For tasks like manipulation, fine-grained geometric information is crucial. Additionally, persistently representing the environment helps avoid collisions with objects not under direct observation and quickly updates changes, especially when influenced by other agents like humans. This information is vital during teleoperation, where full contextual data isn't available. Existing object collision avoidance methods often simplify representations or rely on raw point cloud data, which can miss finer details, lack persistence, or fail due to occlusions.

With such factors in mind, we propose a new force vector calculation method that leverages the well-established signed distance field (SDF)-based 3D reconstruction that can be used for collision avoidance. Furthermore, the method's flexible nature allows its application as a haptic feedback generator, which can assist users in teleoperation scenarios.

*:Corresponding author. This work was supported by the Lighthouse Initiative Geriatronics by StMWi Bayern (Project X, grant no. IUK-1807-0007// IUK582/001) and LongLeif GaPa gGmbH (Project Y). [1]: Affiliation with Munich Institute of Robotics and Machine Intelligence (MIRMI), Technische Universität München (TUM), Germany. [1]: Affiliation with School of Computer Science, University of Nottingham, UK. Email:{s.bien, djallil.naceri, haddadin}@tum.de, Luis.Figueredo@nottingham.ac.uk

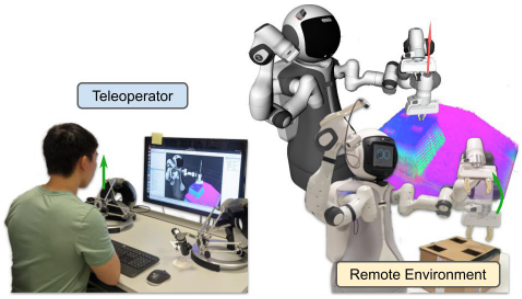


Fig. 1: The proposed system in action, showing the reconstructed environment scanned by the robot and the resulting force vector that is fed into the robot's controller (red arrow) and the haptic teleoperation device (green arrow).

II. Related Works

A. Object Collision Avoidance

Several works have explored using RGBD sensors for object collision avoidance [1], [2], [3]. These methods typically involve placing one or more static RGBD cameras to monitor the workspace and using continuous camera feeds for real-time collision avoidance with dynamic obstacles. Collision avoidance is often achieved through artificial potential fields, generating repulsive or attractive forces between the robot, obstacles, and goals to guide the robot in a collision-free manner.

A common issue with these methods is occlusion, where obstacles are partially or fully hidden from view. This is addressed by either conservatively estimating the obstacle's position to minimize collision risk [3] or adding more sensors to reduce occlusions [1], [2]. However, the former can lead to suboptimal strategies, while the latter is impractical for mobile robots due to the larger workspace and limited sensor coverage.

Other approaches approximate object geometry using simple shapes like spheres and cylinders [4], [5], [6], enabling faster computation but resulting in conservative estimates. A recent method by Eckhoff et al. achieved true real-time collision avoidance with a 1kHz command frequency by creating a digital twin of the scene, but this required prior knowledge of object locations and could not handle dynamic objects [6].

B. SDF-Based Collision Avoidance

Instead of using live data from an RGBD sensor, an alternative way is to create a persistent reconstruction of the environment for collision avoidance. A family of 3D reconstruction methods employ signed distance fields (SDF),

which are voxel-based representations of the environment comprised of discrete voxels that store the distance to the closest surface. SDFs have gained a large amount of attention for performing environmental collision avoidance on unmanned aerial vehicle (UAV) platforms [7], [8], [9], as the stored distance values can be interpreted as discrete values of a potential field function for trajectory planning. Many such applications calculate ESDFs based on observations, which provides a more accurate potential field. While calculation of ESDFs is computationally expensive, works such as VoxBlox [7] and FIESTA [8] deal with this by reducing the ESDF's resolution paired with sophisticated update algorithms, allowing for real-time computation at the cost of losing finer details. They have also found applications in both wheeled and legged robots for environmental collision avoidance [5], [4].

C. Haptic Feedback from 3D Environments

In addition to audiovisual feedback, haptic feedback represents another useful modality for users to gain more information about the environment during teleoperation. Existing works on generating haptic feedback from 3D environment observations fall into two main categories: visual detection, and using raw point clouds.

Detection-based methods first locate the target object using techniques like deep learning or color detection, then generate haptic feedback based on it. A common strategy involves enveloping the target in a potential field to guide the teleoperator [10], [11]. Another approach creates a virtual fixture between the object and the robot, then using a controller to both control the robot and provide haptic feedback [12].

