Interactive Distance Field Mapping and Planning to Enable Human-Robot Collaboration

Usama Ali[1*], Lan Wu[2*], Adrian Müller[1], Fouad Sukkar[2], Tobias Kaupp[1], Teresa Vidal-Calleja[2]

***Abstract*—Robot manipulation in human-robot collaborative applications require scene representations that are kept up-to-date and facilitate safe motions in dynamic scenes. We present an interactive distance field mapping and planning (IDMP) framework that handles dynamic objects and collision avoidance through an efficient representation. We define *interactive* mapping and planning as the process of creating and updating the representation of the scene online while simultaneously planning and adapting the robot's actions based on that representation. Given depth sensor data, our framework builds a continuous field that allows to query the distance and gradient to the closest obstacle at any required position. The key aspect of this work is an efficient Gaussian Process field that performs incremental updates and implicitly handles dynamic objects with a simple and elegant formulation based on a temporary latent model. Accompanying video, code, and datasets are made publicly available[3].**

***Index Terms*—Interactive Mapping and Planning, Euclidean Distance Fields, Gaussian Process, Human-Robot Collaboration.**

I. Introduction

Robot manipulation in human-robot collaboration (HRC) and other applications of robots in the field call for interactive representations to deal with dynamic and evolving scenes. For true collaboration in industrial settings, humans and robots physically share the same space, e.g. working jointly and simultaneously on the assembly of a product. Euclidean Distance Field (EDF) representations [1], [2], [3], [4], [5] that aim to fulfil some of the above-mentioned requirements have been proposed in the robotics literature. These representations have the ability to dynamically update the changes in the scene via the so-called free space carving method. This type of method performs expensive ray-casting and progressive integration of the Truncated Signed Distance Field (TSDF) to update the map, resulting in free space that is only gradually cleared when objects move in the scene.

This paper presents an interactive distance field mapping and planning (IDMP) framework aimed at dynamic scenes common in human-robot collaboration scenarios (see Fig. 1 as an example). We propose to build and maintain an up-to-date distance and gradient field using a Gaussian-Process-based method capable of integrating depth sensor measurements following our previous work in [6], [7]. Here, we present a simple and elegant method for dynamic updates and fusion of our Gaussian Process (GP) distance field by querying distances and gradients from a temporary latent GP distance field generated with only the points of the current frame. We name this latent representation the "Frustum Field". The Frustum Field is able to deal with the discrepancy in the distance between surface points that move in the current frame without the need for ray-tracing. It also allows coherent fusion with a previously mapped distance and gradient field. The fused representation is continuous allowing querying of surface normals and the Euclidean distance to the nearest surface and its gradient at an arbitrary spatial resolution. These properties, through a single unified representation, facilitate various downstream robot manipulation tasks, such as 6D object pose estimation, motion planning and grasp pose generation [8]. We experimentally evaluate IDMP's performance in both static and dynamic scenes and benchmark it against state-of-the-art algorithms. The key contributions of our proposed framework are:

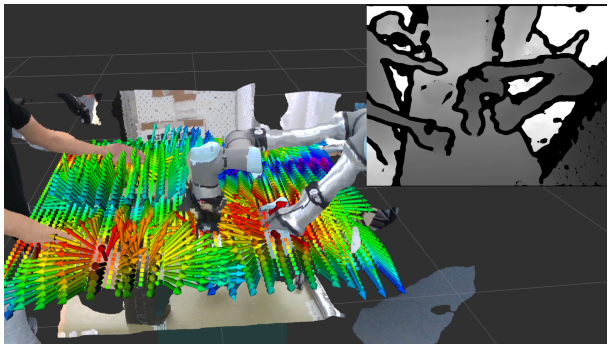


Fig. 1: Interactive generation of a distance and gradient field in an HRC setting. Coloured point cloud map of the scene and one horizontal slice of the field.

- A novel dynamic update and fusion method for GP-based distance and gradient fields. This method is based on a latent representation named Frustum Field that enables dealing with static, moving, and new points.
- An interactive distance field mapping and planning (IDMP) framework that generates a continuous Euclidean distance and gradient field online, and is integrated with a gradient-based planner for 3D collision-free navigation.
- An efficient open-sourced ROS-based implementation of the full IDMP framework.

