Computational Visual Media
https://doi.org/10.1007/s41095-0xx-xxxx-x

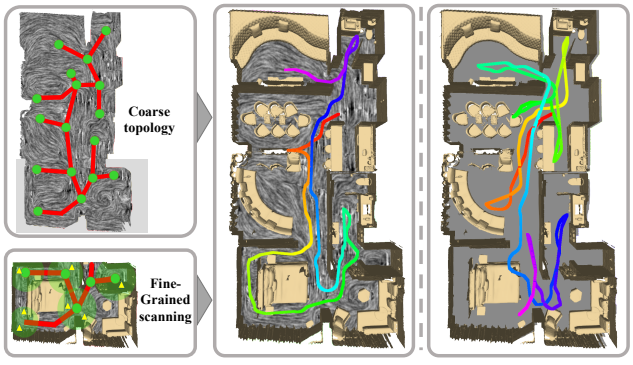
Research/Review Article

THP: Tensor-Field-Driven Hierarchical Path Planning for Autonomous Scene Exploration with Depth Sensors

Yuefeng Xi[1], Chenyang Zhu[1](✉), Yao Duan[1], Renjiao Yi[1], Lintao Zheng[1], Hongjun He[1],and Kai Xu[1](✉)

Abstract It is challenging to automatically explore an unknown 3D environment with a robot only equipped with depth sensors due to the limited field of view. We introduce *THP*, a tensor field-based framework for efficient environment exploration which can better utilize the encoded depth information through the geometric characteristics of tensor fields. Specifically, a corresponding tensor field is constructed incrementally and guides the robot to formulate optimal global exploration paths and a collision-free local movement strategy. Degenerate points generated during the exploration are adopted as anchors to formulate a hierarchical TSP for global path optimization. This novel strategy can help the robot avoid long-distance round trips more effectively while maintaining scanning completeness. Furthermore, the tensor field also enables a local movement strategy to avoid collision based on particle advection. As a result, the framework can eliminate massive, time-consuming recalculations of local movement paths. We have experimentally evaluate our method with a ground robot in 8 complex indoor scenes. Our method can on average achieve 14% better exploration efficiency and 21% better exploration completeness than state-of-the-art alternatives using LiDAR scans. Moreover, compared to similar methods, our method makes path decisions 39% faster due to our hierarchical exploration strategy.

Keywords Tensor field, indoor scene exploration, path planning, trajectory optimization



(a) Our hierarchical exploration driven by tensor field (b) Path w/o tensor field

Fig. 1 THP plans trajectories (a) based on hierarchical exploration strategies driven by tensor fields. Based on these fields, our method provides a coarse topology matching the structure of the scene for the robot to navigate, and groups for robot scanning at a fine-grained level. With similar completeness of scanned scene, the path in (a) is 52.734 m but 105.282 m in (b), doubling the efficiency of exploration.

1 Introduction

With the increasing demand for intelligent robot applications such as vacuum cleaner robots and autonomous navigation, how to thoroughly explore an unknown 3D environment efficiently has emerged as a core problem in modern robotics [1]. Finding an optimal exploration path while the detected environment is dynamically updated is challenging. This problem becomes critical when the target environment is large and complex. The main challenge is how to balance exploration completeness and speed given a limited vision of the whole picture.

Commodity depth sensors such as RealSense and Kinect are widely available for robots, greatly enhancing perception on a low budget [2]. However, most previous works [3–6] still rely on expensive LiDAR devices for exploration and mapping. LiDAR sensors can provide a 360-degree field of view (FOV) for orientation-independent scanning, which enables these frontier-based methods to focus more on scanning completeness, but not efficiency. Consequently, adopt-

1 College of Computing, National University of Defense Technology, Changsha 410073, China. E-mail: Y,Xi, 295482951@qq.com; C.Zhu, zhuchenyang07@nudt.edu.cn; Yao Duan, duanyao16@nudt.edu.cn; R.Yi, yirenjiao@nudt.edu.cn; L.Zheng, zhenglintao13@nudt.edu.cn; H.He, hhj_hi@nudt.edu.cn; K.Xu, kevin.kai.xu@gmail.com.

Manuscript received: 2022-01-01; accepted: 2022-01-01

ing these methods directly for a robot with depth sensors would usually fail due to the limited FOV and additional computation introduced by the changing orientation.

A suitable representation of the progressively reconstructed scene is the key to efficient and complete exploration. Xu et al. [7] introduce the tensor field method for smooth scene reconstruction. A time-varying tensor field can be updated in real-time and used to directly guide the robot's movement. While the scanning FOV is limited, [7] demonstrates that the topological structure of tensor fields is adequate for efficient global path routing within a partially reconstructed scene. However, this method is ineffective for large and complex scenes. In such cases, exploration is severely limited by the initially extracted topology without global optimization as shown in Figure 11.

We introduce *THP*, a hierarchical tensor field-based robot exploration framework, which takes advantage of both frontier-based and tensor-field-driven methods. To further release the power of the tensor field for global optimization, we adopt sparse degenerate points as anchors to better utilize the partial topological structure for global path optimization. Specifically, we have found that the degenerate points are usually generated at joint points of the structure, such as entrances and porches. Since these anchors are generated dynamically during the exploration, we denote them as extended nodes in a travelling salesman problem (TSP) to determine an optimal tour to form the coarse topology (Figure 1(left)), which can help the robot avoid long-distance round trips during exploration. Furthermore, a frontier grouping-based scanning strategy is proposed for fine-grained level exploration. We dynamically group the frontiers around the anchors, and a local fine-level exploration path (Figure 1 right) is generated for complete scanning. Thus, a hierarchical exploration strategy is introduced to ensure scanning completeness while maintaining exploration efficiency. We have evaluated our method with a ground robot using 8 complex indoor scenes. Our method achieves on average 14% better exploration efficiency and 21% better exploration completeness than state-of-the-art LiDAR-based methods. Moreover, compared to similar methods, our method is 39% faster at path decisions (locating the next exploration goal point) with the help of our hierarchical exploration strategy.

Furthermore, the tensor field also enables a movement strategy which avoids collisions, via a particle advection approach. The A* algorithm is the most commonly used approach to find the shortest collision-free path to the next exploration goal point. However, it is time-consuming if the environment is large. Previous works [8] based on LiDAR scans adopt techniques such as offline search strategies to find local movement paths, which are highly efficient. However, it is highly dependent on the complete FOV provided by LiDAR. In contrast, our framework can eliminate massive, time-consuming recalculations with the tensor field, which can be updated in real-time and has no requirement for wide FOV. As a result, our tensor field-based movement strategy with depth scans can perform as well as or even better than state-of-the-art alternatives using LiDAR scans.

In summary, the contributions of this paper include:

- Adoption of degenerate points in the tensor fields as anchors to formulate tour optimization as an online TSP for coarse topology generation, which significantly increases the effectiveness of the tensor-field-driven method in complex environments.
- A frontier grouping-based scanning strategy for fine-grain exploration. This dynamic grouping strategy, which depends on the degenerate points, can ensure scanning completeness while maintaining exploration efficiency.
- A tensor field-based hierarchical exploration framework for an autonomous robot using only depth sensors, which can significantly outperform even state-of-the-art methods using LiDAR.