For the latter category, many works are based on Rydén et al.'s method, where a control point is enveloped by several spheres of different radii, and forces are calculated based on the location of point clouds that enter these spheres [13], [14]. This concept has been applied in context of a teleoperated robotic welding task [15] or combined with a digital twin to improve robustness against delays during teleoperation [16].

D. Contribution

The literature review reveals two important gaps: 1) they often require continuous observation of the environment, and cannot deal with occlusions in an elegant manner, and 2) the resolution of SDF-based methods are often reduced to deal with memory costs. Our work attempts to address these two issues by making the following contributions to the field:

1) A new merging scheme for projective TSDFs that retains the benefits of both minimum merging [17] and average merging for more accurate TSDF construction;
2) A method for calculating forces from TSDFs that can be used for both collision avoidance and haptic feedback;
3) Implementation of the framework in a service humanoid robot and a teleoperation user study to validate the usefulness of the proposed system.

III. Method

A. KinectFusion

1) Background to KinectFusion: KinFu represents surfaces in the environment implicitly using a global projective TSDF. The field is discretized into voxels at a chosen resolution and stored into GPU memory. A ray is projected from the camera until it reaches an obstacle, and the distance the ray travels until it reaches the obstacle at each discrete position is stored in the voxels. Negative distance value indicates that the voxel is located inside the object. The surface of the obstacle is thus calculated as the zero-crossing between cells.

To deal with differing distances from multiple camera viewpoints, the authors of KinFu used average merging to combine multiple observations across time[18]:

$$f_k(\boldsymbol{p}) = \frac{w_{k-1}(\boldsymbol{p}) f_{k-1}(\boldsymbol{p}) + w_{\boldsymbol{M}_k}(\boldsymbol{p}) f_{\boldsymbol{M}_k}(\boldsymbol{p})}{w_{k-1}(\boldsymbol{p}) + w_{\boldsymbol{M}_k}(\boldsymbol{p})} \tag{1}$$

$$w_k(\boldsymbol{p}) = w_{k-1}(\boldsymbol{p}) + w_{\boldsymbol{M}_k}(\boldsymbol{p}) \tag{2}$$

$$w_0(\boldsymbol{p}) = 1 \tag{3}$$

$$w_{\boldsymbol{M}_k}(\boldsymbol{p}) = 1 \tag{4}$$

where k is the current step, $\boldsymbol{p}$ is the voxel, $f(\boldsymbol{p})$ is the SDF value, $w(\boldsymbol{p})$ is the weight, and the subscript M_k is the current observation. They noted that simply setting $w_{\boldsymbol{M}_k} = 1$, i.e. simple averaging, provides good results.

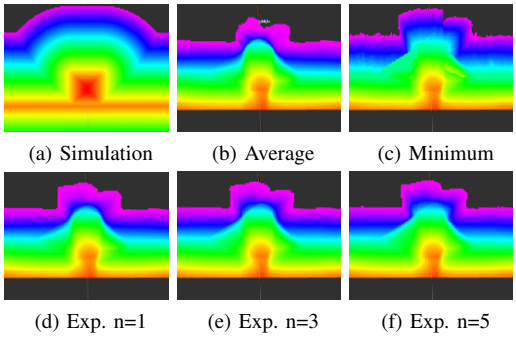
Populating every cell that the projected ray passes through would result in a huge memory requirement. Also, only cells close to surfaces are necessary for surface reconstruction. As such, cells beyond the threshold distance μ from the zero-crossings are truncated.

2) KinectFusion Modification: Projective TSDF frequently overestimates the true distance to the nearest surface, as the true distance can only be obtained if the ray's angle coincides with the normal of the surface. An ESDF, on the other hand, updates the cells based on the shortest Euclidean distance to the nearest surface. This has the advantage of estimating the true distance better at the cost of steep processing requirements.

Oleynikova et al. [17] showed that using a minimum merging scheme, where the minimum value of all observed TSDF values for each cell is taken, allows the TSDF to converge to an ESDF. This works because as the number of camera viewpoints increases, so does the likelihood of a viewpoint closely aligning with the surface normal, enabling TSDF cells to store a distance that closely approximates the Euclidean distance to the obstacle.

However, they assumed perfect, noise-free sensors. In practice, sensor noise and other disturbances can lead to erroneous observations. Simply taking the minimum of all observations does not allow for these errors to be averaged out, causing them to persist throughout the reconstruction.