II. Proposed Framework

A. Gaussian Process Distance Field

We use the so-called reverting GP distance field originally presented in [7] to build a distance and gradient field. Consider a surface $\mathcal{S}$ in a Euclidean space $\mathbb{R}^D$, and a set of discrete observations of S as $\mathbf{y} = \{y_j\}_{j=1}^{J} \in \mathbb{R}$ taken at locations

This work was supported by the Industrial Transformation Training Centre (ITTC) for Collaborative Robotics in Advanced Manufacturing (also known as the Australian Cobotics Centre) funded by ARC (Project ID: IC200100001) and the Bavarian Research Foundation (grant AZ-1512-21).

[1]Authors are with the Center for Robotics (CERI) at the Technical University of Applied Sciences Würzburg-Schweinfurt (THWS), Germany.

[2]Authors are with the Robotics Institute, Faculty of Engineering and IT, University of Technology Sydney (UTS), Australia.

[*]These authors are co-first authors and contributed equally to this work. Corresponding author: Lan.Wu-2@uts.edu.au

[3]https://uts-ri.github.io/IDMP

$\mathbf{X} = \{\mathbf{x}_j\}_{j=1}^{J} \in \mathbb{R}^D$. By modelling the occupancy $o(\mathbf{x}) : \mathbb{R}^D \mapsto \mathbb{R}$ of the space with a GP as $o \sim \mathcal{GP}(0, k(\mathbf{x}, \mathbf{x}'))$, it is possible to infer the surface occupancy $\hat{o}(\mathbf{x}_*)$ at any location in the space. We arbitrarily define the occupied area to be equal to 1. Therefore, $\mathbf{y}$ is equal to $\mathbf{1}$ for the GP inference:

$$\hat{o}(\mathbf{x}_*) = \mathbf{k}_{\mathbf{x}_*\mathbf{X}} \left(\mathbf{K}_{\mathbf{XX}} + \sigma_o^2 \mathbf{I}\right)^{-1} \mathbf{1}. \tag{1}$$

The distance field $\hat{d}(\mathbf{x}_*)$ given any location $\mathbf{x}_*$ is obtained by applying a *reverting* function r to the occupancy field as

$$\hat{d}(\mathbf{x}_*) = r\left(\hat{o}\left(\mathbf{x}_*\right)\right). \tag{2}$$

Considering the square exponential kernel $k(\mathbf{x}, \mathbf{x}') = \sigma^2 \exp\left(-\frac{\|\mathbf{x}-\mathbf{x}'\|^2}{2l^2}\right)$, substituting the reverting function of square exponential kernel into Eq. 2 gives us:

$$\hat{d}(\mathbf{x}_*) = \sqrt{-2l^2 \log\left(\frac{\hat{o}\left(\mathbf{x}_*\right)}{\sigma^2}\right)}. \tag{3}$$

We apply the linear operator of GP [9] to infer the gradient along with the distance field without the normal as input. Applying the linear operator to Eq. 1:

$$\nabla\hat{o}\left(\mathbf{x}_*\right) = \nabla\mathbf{k}_{\mathbf{x}_*\mathbf{X}} \left(\mathbf{K}_{\mathbf{XX}} + \sigma_o^2 \mathbf{I}\right)^{-1} \mathbf{1}, \tag{4}$$

Computing the gradient of Eq. 3 with respect to the distance shows that the gradient of $\hat{d}(\mathbf{x}_*)$ aligns directionally with the gradient of $\hat{o}(\mathbf{x}_*)$, albeit subject to a scaling factor. This implies that $\nabla\hat{d}(\mathbf{x}_*) \approx \nabla\hat{o}(\mathbf{x}_*)$ in term of direction.

B. Framework Overview

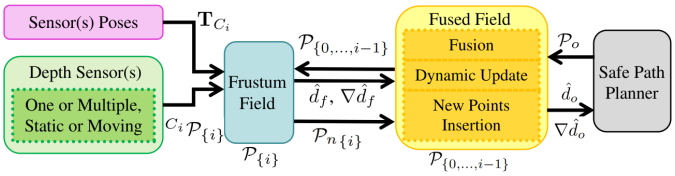


Fig. 2: System diagram of IDMP.

Given depth data in the form of pointclouds $\mathcal{P}_{\{i\}}$ and sensor poses $\mathbf{T}_{C_i}$, we first model the temporary latent Frustum Field (blue) with a GP using only $\mathcal{P}_{\{i\}}$ as training points. All prior training points $\mathcal{P}_{\{0,\dots,i-1\}}$ in the Fused Field (yellow) (another GP) are then passed to the Frustum Field. Given the sensor pose, the Frustum Field selects from $\mathcal{P}_{\{0,\dots,i-1\}}$ the points that are within the frustum area $\mathcal{P}_{f\{0,\dots,i-1\}}$ and returns the inferred values $\hat{d}_f$ and $\nabla\hat{d}_f$ to the Fused Field. These distances and gradients are used to perform fusion and dynamic update by updating the training points that model the Fused Field. The path planner then queries $\hat{d}_o$ and $\nabla\hat{d}_o$ at the locations $\mathcal{P}_o$ to adapt its motion plans in response to a changing map.