2 Related work

2.1 Field-guided path planning

Adopting vector fields to efficiently guide robot navigation has been practised for decades [9]. However, their crowded singularities usually trap the robot in local minimum or generate discontinuous robot motions. Vector field histograms (VFHs) [10] were proposed to solve such problems. VFHs can drive the robot to smoothly move and quickly respond in an environment arranged with dense obstacles, and the robot does not need to stop for the next plan. The main drawback of this representation lack of efficiency for global guidance. The gradient field method [11] combines potential fields with a global guiding structure to perform global path planning. In addition, it also incorporates frontier-based and local vector field-based strategies to generate shorter exploration trajectories. However, it suffers from a very high computational cost. Tensor fields were introduced to guide the robot to autonomously reconstruct the scene and smooth the routing path in time-varying environments [7], but such methods fail in scenes with complex structures as no global optimization is performed.

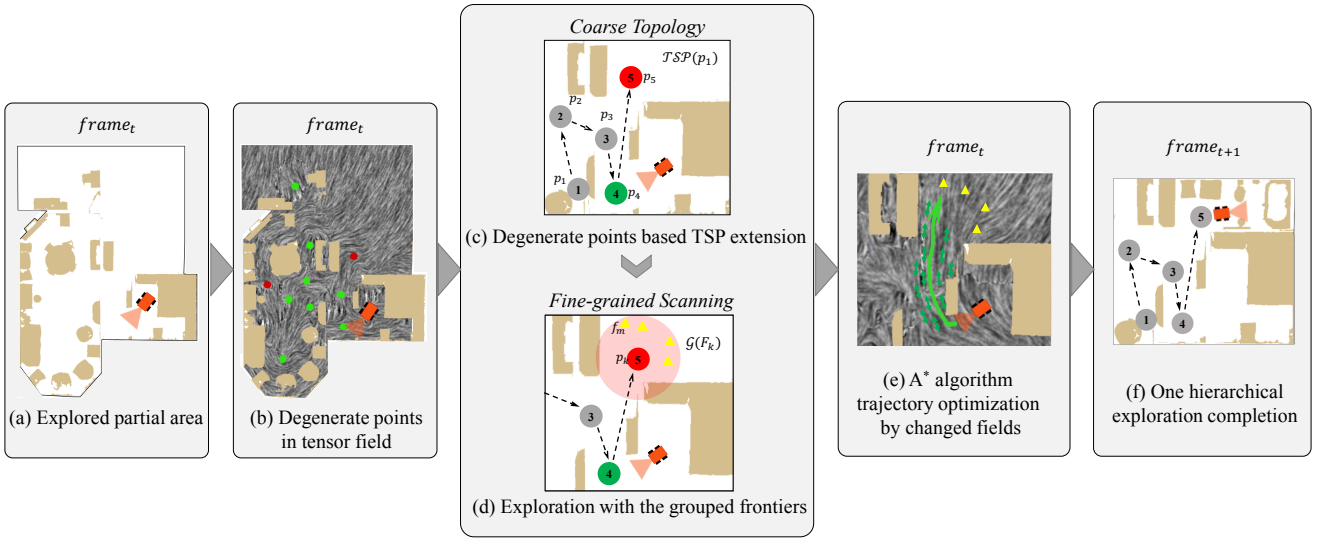


Fig. 2 We show one exploration at frame t in detail to describe the pipeline of THP. Firstly, THP obtains a set of degenerate points (green circles) and generated degenerate points during exploration (red circles) in (b) for the partial scene (a). Secondly, degenerate points based *TSP* extension (see Section 3.3) dynamically extends the *TSP* by the generated points. The extended $\mathcal{TSP}(p_1)$ provides the order for all degenerate points with starting point p_1. Next, fine-grained scanning is conducted within the group F_k computed by $\mathcal{G}(F_k)$ which groups the frontiers f_m to the nearest degenerate point p_k (see Section 3.4). Once the frontier in F_k has been explored, the degenerate point p_k is marked as visited (gray circles). Meanwhile, when the trajectory is planned by A^*, the surrounding tensor fields will also change and create a new advection (green arrows) for the robot to avoid obstacles (see Section (3.5)). Overall, complete and efficient exploration is accomplished by THP for frame t.

2.2 Frontier-based Exploration

Unlike field-guided path planning, frontier-based methods focus on exploration completeness. Yamauchi [12] introduced the notion of frontiers, the boundary between free space and unexplored space, to guide robot exploration. They can help the robot to perform more complete and efficient exploration. However, they usually make wrong decisions due to lack of global planning. Dirk et al. [13] proposed two simple heuristics to improve [12]. The first is to repeatedly validate the current target. They also adopt Voronoi diagrams to ensure a more complete exploration of the entire room. This method depends on the correctness of the scene segmentation while it is challenging to make one in an open environment. Miroslav [14] formulates this problem as a *traveling salesman problem (TSP)* and uses the chained Lin-Kernighan heuristic to define the distance cost. Doing so can significantly reduce the exploration time and generate more feasible trajectories. However, unnecessary computation is repeated on all unvisited frontiers even if some frontiers are close. TARE [15] is the state-of-the-art frontier-based method for environment exploration. It requires expensive LiDAR devices to provide high-quality and complete FOV scans, and it is unsuitable for low-cost depth cameras.

3 Method

3.1 Overview

Vector fields and tensor fields are commonly used to guide autonomous robot navigation [9, 16]. Tensor fields can be utilized to provide locally smoother path advection and better obstacle avoidance [7]. In addition, the topology generated by connecting all singularities (degenerate points), provides an efficient reference for global path planning. Therefore, our proposed THP is driven by tensor fields to perform hierarchical exploration.

We find that the degenerate points distributed at the joint points of the scene's structure can act as anchors, used to drive the global path generation process via TSP [17]. These anchors are constantly generated and the trajectories should be updated simultaneously. We formulate an online TSP on these dynamically updating anchors and find a tour with minimal cost connecting all anchors to form a coarse topology for navigation. Note that our *degenerate point based online TSP* design (see Section 3.3) can significantly enhance the original tensor-field-driven method if the environment is complex.

Having a coarse topology for routing, we try to scan the scene at a fine-grained level. A common formulation of such exploration utilizes frontiers. Most previous methods [12, 18] only greedily select frontiers for exploration which leads

to a more time-consuming computation for global optimization. Instead, we design a *frontier grouping strategy* (see Section 3.4) which groups the frontiers near the anchors and provides better ordering for access. This strategy helps the robot perform a complete scan with lower TSP computation cost.

Given the routing topology and next goal position, we then plan the robot's moving path. Directly routing between the current position and a goal point is sometimes discontinuous or ambiguous [7]. To plan the moving path, we adopt A^* [19] to find the shortest collision-free path (see Section 3.5). However, it is time-consuming for large scenes. We find that the tensor fields change in real-time as the environment is updated, which can guide the robot moves along the advection without A^*. This can help us to reduce the frequency of usage of the A^* algorithm and smooth the path. Unlike [15], our method has no requirement for large FOV. Figure 2 shows the pipeline of our method.