In light of this, a weighted merging scheme, which gives more weight to smaller observed distances, is presented. This merging scheme aims to combine the benefits of both the minimum and the original schemes. The merging is



(a) Simulation (b) Average (c) Minimum

(d) Exp. n=1 (e) Exp. n=3 (f) Exp. n=5

Fig. 2: Visualized slices of the TSDF reconstructions under different conditions, all generated from an identical cylinder. The colors indicate the distance values of the respective cell.

equivalent to (1), with the following equation for $w_{\boldsymbol{M}_k}$:

$$w_{\boldsymbol{M}_k} = \exp(n|1 - f_{\boldsymbol{M}_k}|)/\exp(n) \tag{5}$$

where n is an adjustable parameter, and the divisor $\exp(n)$ normalizes the value to $[0, 1]$.

Fig. 2 demonstrates the effect of this weighting. A simple 3D-printed cylinder placed on a table was used as the target object reconstruction, as seen in Fig. 3b. Fig. 2a shows the ideal TSDF generated from a digital twin of the setup in Gazebo. Figs. 2b and 2c show the TSDF from the scan using the original weighting (1) and the minimum merging scheme respectively, while Figs. 2d-2f show the new merging scheme with different values of n.

It can be seen that the minimum merging most closely approximates the ideal TSDF of the simulation, but its sensitivity to noise is readily apparent, indicated by the "spikes" of colors. On the other hand, average merging results in a smoother but more abrupt change of the TSDF values, as indicated by the sharper transition of color at the blue-green border near the center. The proposed methods result in TSDFs that lie between the two, with higher n parameters getting closer to the minimum merging scheme.

3) Extending the TSDF Range: In the original KinFu, due to the memory issue described in Sec. III-A.1, the truncation distance μ is set to 0.03m by default. However, while such a short truncation distance is sufficient for performing 3D reconstruction, for tasks such as obstacle avoidance, bigger truncation distance is necessary.

This is due to the force vector calculation method described in Sec. III-B requiring all the cells within a neighborhood range of size n of the control point to be populated. As such, the methods require the control point's distance from the obstacle surface to be less than $\mu - \beta * \frac{n}{2}$m, where β is the size of the voxel in m, even in ideal cases where all cells in the neighborhood have been observed.

While increasing the truncation distance to the system's maximum limit could address the issue, it leads to significant overestimation of distances for cells far from any surface, particularly when camera viewpoints are limited. Additionally, this approach does not resolve edge cases where some neighboring voxels are unpopulated, potentially causing force vector calculations to fail.

To address these issues, we apply a wavefront algorithm to the neighborhood of the control point when the number of valid neighboring points exceeds a predefined threshold v. This algorithm calculates the Euclidean distance from every observed point to every unobserved point in a brute-force manner. The unobserved points are then set to the closest Euclidean distance plus the projective distance of the selected source cell. This method extends the effective range of force vector calculations and makes them more robust to random unobserved points in the reconstruction. Additionally, since the values of missing cells are derived from observed cells, the problem of overestimation is mitigated.

To further increase the effective range, we also introduce a step parameter s to query every sth neighboring voxel; e.g. with $n = 3$ and $s = 2$, the 1st, 3rd, and 5th voxels are queried. This allows a larger neighborhood to be queried at the cost of losing finer details of the TSDF.

B. Estimating Force Vectors from TSDF

Since each voxel stores the distance to the closest surface, the direction of greatest change of a local neighborhood will be towards the surface. Therefore, the resulting force vector to the closest obstacle, $\boldsymbol{F}$, is calculated as the inverse of the current voxel's TSDF value, $\frac{1}{d}$, and the direction from the gradient $\nabla \boldsymbol{C}(\boldsymbol{p})$ of its local neighborhood, $\boldsymbol{C}(\boldsymbol{p})$, with a nonlinear clamping function Ψ to prevent the force vector's magnitude from reaching infinity as the distance of the control point to the surface approaches zero:

$$\boldsymbol{F} = \Psi\left(\frac{\nabla \boldsymbol{C}(\boldsymbol{p})}{d\,\|\nabla \boldsymbol{C}(\boldsymbol{p})\|}\right) \tag{6}$$

$$\nabla \boldsymbol{C}(\boldsymbol{p}) = \begin{bmatrix} \frac{\delta \boldsymbol{C}(\boldsymbol{p})}{\delta x} & \frac{\delta \boldsymbol{C}(\boldsymbol{p})}{\delta y} & \frac{\delta \boldsymbol{C}(\boldsymbol{p})}{\delta z} \end{bmatrix}^T \tag{7}$$

The gradient of the TSDF $\nabla C(p)$ can be calculated directly by convolving each voxel in the local neighborhood of a control point with a kernel to calculate the central difference. The TSDF is first smoothed using a Gaussian smoothing kernel $G(t)$ before approximating its gradient. This kernel is less sensitive to noise and can be generated depending on the desired size n of the local neighborhood.