C. Frustum Field

We use $\mathcal{P}_{\{i\}}$ as a set of observations to model the Frustum Field via Eq. 1 and apply the reverting function to obtain the distance field. The Frustum Field is then inferred via Eq. 3. By using the linear operator of GP for gradient inference, we employ Eq. 4 to infer the gradients along with the distance. After the Frustum Field is modelled, we query the distance and gradient from the points $\mathcal{P}_{f\{0,\dots,i-1\}}$ to pass them to the Fused Field for the interactive updates.

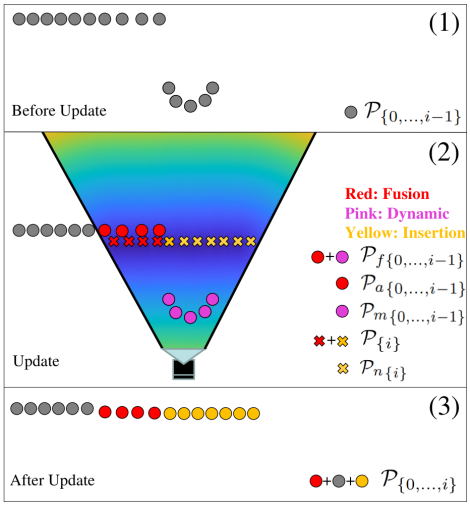


Fig. 3: A simplified illustration of the proposed method.

D. Fused Field

Before the update, the Fused Field is modelled based on $\mathcal{P}_{\{0,\dots,i-1\}}$. After the fusion and dynamic update it is modelled using $\mathcal{P}_{\{0,\dots,i\}}$. There are three key processes performed at every frame: adjusting points for fusion, removing points for dynamic updates, and inserting points for measurements in new areas. Fig. 3 illustrates the way points are selected given the Frustum Field. The top figure shows the points $\mathcal{P}_{\{0,\dots,i-1\}}$ before the update. The middle figure shows the Frustum Field and the points involved in the three processes. Red and yellow crosses are the new measurements $\mathcal{P}_{\{i\}}$. The coloured background is the Frustum Field modelled using $\mathcal{P}_{\{i\}}$. Circles coloured in red and pink are the prior training points of the Fused Field located in the frustum, denoted as $\mathcal{P}_{f\{0,\dots,i-1\}}$. Note that we use the sensor pose to select $\mathcal{P}_{f\{0,\dots,i-1\}} \subset \mathcal{P}_{\{0,\dots,i-1\}}$. We then query the Frustum Field to infer $\hat{d}_f$ and gradient $\nabla\hat{d}_f$ to perform the fusion and dynamic update. An advantage of this approach is that we only query the Frustum Field once to perform the three further processes described as follows.

1) ***Fusion:*** Let us denote the prior training points within the frustum as $\mathcal{P}_{a\{0,\dots,i-1\}} \subset \mathcal{P}_{f\{0,\dots,i-1\}}$ (red circles) and corresponding distances and gradients as $\hat{d}_a$, $\nabla\hat{d}_a$. A threshold η on the distance value is then used to indicate if $\mathcal{P}_{a\{0,\dots,i-1\}}$ are relatively close to the surface given the current Frustum Field. For each point $\mathbf{p}_{a\{0,\dots,i-1\}}$ in $\mathcal{P}_{a\{0,\dots,i-1\}}$, the fusion is performed by updating its position through the distance to the surface $\hat{d}_a$ in the direction of the gradient $\nabla\hat{d}_a$ as:

$$\hat{\mathbf{p}}_a = \mathbf{p}_a - \hat{d}_a \nabla\hat{d}_a. \tag{5}$$

Note that $\hat{d}_a$ and $\nabla\hat{d}_a$ are the result of simply querying the Frustum Field. No data association nor iteration is required as GP distance inference produces directly the distance to the surface and it is accurate through our reverting GP model. After the fusion, $\mathcal{P}_{a\{0,\dots,i-1\}}$ is replaced by $\hat{\mathcal{P}}_{a\{0,\dots,i-1\}}$ as the training point of the Fused Field. Note that since we have $\hat{\mathcal{P}}_{a\{0,\dots,i-1\}}$, we do not need to include new points in the overlapping area (red crosses) into the Fused Field.