3.2 Concepts and Terminology

A tensor field $T \subset \mathbb{R}^{2\times2}$ for a 2D plane $\mathcal{D} \subset \mathbb{R}^2$ is a smooth tensor-valued function which associates every point $p \in \mathcal{D}$ with a second-order tensor:

$$T(p) = \begin{pmatrix} \tau_{11}(p) & \tau_{12}(p) \\ \tau_{21}(p) & \tau_{22}(p) \end{pmatrix}. \tag{1}$$

A tensor $[\tau_{ij}]$ is symmetric if and only if $\tau_{ij} = \tau_{ji}$. The tensor T used in our method is symmetric, and so can be uniquely decomposed into an isotropic part S and an anisotropic part A:

$$T = S + A = \lambda \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} + \mu \begin{pmatrix} \cos 2\theta & \sin 2\theta \\ \sin 2\theta & -\cos 2\theta \end{pmatrix} \tag{2}$$

where $\mu > 0$. A has eigenvalues $\pm\mu$. A deviate tensor field $A(p)$ is equivalent to two orthogonal eigenvector fields when $A(p) \neq 0$:

$$\begin{aligned} E_1(p) &= \mu(p)e_1(p) \\ E_2(p) &= \mu(p)e_2(p) \end{aligned} \tag{3}$$

where $e_1(p)$ and $e_2(p)$ are unit eigenvectors corresponding to eigenvalues μ and $-\mu$. As a result, E_1 and E_2 are the *major* and *minor* eigenvector fields of A. We move the robot directed by the *major* eigenvector in our approach.

Degenerate points are used to construct the coarse topology. We consider point p to be a *degenerate point* for a tensor field T if and only if $A(p) = 0$, and *regular* otherwise. The most basic types of degenerate points are *wedges* and *trisectors* (see Figure 3).

Next, we consider how to build a complete tensor field. A tensor field is constructed on a set of *elements*. An *element* can be either *regular* if a tensor value is specified, or *singular* if a particular type of degenerate point is needed. Every element is extended to a *basis tensor field* and the final tensor field is a sum of these basis fields [20].

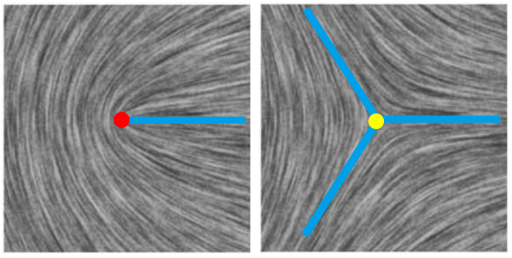


Fig. 3 Degenerate points. Blue lines are separatrix lines.

Given a regular element (S_0, T_0) defined at p_0, we calculate $\rho_0 = \sqrt{S_0^2 + T_0^2}$ and $\theta_0 = \arctan(T_0/S_0)$. A basis field is then defined as:

$$\begin{aligned} T_0(p) &= \rho_0 \begin{pmatrix} \cos 2\theta_0 & \sin 2\theta_0 \\ \sin 2\theta_0 & -\cos 2\theta_0 \end{pmatrix} \\ T(p) &= \exp\left(-d\|p - p_0\|^2\right) T_0(p) \end{aligned} \tag{4}$$

where d is a decay constant used to control the influence of the basis field. The exponential weight is strong at the center of the element and becomes weaker for distant points p. This ensures that the sum of basis tensor fields maintains desired values at the specified location.

3.3 Degenerate Point Based Online TSP

3.3.1 Idea

[7] views the topological skeleton of the tensor field as an undirected graph with nodes all the degenerate points and edges the separatrices connecting them. Inspired by [7], we utilize the graph which has a rough coverage of the entire scene to drive the topology-based path routing. However, degenerate points are generated during exploration. [7] routes using static degenerate points and so is unable to support global path optimization. Previous works use a *greedy* strategy to find the exploration trajectory, always selecting the nearest point for the next goal position. Such a fixed strategy prefers the goal position closest to the robot without considering subsequent actions.

Instead, we formulate an online TSP by conducting TSP on updated degenerate points to maintain an optimal tour order for them. This ensures that the coarse topology for navigation is dynamically generated. Furthermore, our online TSP can also reduce the time needed to make exploration decisions. Our experiments (see Figure 8) demonstrate the efficiency of our method.

3.3.2 *Online TSP formulation*

Given the tensor fields T in the current state, we obtain a set of degenerate points $\mathcal{P} = \{p_0, \ldots, p_i\}$, $p_i \in \mathcal{D}$. The set of new generated degenerate points is denoted $\mathcal{P}^* = \{p_0, \ldots, p_j\}$. The updated $\mathcal{P}$ is defined as:

$$\begin{aligned} \mathcal{P} \leftarrow \mathcal{P} \cup \mathcal{P}^* = \{p_0, \ldots, p_n\}, \quad n = i + j \\ \mathcal{P} \cap \mathcal{P}^* = \emptyset \end{aligned} \tag{5}$$

If a degenerate point p_i is visited, we remove it from $\mathcal{P}$: $\mathcal{P} \leftarrow \mathcal{P} - \{p_i\}$.

We denote the length of the path between degenerate points i and j as $\text{dist}(p_i, p_j)$. Given degenerate points $\{p_0, \ldots, p_n\}$, the TSP is an optimization problem: find a permutation $\Pi = \{\pi_1, \ldots, \pi_n\}$ of $\mathcal{I} = \{1, \ldots, n\}$ starting from p_0 such that the length of the tour over all permutations is minimal. The TSP problem is thus formulated as:

$$\mathcal{TSP}(p_0) = \min_{\Pi}(\text{dist}(p_0, p_{\pi_1}) + \sum_{i=1}^{n-1} \text{dist}(p_{\pi_i}, p_{\pi_{i+1}})) \tag{6}$$

The point p_{π_1} is then selected as the next goal position. Note that the TSP finds the best-closed tour and ends at an arbitrary point instead of returning to the beginning. This prevents the TSP solver from producing unnecessarily long trajectories.

3.3.3 *Online TSP Solver*

To solve the problem, we should first select the optimal starting point p_0. It is natural to find the nearest degenerate point p_0 to the robot location c_r using $\text{dist}(p_0, p_r)$. This strategy prevents the robot from moving far away to start traveling. After choosing the starting point, we have to calculate the distance between the current point and the next as cost for object function optimization. Euclidean distance is commonly adopted to calculate spatial distance. However, it is not an accurate measure of the length of a path when there are obstacles (e.g. walls, furniture) between two points. Therefore, we use the A^* algorithm for $\text{dist}(p_i, p_j)$ computation. A solution of TSP is found using the starting point and distance cost. Algorithm 1 summarizes the extension process.

Algorithm 1 Online TSP.