Differentiating a convolution operation is equivalent to convolving the TSDF itself with the derivative of the kernel:

$$\nabla \boldsymbol{C} = d(\boldsymbol{C} * G(t)) = \boldsymbol{C} * dG(t) \tag{8}$$

where t is the displacement of the Gaussian from the center, $a * b$ convolves a with b and the Gaussian function and its first derivative with zero mean are given by:

$$G(t) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{t^2}{2\sigma^2}} \tag{9}$$

$$\frac{dG(t)}{dt} = \frac{-t}{\sqrt{2\pi}\sigma^3} e^{-\frac{t^2}{2\sigma^2}} \tag{10}$$

The Gaussian kernel is generated by sampling the Gaussian distribution at discrete intervals. Since the negative values of the exponential in Eq. 9 quickly approach zero,

we truncate the function as we move away from the center of our kernel based on the value of σ and the neighbour size.

Thus, discretizing Eq. 9 and Eq. 10 into m steps around point $\boldsymbol{p}$ gives us 1D discrete Gaussian smoothing and derivative kernels in the x, y and z directions:

$$\boldsymbol{k}_x, \boldsymbol{k}_y, \boldsymbol{k}_z = \left[\eta\left(G(t)|_{t=j}\right),\ \forall\ j{=}[\tfrac{-(m-1)}{2}, \tfrac{m-1}{2}]\right] \tag{11}$$

$$\boldsymbol{dk}_x, \boldsymbol{dk}_y, \boldsymbol{dk}_z = \left[\eta\left(\frac{G(t)}{dt}\bigg|_{t=j}\right),\ \forall\ j{=}[\tfrac{-(m-1)}{2}, \tfrac{m-1}{2}]\right] \tag{12}$$

where η is a normalization term such that for a vector $\boldsymbol{v}$ of size m, $\sum_{i=1}^{m} \eta(\boldsymbol{v}_i) = 1$, $\boldsymbol{v}_x \in \mathbb{R}^{m\times1\times1}$, $\boldsymbol{v}_y \in \mathbb{R}^{1\times m\times1}$ and $\boldsymbol{v}_z \in \mathbb{R}^{1\times1\times m}$.

To calculate the derivative in the x, y and z directions of a discrete, 3D field, we can use the separability property of a Gaussian kernel to generate a 3D kernel for each direction in x, y and z. Each kernel calculates the derivative in the specific direction and performs smoothing in the other two perpendicular directions. We can generate such a kernel using the Kronecker product of the 1D Gaussian smoothing and derivative kernels from Eq. 11 and Eq. 12, respectively:

$$\boldsymbol{K}_x = (\boldsymbol{dk}_x \otimes \boldsymbol{k}_y) \otimes \boldsymbol{k}_z \tag{13}$$

$$\boldsymbol{K}_y = (\boldsymbol{dk}_y \otimes \boldsymbol{k}_x) \otimes \boldsymbol{k}_z \tag{14}$$

$$\boldsymbol{K}_z = (\boldsymbol{dk}_z \otimes \boldsymbol{k}_x) \otimes \boldsymbol{k}_y \tag{15}$$

where $\boldsymbol{K}_{x,y,z} \in \mathbb{R}^{m\times m\times m}$ and $a{\otimes}b$ is the Kronecker product of a and b.

C. Collision Avoidance and Haptic Feedback

A force vector calculated from the above method can be inserted into a controller as a virtual force to enable a collision avoidance behavior on the robot. Here, this is realized by setting a control point at the end effector of a Franka Emika Robot, where the the force vector can be interpreted as an external force $\boldsymbol{F}_{ext}$ acting upon it. Additionally, this value can then be directly used for calculating the haptic feedback during a teleoperation scenario.