2) ***Dynamic Update:*** Let us denote the prior training points to-be-removed as $\mathcal{P}_{m\{0,\dots,i-1\}} \subset \mathcal{P}_{f\{0,\dots,i-1\}}$ (pink circles). Note that in this case and without loss of generality, capturing

measurements behind $\mathcal{P}_{f\{0,\dots,i-1\}}$ implies that the points $\mathcal{P}_{f\{0,\dots,i-1\}}$ have moved and are not in the scene anymore. The Frustum Field distance values $\hat{d}_m$ for these points are bigger than η as they are relatively far away from the current captured measurements, which means the object has moved. We then eliminate from the training set the removed points $\mathcal{P}_{m\{0,\dots,i-1\}}$ of the Fused Field.

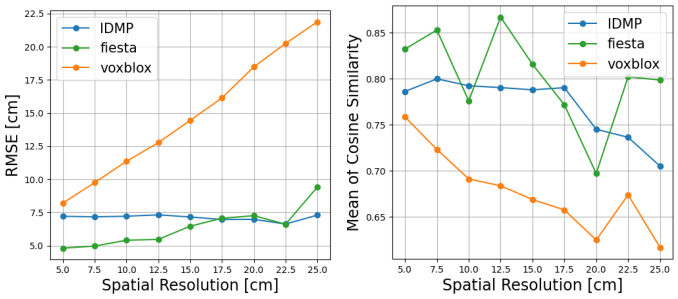
*3) **New Points Insertion:*** Let us denote the points in $\mathcal{P}_{\{i\}}$ that are outside of the overlapping area as $\mathcal{P}_{n\{i\}}$ (yellow crosses). These points are directly added to the Fused Field training set. Points outside the overlapping area in the Fused Field remain the same (grey circles).

III. Evaluation

We evaluate and benchmark the proposed mapping approach in static and dynamic scenes. We also show the planning performance in the presence of a dynamic object.

A. Mapping Results

1) Static Scene: Fig. 1 presents qualitative results where IDMP queries are visualised. For every point in space, a distance (colour) and direction (arrow) away from the closest obstacle can be computed online and on-demand. The queries are taken using a regular grid on a plane parallel to the table. Note that because of the continuous nature of IDMP, the query points can be located anywhere in space and are neither constrained to be on a plane or a regular grid nor bound to a specific spatial resolution.

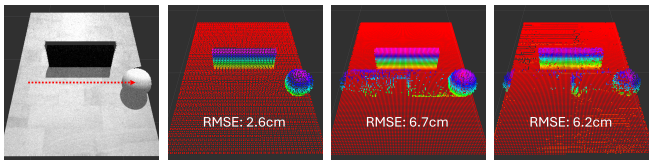


(a) Distances (b) Gradients

Fig. 4: Evaluation using the Cow & Lady dataset: (a) RMSE for distances; (b) cosine similarity for gradients (1.0 means perfect alignment with ground truth).

To compare IDMP's distance field accuracy to Voxblox [1] and FIESTA [2], we compute the RMSE for each framework while varying the *spatial resolution*. For Voxblox/FIESTA, this corresponds to the voxel size. For IDMP we query at a regular grid that corresponds to the centre of each voxel. Figure 4a shows our experimental results. The accuracy for Voxblox and FIESTA decreases with increasing spatial resolution. In contrast, IDMP's accuracy remains constant and there is no requirement to pre-set a certain resolution illustrating the advantage of using a continuous representation. The evaluation of gradients is presented in Fig. 4b. This figure shows the mean of the cosine similarities between the computed gradients and the ground truth. Both IDMP and FIESTA perform better than Voxblox for all voxel sizes and resolutions. FIESTA shows slightly better performance than IDMP in some cases; however, at the cost of higher variability. In terms of computation time for IDMP, updating the GP takes on average $250ms$ per frame for this scene while the query step takes $2\mu s$ per point.

2) Dynamic Scene: We evaluate how each framework handles dynamic scenes with moving objects. Fig. 5a shows a Gazebo scene with a ball rolling on the table from left to right. The last three subfigures show the surface points at the end of the run for the three frameworks. Both FIESTA and Voxblox show artefacts due to the progressive weighting and integration while IDMP does not. IDMP results in the best RMSE of $2.6cm$ which is more than a factor of 2 better than FIESTA/Voxblox. Updating the GP for a scene of this size takes on average $50ms$ per frame while the query step takes $2\mu s$ per point. Only the time for processing the frame is dependent on the size of the scene whereas the time needed for the query remains constant.