Input: Degenerate point set: $\mathcal{P} = \{p_0, \ldots, p_i\}$.

while $\mathcal{P}$ is not $\emptyset$ **do**
  Select the starting point closest to the robot:
  $p_0 = \arg\min_{p_0 \in \mathcal{P}} \text{dist}(p_0, c_r)$;
  Compute $\mathcal{TSP}(p_0)$ giving order
  $\Pi = \{\pi_1, \ldots, \pi_n\}$;
  Add starting point to Π: $\Pi \leftarrow \{p_0\} \cup \Pi$
  foreach π_n in Π **do**
    Visit p_{π_n};
    Remove p_{π_n} from $\mathcal{P}$: $\mathcal{P} \leftarrow \mathcal{P} - \{p_{\pi_n}\}$;
    Compute extension degenerate points: $\mathcal{P}^*$:
    $\mathcal{P}^* = \{p_0, \ldots, p_j\}$;
    if $\mathcal{P}^*$ is not $\emptyset$ **then**
      Break from the loop;
    end if
  end foreach
  Update the degenerate point list: $\mathcal{P} = \mathcal{P} \cup \mathcal{P}^*$;
end while

3.4 Frontier grouping strategy

3.4.1 *Approach*

To plan exploration, our method scans the scene at a fine-grained level based on the coarse topology. Most frontier-based exploration methods detect all frontiers and iteratively select the most appropriate frontier as the next robot goal to generate the trajectory. Miroslav et al. [14] formulate the TSP problem on frontiers to decide the exploration order, and the computation is repeated on each unvisited frontier, which is time-consuming. Instead, we have planned an optimal tour topology based on degenerate points and so exploration can be carried out along the topology. Therefore, we dynamically group the frontiers around the degenerate points. The key idea is that the global path generated from the topological skeleton of the degenerate points covers the entire scene, so provides a reasonable basis for frontier grouping. In addition, exploration efficiency will be improved by visiting groups instead of visiting a single one.

3.4.2 *Frontier grouping*

The frontier is the boundary between free space and the explored area [12]. Let the position of the frontier be f. For each f_m in $\mathcal{F} = \{f_0, \ldots, f_m\}$, we calculate the Euclidean distance between it and degenerate points $\mathcal{P} = \{p_0, \ldots, p_n\}$ to find the nearest degenerate point p_k:

$$d_k(f_m) = \arg\min_{p_k \in \mathcal{P}} \|f_m - p_k\|. \tag{7}$$

Then we define a grouping function $\mathcal{G}(F_m)$ which groups f_m into F_k whose distance to p_k is smaller than the distance threshold d_f:

$$\mathcal{G}(F_k) = \{f_m | \, \|f_m - d_k(f_m)\| \leqslant d_f\} \tag{8}$$

F_k is updated to include f_m: $F_k \leftarrow F_k \cup \{f_m\}$. The threshold d_f is set to 2 m in our experiments.

However, some frontiers are too far to be grouped: $\|f_m - d_k(f_m)\| > d_f$. To cover these frontiers, we take them as degenerate points to automatically extend the degenerate point list $\mathcal{P} \leftarrow \mathcal{P} \cup \{f_m\}$.

After grouping, we scan each frontier group along the degenerate points in order of permutation Π. During scanning,

Algorithm 2 Frontier grouping and splitting.

Input: Degenerate points ordered by TSP: $\mathcal{P} = \{p_0, p_1, \ldots, p_n\}$.
Input: Frontier set: $\mathcal{F} = \{f_0, \ldots, f_m\}$.

foreach f_m in $\mathcal{F}$ **do**
  if $\mathcal{G}(f_m)$ is not $\emptyset$ **then**
    $p_k = \mathcal{G}(f_m)$;
    Add f_m to group F_k: $F_k = F_k \cup \{f_m\}$;
  else
    $\mathcal{P} = \mathcal{P} \cup \{f_m\}$;
  end if
end foreach
foreach group F_k **do**
  foreach $f_k \in F_k$ **do**
    if $\mathcal{S}(f_k) = \emptyset$ **then**
      Set f_k as a degenerate point:
      $\mathcal{P} \leftarrow \mathcal{P} \cup \{f_k\}$;
    else
      Scan f_k;
      Remove f_k from group F_k:
      $F_k \leftarrow F_k - \{f_k\}$;
    end if
    if $F_k = \emptyset$ **then**
      Remove p_k from $\mathcal{P}$: $\mathcal{P} \leftarrow \mathcal{P} - \{p_k\}$;
      Break from the loop;
    end if
  end foreach
  Delete the group F_k;
end foreach

if the frontier $f_k \in F_k$ is visited, it is removed from the group $F_k \leftarrow F_k - \{f_k\}$. When the group F_k is empty, the degenerate point p_k is also considered as visited and removed from the degenerate point list: $\mathcal{P} \leftarrow \mathcal{P} - \{p_k\}$. Then, we follow the permutation Π to find the next degenerate point and scan its corresponding group.

3.4.3 Group splitting

However, simply grouping the frontiers is not sufficient because an obstacle (e.g. a wall) may lie between two frontiers in the same group. A robot path including two such frontiers will include long detours. See Figure 4(a): when the robot is exploring frontier group F_i in the top-left room, it is a poor choice for it to go out to scan the other frontiers of F_i (blue trajectory) and return to the room to visit the next group $\mathcal{F}_j$ (pink trajectory). To solve the problem, any group spanning two rooms is split dynamically.

Let the distance between the robot's location c_r and group F_k be $\|c_r, p_k\|$ as measured by Euclidean distance between c_r and p_k. The path length between $f_k \in F_k$ and the robot is

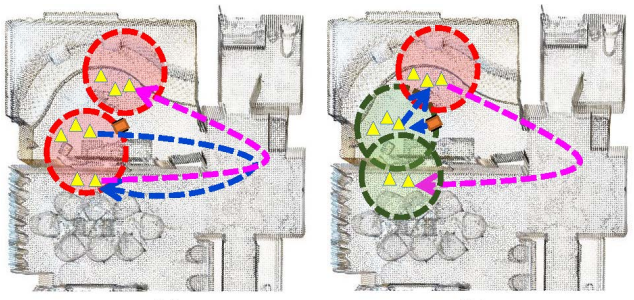


Fig. 4 Group splitting. An obstacle (e.g. a wall) may lie between two consecutive frontiers of one group, resulting in long detours (a). We dynamically divide the frontier into two groups (b).

$\text{dist}(c_r, f_k)$ which is calculated using the A^* algorithm. Then we define a split function $\mathcal{S}(f_k)$ based on these two distances. $\mathcal{S}(f_k)$ splits F_k by filtering out frontiers whose path length to the robot is significantly different from the distance between the robot and its original group center:

$$\mathcal{S}(F_k) = \{f_k | f_k \in F_k, \nu \geqslant d_g \vee (\mu < d_p \wedge \nu < d_g)\} \quad (9)$$

The filtered frontiers $\{F_k - \mathcal{S}(F_k)\}$ are viewed as new degenerate points and inserted into $\mathcal{P}$. The thresholds d_g and d_p are set to 2 m and 3 m respectively in our experiments.

As Figure 4(b) shows, the planned trajectories are shorter after splitting such a group (green circles), reducing exploration time. Details are given in Algorithm 2.