By placing multiple control points around the end effector, as in Fig. 3a, and taking the transforms ${}^{r}\boldsymbol{p}_{cp}$ of those control points relative to the root frame r, torque terms $\boldsymbol{T}_{ext}$ can also be derived to obtain the external wrench $\boldsymbol{\mathcal{F}}_{ext} = [\boldsymbol{F}_{ext}, \boldsymbol{T}_{ext}]$:

$$\boldsymbol{\mathcal{F}}_{ext} = \frac{1}{n_{cp}}[\sum_{i=1}^{n_{cp}} \boldsymbol{F}_i, \sum_{i=1}^{n_{cp}} \boldsymbol{T}_i] \in \mathbb{R}^6 \tag{16}$$

$${}^{r}\boldsymbol{F} = \text{abs}({}^{cp}\hat{\boldsymbol{F}})\ {}^{cp}\boldsymbol{F} \in \mathbb{R}^3 \tag{17}$$

$${}^{r}\boldsymbol{T} = {}^{r}\boldsymbol{p}_{cp} \times {}^{cp}\boldsymbol{F} \in \mathbb{R}^3 \tag{18}$$

where $\text{abs}({}^{cp}\hat{\boldsymbol{F}})$ calculates the $\cos\boldsymbol{\theta}$ of the force vector along the x-, y-, and z-axes of the root frame. The magnitude of $\boldsymbol{\mathcal{F}}_{ext}$ is then clamped to a pre-defined maximum value f_{max}.

The calculated wrench term is then fed into a Cartesian impedance controller of the form

$$\boldsymbol{\tau}_a = \boldsymbol{\tau}_{task} + \boldsymbol{\tau}_{\boldsymbol{N}} + \boldsymbol{\tau}_{ff} + \boldsymbol{\tau}_{ext}, \tag{19}$$

$$\boldsymbol{\tau}_{task} = \boldsymbol{J}^T(-\boldsymbol{K}_p\boldsymbol{e} - \boldsymbol{K}_d(\boldsymbol{J}\dot{\boldsymbol{q}})), \tag{20}$$

$$\boldsymbol{\tau}_{ext} = \boldsymbol{J}^T\boldsymbol{\mathcal{F}}_{ext} \tag{21}$$

with $\boldsymbol{J}^T \in \mathbb{R}^{7\times6}$ the transposed Jacobian matrix, $\boldsymbol{K}_{p/d} \in \mathbb{R}^{6\times6}$ the proportional and derivative gains, $\boldsymbol{e} \in \mathbb{R}^6$ the task-space error, $\boldsymbol{\tau}_{task/N/ext} \in \mathbb{R}^7$ the task, nullspace, and external torques respectively, and $\boldsymbol{\tau}_{ff} \in \mathbb{R}^7$ the feed forward term including gravity and Coriolis/centrifugal terms.

IV. Experiments

A. Comparing the Merging Schemes

We conducted a study to compare the quantitative quality of the force vector under different TSDF merging schemes. A single 5cm-by-10cm cylinder was placed on a flat table, and the cylinder was scanned using an Intel Realsense D435 mounted rigidly on a 7DoF Franka Emika Robot, with an identical trajectory for all merging schemes. As ground truth, a digital twin of the setup was created in Gazebo. A single control point was placed at the edge of the gripper's tip (the bottom point in Fig.3a). Data from the force vector algorithm was collected by moving the robot into 4 different positions, as seen in Fig. 3b. The first position was directly above the table to serve as a sanity check, as it should in theory perfectly align with the simulation's force vector, given that it is a flat surface. Afterwards, the robot was brought to 3 different positions. For each one, the cosine similarity between real-life and simulation's force vectors were calculated, and their magnitudes were sampled. For all scenarios, resolution of 512 voxels, $1.2m^3$ volume, $\mu = 0.15m$ and step size $s = 2$ were used.

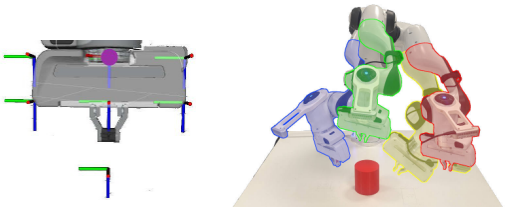
The results of this experiment are summarized in Figs. 3c and 3d. In general, the proposed merging schemes perform better or close to the average merging scheme, with $n = 3$ showing the best performance on average. All merging schemes underestimate the proximity of the obstacle, resulting in a large norm difference. However, this is an issue that can be addressed by scaling the force vector to increase its effect at longer distances.

The minimum merging scheme, while it marginally better estimates the proximity of the object given by its smaller norm difference in Fig. 3d, its sensitivity to noise is apparent in its lower cosine similarity score at position 1. Moreover, the scheme reconstruction failed to capture the geometry of the cylinder, leading the algorithm to believe that the robot arm was placed inside the object at position 4, resulting in the angle of the vector to be grossly different and the magnitude of the vector to be much greater than all others.