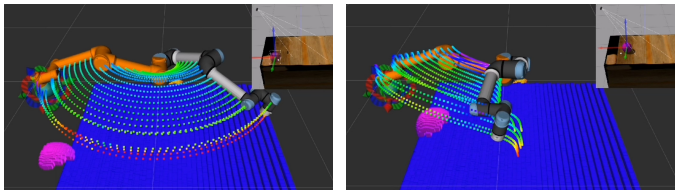


(a) Gazebo (b) IDMP (c) FIESTA (d) Voxblox

Fig. 5: Evaluation of a dynamic scene: (a) setup in simulation; (b) to (d) show the fused surface points at the end of the run. Colours indicate the height of the points.

B. Motion Planning in Dynamic Scenes

In this experiment, we demonstrate that our method is able to facilitate safe motion planning of a robot arm in the presence of dynamic obstacles which is useful for close-proximity human-robot collaboration tasks, see Fig. 6. Here, we utilise the Covariant Hamiltonian optimisation-based motion planner (CHOMP) [10], a gradient-based trajectory optimisation method that iteratively perturbs a given initial trajectory away from obstacles. Due to a) the ability of our framework to compute updated gradients analytically and b) the need to only query the points $\mathcal{P}_o$ the motion planner requires, we can continuously replan online. The computation time for the query is approximately $6.4ms$ (3190 points at $2\mu s$/point).



(a) Original planned trajectory (no obstacle encountered). (b) Modified trajectory due to dynamic obstacle.

Fig. 6: Re-planning in the presence of a dynamic obstacle. CHOMP approximates the manipulator via 29 spheres.

IV. Conclusion

In this paper, we presented an efficient interactive mapping and planning framework named IDMP. Going forward, we aim to further explore IDMP's capabilities, such as multi-resolution sampling and mapping uncertainty, for enhancing motion planning, particularly for manipulation tasks in HRC applications.

V. Acknowledgement

The authors would like to thank Cedric Le Gentil for useful discussions and code collaboration.

REFERENCES

[1] H. Oleynikova, Z. Taylor, M. Fehr, R. Siegwart, and J. Nieto, "Voxblox: Incremental 3D Euclidean Signed Distance Fields for on-board MAV planning," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Vancouver, BC: IEEE, Sep. 2017, pp. 1366–1373. [Online]. Available: http://ieeexplore.ieee.org/document/8202315/

[2] L. Han, F. Gao, B. Zhou, and S. Shen, "FIESTA: Fast Incremental Euclidean Distance Fields for Online Motion Planning of Aerial Robots," 2019.

[3] D. Zhu, C. Wang, W. Wang, R. Garg, S. Scherer, and M. Q.-H. Meng, "VDB-EDT: An Efficient Euclidean Distance Transform Algorithm Based on VDB Data Structure," May 2021. [Online]. Available: https://arxiv.org/abs/2105.04419v1

[4] 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 *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Kyoto, Japan: IEEE, Oct. 2022, pp. 5331–5338. [Online]. Available: https://ieeexplore.ieee.org/document/9981318/

[5] Y. Bai, Z. Miao, X. Wang, Y. Liu, H. Wang, and Y. Wang, "Vdbblox: Accurate and efficient distance fields for path planning and mesh reconstruction," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2023, pp. 7187–7194.

[6] L. Wu, K. M. B. Lee, L. Liu, and T. Vidal-Calleja, "Faithful Euclidean Distance Field from Log-Gaussian Process Implicit Surfaces," Jan. 2021, arXiv:2010.11487 [cs]. [Online]. Available: http://arxiv.org/abs/2010.11487

[7] C. L. Gentil, O.-L. Ouabi, L. Wu, C. Pradalier, and T. Vidal-Calleja, "Accurate Gaussian Process-based Distance Fields with applications to Echolocation and Mapping," Sep. 2023, arXiv:2302.13005 [cs] version: 2. [Online]. Available: http://arxiv.org/abs/2302.13005

[8] R. Newbury, M. Gu, L. Chumbley, A. Mousavian, C. Eppner, J. Leitner, J. Bohg, A. Morales, T. Asfour, D. Kragic *et al.*, "Deep learning approaches to grasp synthesis: A review," *IEEE Transactions on Robotics*, 2023.

[9] S. Särkkä, "Linear operators and stochastic partial differential equations in gaussian process regression," in *Artificial Neural Networks and Machine Learning–ICANN 2011: 21st International Conference on Artificial Neural Networks, Espoo, Finland, June 14-17, 2011, Proceedings, Part II 21*. Springer, 2011, pp. 151–158.

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