3.5 A^* algorithm based path planning

Given the coarse topology and next goal position, the exploration trajectory should be planned in the next step. [7] directly computes the robot movement path as a pathline defined by a particle advected by the fields. However, the path may be discontinuous and ambiguous if the scene is complex. The A^* algorithm is a common method for path planning, and we adopt it to find the shortest collision-free path between the current position and the next exploration goal point.

The depth sensor has a limited FOV, and it cannot be freely used to enhance efficiency of exploration, as moving safety is also critical. In our strategy, the depth sensor generally faces the robot's moving direction to prevent collisions. Therefore, our planning algorithm only outputs positions for the moving trajectory.

A^* computation is time-consuming for large scenes. To reduce the frequency of A^* algorithm computation, TARE [15] adopts an offline searching strategy to find a local movement path but it highly depends on having a complete FOV. In contrast, we move the robot with the power of tensor fields (see Figure 5(b)). We note that in our approach, the path planned

by A^* changes the surrounding tensor fields, but the changed tensor fields also provide new advection for A^* to update the trajectory. Positive feedback between the planned path and tensor fields help our approach to further guide the robot to determine a better path. Consequently, the frequency of the A^* algorithm computation is significantly reduced.

Given a global path $\mathcal{T}$ planned by the A^* algorithm, we compute the surrounding tensor field. [7] projects the grid cells onto the floor plane and performs farthest point sampling with distance $d_s = 0.2$ m over the centers of the projected boundary cells to select a set of 2D constraint points. We project $\mathcal{T}$ to the plane to sample constraint points $\mathcal{P}_c = \{p_0, \ldots, p_c\}$. Note that the points in $\mathcal{P}_c$ are ordered by the direction of the path. Then we compute the regular element (S_c, T_c) based on the two sequential points p_{c-1} (x_{c-1}, y_{c-1}) and p_c (x_c, y_c):

$$\begin{aligned} S_c &= x_c - x_{c-1} \\ T_c &= y_c - y_{c-1} \end{aligned} \tag{10}$$

Next, we compute the basis tensor field $T_c(p)$ for every constraint point $p_c \in \mathcal{P}_c$ using Eq. 4. The fields are regular, with major eigenvector aligned with the tangent to the path through the points on the 2D path. Therefore, the fields can provide new advection for the robot to move and avoid obstacles. The final tensor field is computed by summing the basic fields of all constraint points using a Gaussian radial basis function:

$$T(p) = \sum_c \exp\left(-d \left\|p - p_c\right\|^2 / \sigma^2\right) T_c(p) \tag{11}$$

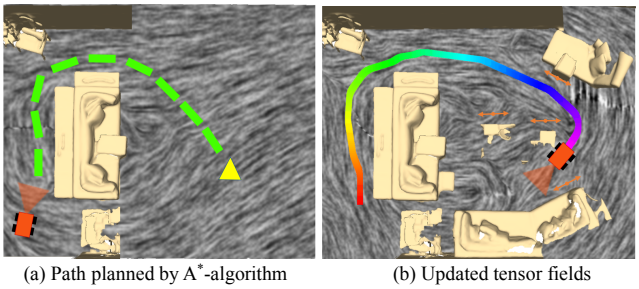
with decay constant $d = 1$. The Gaussian bandwidth σ can be used to control the range of influence of a basic field; we set the value to $\sigma = 2.5 d_s$. Figure 5(a) shows fields generated on $\mathcal{T}$.

The advantages of this design are clear. The field updating process is repeated during global path routing, and corresponding degenerate points may disappear which solves the problem of discontinuous or ambiguous movement. In addition, the robot can route along the particle advection of fields when an obstacle blocks the global path without another A^* search calculation. Figure 5(b) shows changes in the tensor fields affected by the global path.

4 Experiments

4.1 Setting

We have implemented our method using ROS on a 3.8GHz AMD3900X computer, using a single CPU thread. The perception range of the depth camera is set to 5 m in our simulation. When computing the tensor field, we employ a 2D spatial grid with a resolution of 0.1 m.



(a) Path planned by A*-algorithm (b) Updated tensor fields

Fig. 5 Changes in tensor fields affected by the A^* planned path. (a) A path is generated by the A^* algorithm but blocked by the obstacles in the unexplored area. (b) The robot moves along the path and surrounding tensor fields are changed, providing new advection for obstacle avoidance.

To ensure efficiency, the global path given by A^* can only modify the tensor field within its $\sigma = 2.5 d_s = 0.5$ m range. For ROS simulation, the maximum speed is set to 1m s^{-1} for the ground vehicle. The vehicle is equipped with a fixed (forward-looking) RGB-D camera (Kinect), used for exploration and mapping.

4.2 Benchmark

4.2.1 *Datasets and comparison*

We conduct comparison based on the Matterport dataset [21]. We use all 8 large single-layer scenes which are available for complete exploration via a ground robot in the simulation. We compare our method to three state-of-the-art methods on this dataset: *TARE*[15], *BAYES*[22] and *TENSOR*[7], and to further demonstrate the advantages of our method, we also use two further specifically designed alternatives:

- *TARE*: A state-of-the-art exploration framework based on LiDAR scans.
- *BAYES*: The planning strategy uses a Bayesian formulation for target point selection when exploring an unknown environment.
- *TENSOR*: The method harnesses a time-varying tensor field to guide robot movement.
- *GREEDY*: A frontier-based strategy that guides robot exploration based on [12].
- *TARE-RGBD*: To make a fair comparison, we also straightforwardly implemented TARE using depth camera input.

4.2.2 *Metrics*

We use several basic metrics to assess our approach.

- *Completeness* ϵ is defined as the explored volume over the entire run, indicating the completeness of scanning.
- *Length of trajectory* $\mathcal{L}$ indicates the exploration efficiency of the robot.

Table 1 Volumes explored by different methods. Volume is denoted ϵ. The ratio of ϵ compared to that of our method is denoted r_ϵ.

Method	Scene 1		Scene 2		Scene 3		Scene 4		Scene 5		Scene 6		Scene 7		Scene 8	
	ϵ	r_ϵ	ϵ	r_ϵ	ϵ	r_ϵ	ϵ	r_ϵ	ϵ	r_ϵ	ϵ	r_ϵ	ϵ	r_ϵ	ϵ	r_ϵ
Ours	65.184	1.0	98.192	1.0	98.352	1.0	93.248	1.0	44.800	1.0	103.664	1.0	145.752	1.0	77.936	1.0
GREEDY	65.080	0.998	93.712	0.954	98.352	**1.0**	93.392	**1.002**	39.576	0.883	87.488	0.844	144.592	0.992	64.256	0.824
TARE	65.376	**1.002**	91.488	0.932	83.808	0.852	93.256	**1.000**	16.944	0.378	31.896	0.308	136.976	0.940	78.496	**1.007**
TARE-RGBD	38.512	0.591	79.944	0.814	44.176	0.449	83.088	0.891	24.624	0.550	41.768	0.403	107.552	0.738	50.024	0.642
BAYES	57.254	0.878	98.057	0.999	58.356	0.593	86.806	0.931	43.600	0.973	71.171	0.687	130.151	0.893	65.048	0.835
TENSOR	34.096	0.523	56.440	0.575	57.504	0.585	59.440	0.637	44.496	0.993	65.352	0.630	124.968	0.857	10.984	0.141