B. User Study

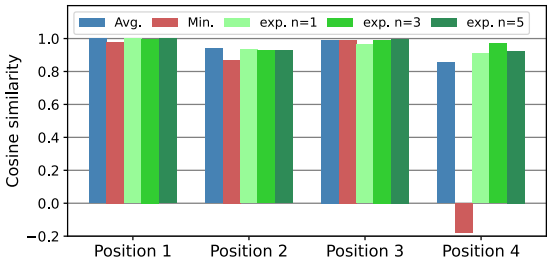
The aim of the user study was to validate the proposed system's usefulness in a teleoperation scenario through haptic feedback. GARMI [19], a robot equipped with two Franka Emika Robots as its arms, was used as the platform. A Realsense D435 camera was mounted rigidly to its right arm, which was used to scan the setup. Based on the quantitative evaluation results, the exp($n = 3$) merging scheme was chosen, with all other parameters being identical.

We recruited 13 non-expert participants with ($n_{\male} = 7$, $n_{\female} = 5$) between 24 and 52 years of age ($\mu = 34.5, \sigma = 7.6$) for the study. The participants were asked to teleoperate the

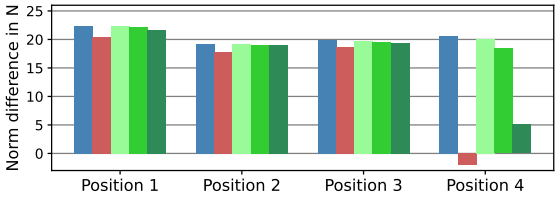


(a) Control points. Root frame in purple.

(b) Sample positions. Color to position: R=1, Y=2, G=3, B=4.



(c) Cosine similarity at each position by merging scheme.



(d) Force norm difference ($||F_{sim}|| - ||F_{scheme}||$) at each position by merging scheme.

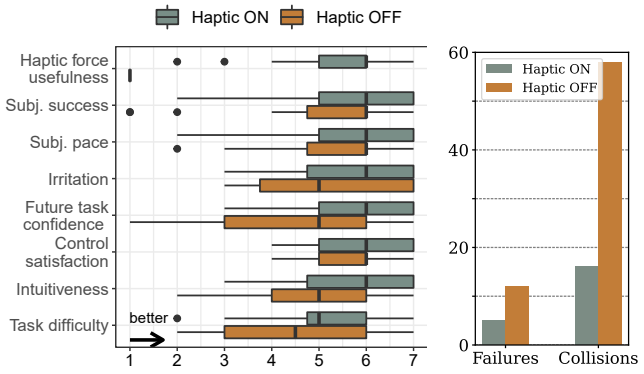
Fig. 3: Comparison of the different merging schemes against the simulation ground truth force vector in sample positions.

(a) (b) (c)

Fig. 4: (a) The initial teleoperation pose. (b)-(c). The torso and right arm-mounted camera's view, respectively. (d) Cross-section of the TSDF of the scene, showing the holes.

left arm of GARMI, holding a small cube, to perform a packing task, dropping the cube into one of the three square openings of a cardboard box. This was randomly chosen by the experimenter for each trial. If the cube failed to enter the correct opening, the trial was regarded as a failure. For all trials, the number of collisions between the arm and the environment was recorded. The study had a 2x2 design: i) the force feedback algorithm was either turned on, or off; and (ii) the participants could view the color image output from either the torso-mounted or the right arm-mounted camera. The second condition was chosen to vary the amount of visibility in the relevant task space. Camera viewpoints and box position were fixed across all sessions. The setup and the camera conditions are shown in Fig. 4.

For teleoperation, Force Dimension's Omega7 was used, which has configurable forces in all 3 axes. Since it does not have configurable torques, orientation control was disabled



(a) (b)

Fig. 5: Results from the study. (a) Boxplot of the NASA TLX surveys. (b) Total number of collisions and failures from the 72 trials for each haptic feedback condition.

for the experiment. The device was used to publish the desired pose of the end effector in Cartesian space of the robot. For haptic feedback, a time-to-impact (TTI) algorithm was implemented:

$$v_d = \frac{\min(\boldsymbol{v} \cdot \boldsymbol{F}, 0)}{||\boldsymbol{f}||} \tag{22}$$

$$\boldsymbol{F}_\Omega = \Phi(k_\Omega v_d \hat{\boldsymbol{v}}) \tag{23}$$

where $\boldsymbol{v}$ is the Omega7's current velocity, $\boldsymbol{F}$ is the force vector, k_Ω is a scalar gain term, Φ is a non-linear clamping function to avoid overloading the device's motors, and $\boldsymbol{F}_\Omega$ is the force value that is set on the Omega7.

After a short introduction, the participants repeated each condition 3 times. The order of the conditions was randomized, resulting in 144 total data points. Number of failures and the number of collisions with the environment were recorded. Task completion time was not tracked on purpose, so as to not place the participants under time pressure. After the experiment, the participants were asked to fill out the NASA TLX questionnaire [20], and an additional question rating the helpfulness of haptic feedback for the conditions where it was enabled. The results are shown in Fig. 5.

TABLE I: ANOVA analysis of the two independent variable conditions and their p-values. $p < 0.05$ is bolded, and borderline values (i.e. p near 0.5 are italicized.

	Cam. View	Haptics
Haptic force usefulness	0.5238	N/A
Subj. success	0.6065	0.2602
Subj. pace	0.0759	0.5470
Irritation	**0.0457**	0.0847
Future task confidence	0.2952	**0.0028**
Control satisfaction	0.7272	0.4864
Intuitiveness	0.0804	0.0804
Task difficulty	0.6204	*0.0528*
Collisions	0.9154	**0.0003**
Failures	0.5411	**0.0283**

Figs. 5a and 5b show the result of the 48 NASA TLX questionnaires from the participants summarized based on the haptic feedback condition, and the number of failures and collisions in the trials, respectively.

Given the small number of participants and the broad range of the qualitative results, an ANOVA analysis was

performed on both the quantitative and qualitative results to examine the significance of the two independent variables. The results are summarized in Table I. Based on this, the only statistically significant items are *Irritation*, *Future task confidence*, and to some extent, *Task difficulty*. Camera viewpoint did not have a significant impact on the way that the participants perceived the experiment on average. On the other hand, it can be implied that the haptic feedback provides helpful information on completing the tasks, though a a larger sample size is needed for a more clear evidence.

The framework's benefit is more evident in Fig. 5b. Out of 72 trials for each haptic feedback condition, there were only 5 failures with haptic feedback, compared to 16 without. Additionally, far more collisions occurred without haptic feedback. This is supported by the p-values in Table I, showing that haptic feedback significantly improves task performance, indicating it provides valuable obstacle avoidance information to users.

V. Conclusion

In this paper, we presented a unified system for calculating force vectors of obstacles from a TSDF-based 3D reconstruction of the environment. The proposed system uses KinFu with a new weighted merging scheme and a wavefront algorithm to extend its effective range, and can calculate force vectors in real time with multiple control points. Quantitative assessment was carried out to examine the quality of the force vectors under the different merging schemes. Moreover, a full haptic teleoperation pipeline was implemented with GARMI and Omega7, and a user study was conducted to validate the usefulness of the system in a robot teleoperation scenario. Statistical analysis on both NASA TLX and quantitative results from the experiment showed that the haptic feedback feature had a significant effect on improving the users' performance in the experiment.

There are several limitations to the system. First, the *KinFu* requires quality camera viewpoints to achieve a good TSDF for accurate force vector calculation, which may be impossible to achieve in certain scenarios. Second, TSDF-based representation is an overly expensive method to represent simple shapes in the environment such as planes and cuboids, which are prevalent in man-made environments.

In future works, we plan to investigate how to deal with limited viewpoints for TSDF reconstruction by using deep learning-based SDF estimation methods that do not require full camera observations. Additionally, we wish to examine the possibility of creating an efficient and unified collision avoidance system by combining geometric primitives-based collision avoidance methods like [6] with dynamic SDFs.

References

[1] L. S. Scimmi, M. Melchiorre, M. Troise, S. Mauro, and S. Pastorelli, "A practical and effective layout for a safe human-robot collaborative assembly task," *Applied Sciences*, vol. 11, no. 4, 2021. [Online]. Available: https://www.mdpi.com/2076-3417/11/4/1763

[2] H. Liu, D. Qu, F. Xu, Z. Du, K. Jia, J. Song, and M. Liu, "Real-time and efficient collision avoidance planning approach for safe human-robot interaction," *Journal of Intelligent & Robotic Systems*, vol. 105, no. 4, p. 93, Aug 2022. [Online]. Available: https://doi.org/10.1007/s10846-022-01687-0

[3] H. Nascimento, M. Mujica, and M. Benoussaad, "Collision avoidance interaction between human and a hidden robot based on kinect and robot data fusion," *IEEE Robotics and Automation Letters*, vol. 6, no. 1, pp. 88–94, 2021.

[4] J. Pankert and M. Hutter, "Perceptive model predictive control for continuous mobile manipulation," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6177–6184, 2020.