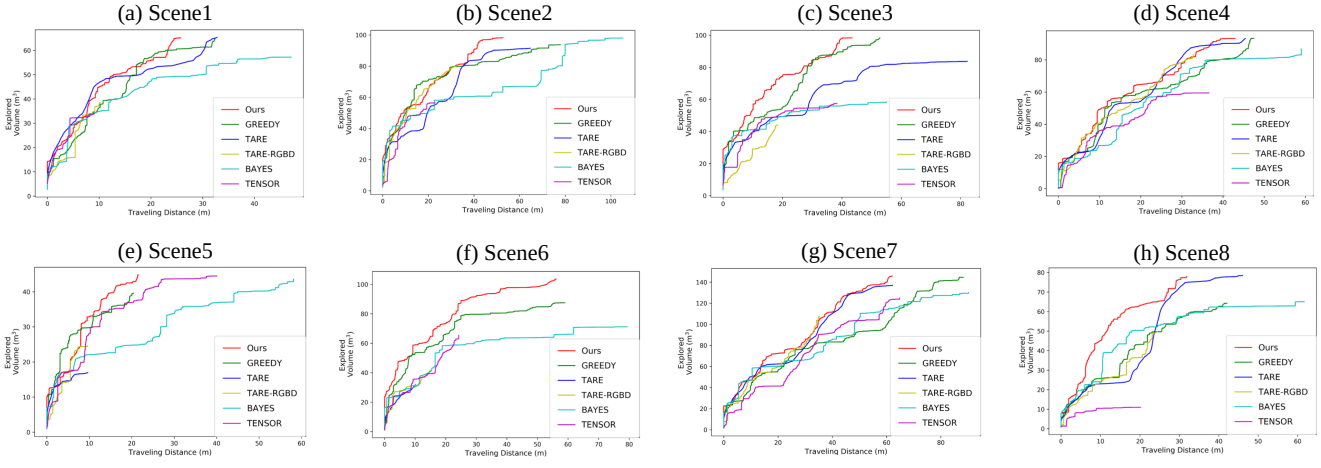


Fig. 6 Volumes ϵ versus length of trajectory $\mathcal{L}$ for various scenes and methods.

4.3 Initial Evaluation

In this section, we conduct a large number of experiments to verify the efficiency of our approach. We confirm the THP plans an efficient path with desired scan quality and show quantitative results in Section 4.3.1. We also analyze the importance of each module in our method in Section 4.4. Visual comparisons are shown between our approach and previous state of the art methods, they directly demonstrate the advantages of THP (see Section 4.5).

4.3.1 Quantitative evaluation

Our evaluations mainly concern two issues, scan quality and efficiency. We show the scanned volumes $epsilon$ for different methods to compare the scan quality. The relative efficiency r_ϵ directly compares efficiency to that of our method. The results are shown in Table 1. Our method only equipped with an RGB-D camera obtains similar scan quality compared to TARE [15], the state-of-the-art method which uses LiDAR scans. Our method outperforms TARE on exploration completeness by 4%. The results demonstrate the efficiency of our hierarchical exploration strategy. GREEDY [12] achieves the same performance on Scenes 3 and 4 but also produces longer trajectories (see Figure 6(c)(d)). TARE-RGBD fails for 50% of the scenes ($r_\epsilon < 0.6$): it stops the robot before accomplishing the exploration, lacking the help of a LiDAR device. TENSOR also fails in some complex environments because of the lack of a globally optimal path.

We analyze the efficiency of exploration based on the length of the trajectory. As shown in Figure 6, with a similar quality of scans, the length of the path generated by TARE is slightly longer than ours in Scenes 1, 2, and 8. We achieve 20% better exploration efficiency on average compared to TARE. BAYES produces the longest paths in 7 out of 8 scenes. In Figure 6(e), we improve performance by 46.1% compared to TENSOR which is also driven by tensor fields. In addition, in half of the scenes, GREEDY plans the same trajectory length as ours but sacrifices the quality of scanning. It is noteworthy that the slope of TENSOR's curve is large at the beginning which indicates that TENSOR explores efficiently in the initial easy environment although it may fail when the environment becomes complex.

We also analyze the memory consumed by generated tensor fields for different scenes. As Figure 7 indicates, we find that the amount of memory consumed is proportional to the size of the scene. At the same time, we compared the time consumed by the various stages. The results are shown in Table 2. Although tensor field update consumes more time than the other phases, our method can still achieve real-time results since our framework can hide the processing time from the pipeline.

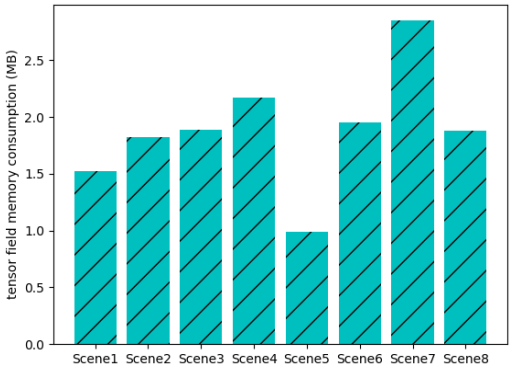


Fig. 7 Memory consumption for different scenes.

Table 2 Average time consumption for the four different stages: (1) tensor fields update, (2) THP & grouping, (3) path planning, and (4) motion control.

Stage	1	2	3	4
Scene 1	216.3 ms	1.2 ms	2.9 ms	49.3 ms
Scene 2	381.5 ms	1.7 ms	5.8 ms	48.7 ms
Scene 3	370.1 ms	1.7 ms	8.5 ms	45.1 ms
Scene 4	320.0 ms	1.6 ms	5.5 ms	45.5 ms
Scene 5	125.8 ms	1.0 ms	3.5 ms	46.7 ms
Scene 6	240.0 ms	1.5 ms	3.1 ms	46.0 ms
Scene 7	472.6 ms	1.8 ms	6.0 ms	51.3 ms
Scene 8	284.4 ms	1.5 ms	6.3 ms	54.0 ms

Table 3 Directly routing with degenerate points and paths generated from extended generate points. $\mathcal{N}_d$, $\mathcal{N}_d^*$: number of degenerate points and extended degenerate points respectively. ϵ: scanned volume. Path length: $\mathscr{L}$.

	Scene 1			Scene 2		
	$\mathcal{N}_d$	$\mathcal{N}_d^*$	$\mathscr{L}$(m)	$\mathcal{N}_d$	$\mathcal{N}_d^*$	$\mathscr{L}$(m)
w/o extension	12	-	72.16	22	-	81.58
our method	12	21	52.73	22	40	70.65

4.4 Ablation study

In this section, we conduct a series of experiments to evaluate the contribution of each module in our method. We also analyze the results to demonstrate their advantages and importance.