[5] M. Gaertner, M. Bjelonic, F. Farshidian, and M. Hutter, "Collision-free MPC for legged robots in static and dynamic scenes," *CoRR*, vol. abs/2103.13987, 2021. [Online]. Available: https://arxiv.org/abs/2103.13987

[6] M. Eckhoff, D. Knobbe, H. Zwirnmann, A. Swikir, and S. Haddadin, "Towards connecting control to perception: High-performance whole-body collision avoidance using control-compatible obstacles," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, Oct. 2023. [Online]. Available: http://dx.doi.org/10.1109/IROS55552.2023.10342515

[7] H. Oleynikova, Z. Taylor, M. Fehr, R. Siegwart, and J. Nieto, "Voxblox: Incremental 3D Euclidean Signed Distance Fields for on-board MAV planning," in *IEEE/RSJ IROS*, Sep. 2017, pp. 1366–1373.

[8] L. Han, F. Gao, B. Zhou, and S. Shen, "FIESTA: Fast Incremental Euclidean Distance Fields for Online Motion Planning of Aerial Robots," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 4423–4430.

[9] Y. Pan, Y. Kompis, L. Bartolomei, R. Mascaro, C. Stachniss, and M. Chli, "Voxfield: Non-projective signed distance fields for online planning and 3d reconstruction," in *Proceedings of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2022.

[10] J. Singh, A. R. Srinivasan, G. Neumann, and A. Kucukyilmaz, "Haptic-guided teleoperation of a 7-dof collaborative robot arm with an identical twin master," *IEEE Transactions on Haptics*, vol. 13, no. 1, pp. 246–252, 2020.

[11] M. S. Mühlbauer, F. Stulp, A. O. Albu-Schäffer, and J. Silvério, "Mixture of experts on riemannian manifolds for visual-servoing fixtures," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems, Workshop on Probabilistic Robotics in the Age of Deep Learning*, October 2022. [Online]. Available: https://elib.dlr.de/189970/

[12] J. Han, K. Cho, I. Jang, C. Ju, H. Il Son, and G.-H. Yang, "Development of a shared controller for obstacle avoidance in a teleoperation system," *International Journal of Control, Automation and Systems*, vol. 18, no. 11, pp. 2974–2982, Nov 2020. [Online]. Available: https://doi.org/10.1007/s12555-019-0410-0

[13] F. Rydén, S. Nia Kosari, and H. J. Chizeck, "Proxy method for fast haptic rendering from time varying point clouds," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011, pp. 2614–2619.

[14] F. Rydén and H. J. Chizeck, "A proxy method for real-time 3-dof haptic rendering of streaming point cloud data," *IEEE Transactions on Haptics*, vol. 6, no. 3, pp. 257–267, 2013.

[15] D. Ni, A. W. W. Yew, S. K. Ong, and A. Y. C. Nee, "Haptic and visual augmented reality interface for programming welding robots," *Advances in Manufacturing*, vol. 5, no. 3, pp. 191–198, Sep 2017. [Online]. Available: https://doi.org/10.1007/s40436-017-0184-7

[16] D. Valenzuela-Urrutia, R. Muñoz-Riffo, and J. Ruiz-del Solar, "Virtual reality-based time-delayed haptic teleoperation using point cloud data," *Journal of Intelligent & Robotic Systems*, vol. 96, no. 3, pp. 387–400, Dec 2019. [Online]. Available: https://doi.org/10.1007/s10846-019-00988-1

[17] H. Oleynikova, A. Millane, Z. Taylor, E. Galceran, J. Nieto, and R. Siegwart, "Signed Distance Fields: A Natural Representation for Both Mapping and Planning," in *RSS Workshop on Geometry and Beyond*, 2016.

[18] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohli, J. Shotton, S. Hodges, and A. Fitzgibbon, "KinectFusion: Real-time Dense Surface Mapping and Tracking," in *10^{th} IEEE ISMAR*, 2011, pp. 127–136.

[19] M. Tröbinger, C. Jähne, Z. Qu, J. Elsner, A. Reindl, D. Wahrmann, S. Parusel, S. Haddadin, and S. Haddadin, "Introducing GARMI - a Service Robotics Platform to Support the Elderly at Home: Design Philosophy , System Overview and First Results," *RA-L*, vol. 3766, no. c, 2021.

[20] S. G. Hart and L. E. Staveland, "Development of nasa-tlx (task load index): Results of empirical and theoretical research," *Human mental workload*, vol. 1, no. 3, pp. 139–183, 1988.