4.4.1 Effectiveness of extended degenerate points

We argue that a limited number of degenerate points is unable to support a global path for a more complex scene. Using degenerate points generated during the exploration provides an updated topology for TSP to extend the path. The planned path thus allows the robot to route more efficiently and scan more completely. We show the results of a planning strategy without extended degenerate points in our method in Table 3. Using extended degenerate points helps the robot reduce the path length by 19.43 m and 10.93 m for Scenes 1 and 2 respectively. The corresponding reductions in path length are 26.9% and 13.4%.

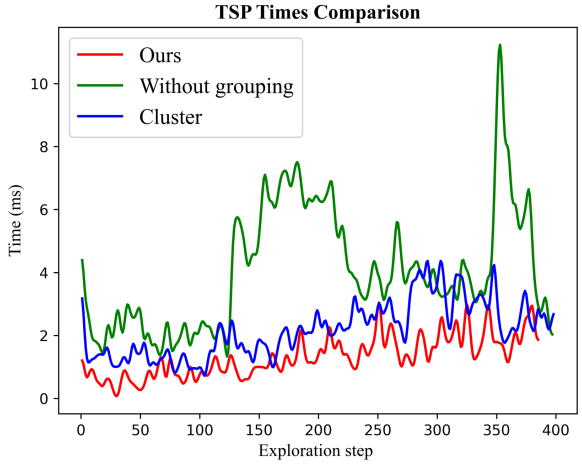


Fig. 8 TSP runtime comparison for different grouping strategies.

4.4.2 Effectiveness of frontier grouping

We compare different frontier grouping strategies in this experiment. Figure 8 gives the TSP runtime for each exploration step. We implement a strategy without grouping which conducts TSP on each frontier. We also implement the k-means clustering strategy on the frontier as described in [14]. Overall, our approach (red line) performs well in almost all steps has lowest TSP cost. The method without grouping (green line) is the most time-consuming. The reason is that a frontier is usually adjacent to at least one frontier and these frontiers can be grouped for visiting together. Our method conducts TSP on each group instead of each frontier and thus reduces the frequency of TSP computation. The performance of the clustering strategy is similar to that of our method but slightly worse than ours in most cases. Clusters often span two rooms which produces needless detours. The results of our approach and the clustering method demonstrate the significance of our group splitting strategy.

4.4.3 Effectiveness of A^* based path planning

In order to verify the advantage of the path planned by the A^* algorithm, we conduct a number of comparisons. We sample partial scenes at different scales ranging from 1 m to 10 m for the robot to move in. For each scene, we randomly crop two partial areas on 10 different scales. Overall, this partial scene dataset consists of 160 cases. Note that these partial scenes are not simple, and obstacles are placed appropriately in them to increase the challenges for navigation.

The success ratio of navigation completeness is calculated to measure the performance of the planning method. We plot the average success ratio of THP (red line) and TARE (blue line) with corresponding standard deviations (red area and blue area) in Figure 9. TARE [15] is the state-of-the-art method and uses LiDAR scans. However, our method has

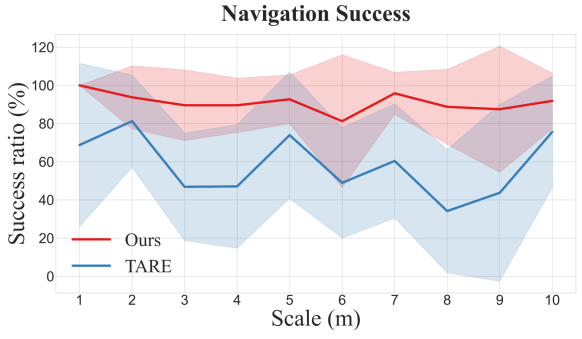


Fig. 9 Ratio of navigation success for THP and TARE at different scales.

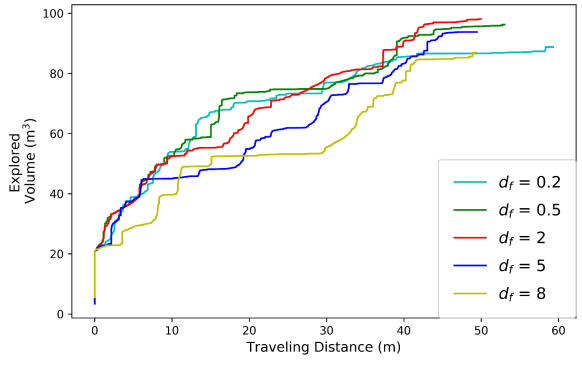


Fig. 10 Efficiency for different values of d_f.

a higher success ratio than TARE and exceeds it by a large margin at each scale. Note that the standard deviations of our approach are also lower than TARE's at every scale which indicates that THP is more stable. TARE can easily fail for 9 m scale areas which contain many obstacles and thus are more complex, because the short paths planned offline by TARE without dynamic global optimization have limited capacity to cope with large or long obstacles. In contrast, our method performs well whether the scene is simple or complex. Once a path is planned by the A^* algorithm, the surrounding tensor fields will be changed. Then the updated tensor fields provide new advection for the robot to move which is critical to obstacle avoidance.

4.4.4 Effect of varying d_f.

To assess the effect of different parameter settings on the efficiency of exploration, we conducted some comparative experiments. We set the distance threshold when grouping d_f to various values from 0.2 m to 8 m. As Figure 10 shows, the setting of d_f should neither be too large nor too small, and a reasonable distance should be customized according to the characteristics of the indoor space. We also experimented with the parameters d_g and d_p when group splitting, as well as the parameters σ when path planning, using values from 1 m to 5 m, but found that changing values had little impact on results.

4.5 Visual comparison

We show the visual results from six indoor scenes in Figure 11. Overall, our method achieves excellent scene coverage giving high-quality scanning with a reasonable planned path for all of these scenes. Compared to other methods based on RGB-D cameras, such as GREEDY, TARE-RGBD, BAYES, and TENSOR, we obtain more complete scanning results, benefiting from our tensor field driven hierarchical exploration. These methods usually quickly fail and cannot explore further because of the complex environment: see Figure 11(a),4–6, etc. Note that we obtain comparable results to TARE which uses precise and long-distance LiDAR scans. However, TARE may also fail when the exploration area is long and narrow(Figure 11(d)-3). Our method performs stably in both situations.

With similar scanning results, the path planned by our method is more efficient and clearer. Compared to our method, GREEDY generates a longer trajectory (Figure 11(a)-2), and elsewhere even passes through a wall (left of Figure 11(e)-2). The path generated by BAYES is disorganized and looping (Figures 11(c)-5 and 11(e)-5). In addition, the BAYES method needs more than 30 minutes and is not as efficient as other alternatives. TENSOR fails in areas with complex structures. These comparisons demonstrate the efficiency of our planning strategies.

When producing the coarse topology for global path routing, it is desirable that the topology of the extended degenerate points reflects the floor structure of the scene being scanned. The visiting order generated by TSP is important in guiding robot movement. However, the pathline generated by advection in tensor fields is usually blocked or disorientated (Figures 11(a)-6 and 11(b)-6), which leads to termination.

A fine-grained level scanning strategy helps the robot explore more carefully and reduce the length of the trajectory. In Figure 11(c)-1,2 we see that the second room from the right-bottom scanned by our method is more complete than when scanned by GREEDY and the trajectory is shorter. The reason is that the robot directly explores the room on grouped frontiers and thus does not route on each frontier to move into the inside of the room. The results show the advantage of the frontier grouping strategy for exploring small rooms (e.g. cubicles).

Our approach performs well on different scenes, benefiting from the A^* planned path. As shown in Figure 11(c)-1,4, TARE-RGBD and our method both use A^* to generate the routing trajectory. However, TARE-RGBD falls into a trap and stops exploring. The positive feedback between the

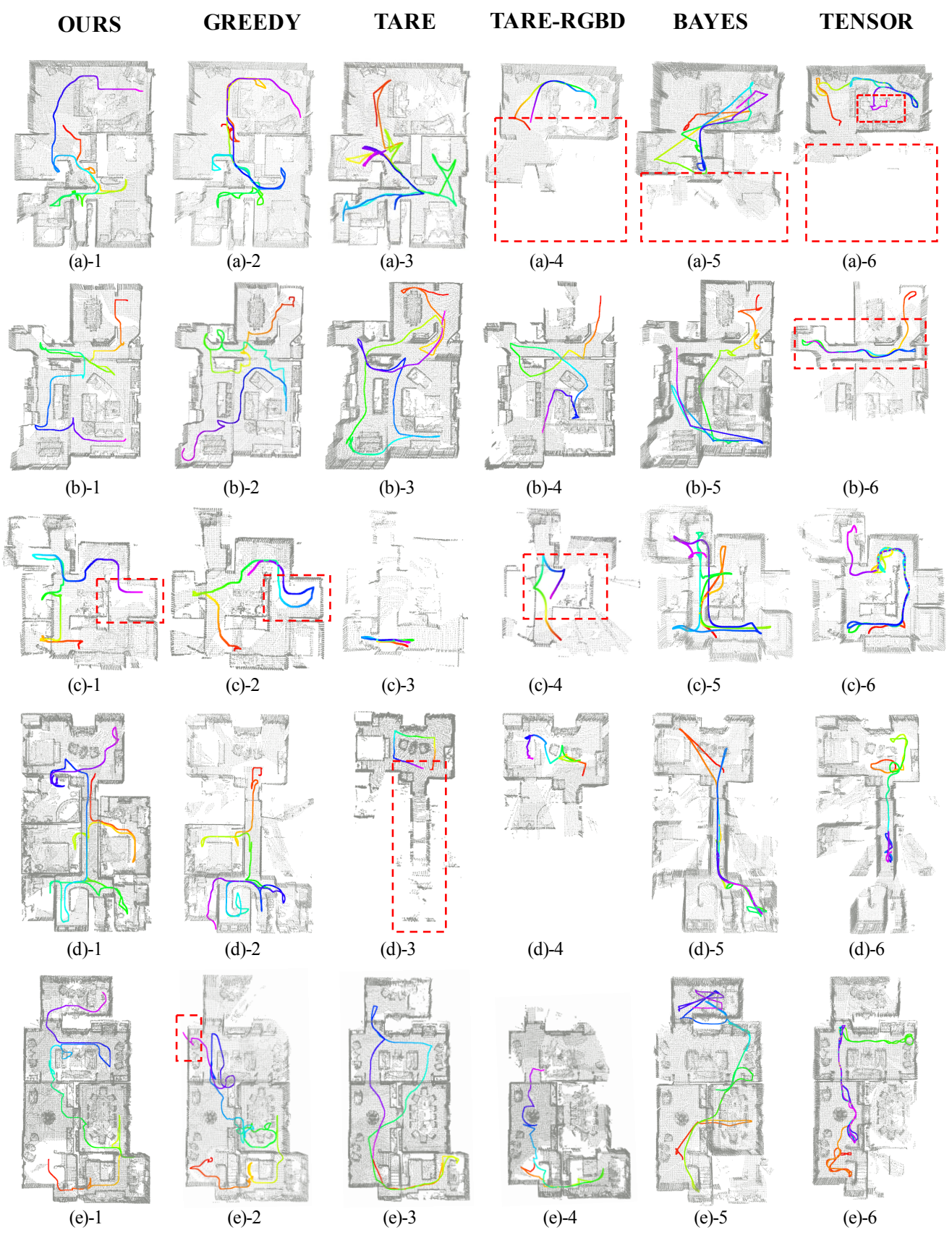


Fig. 11 Visual results for different methods on different scenes. Both the scanning completeness, and planned paths are shown, allowing an intuitive comparison.

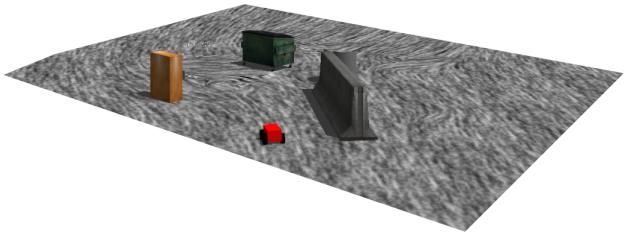


Fig. 12 Less meaningful tensor fields in an open area.

A^* planned path and the updated tensor fields help the robot move smoothly and avoid obstacles.

5 Conclusions

This paper presents a hierarchical robot exploration framework based on tensor fields. The dynamically added degenerate points provide a good reference to form an optimized environment structure topology, enhancing the robot's global perception and so improving efficiency. Furthermore, a coupled frontier group-based exploration strategy is introduced to perform fine-level local scans. These two strategies together formulate our hierarchical exploration framework, which can perform complete exploration of an unknown indoor scene with high efficiency. In addition, we utilize the real-time capability of the tensor field to propose a novel A^* based path planning method. Our method can perform similarly to LiDAR-based methods, in real-time, yet with a limited FOV. The experimental results demonstrate the superiority of our method.

However, due to the perceptual limitation of the depth camera, our method can only explore indoor scenes. Results are not ideal if the environment is an open area since the tensor fields are less meaningful in this case, and cannot guide robot movement effectively, as Figure 12 indicates. Degenerate points are critical for global perception in our method, so an environment with few geometric features, such as a large empty room, cannot demonstrate our advantages over state-of-the-art alternatives.

We assume the camera and vehicle are fixed together in our method. Independently optimizing the camera and vehicle's orientations to maximize efficiency is an interesting possibility. Taking reconstruction quality into consideration as well as exploration efficiency is also worth some effort for many intelligent robotic applications. We hope the proposed tensor field-based framework can inspire further robotic applications based on geometry.

6 Declarations

6.1 Availability of data and materials

Open datasets in the manuscript are from a public repository (https://niessner.github.io/Matterport/).

6.2 Competing interests

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

6.3 Funding

This work was supported in part by the National Key Research and Development Program of China (2018AAA0102200), National Natural Science Foundation of China (62132021, 61902419, 62002375, 62002376, 62102435), Natural Science Foundation of Hunan Province of China (2021RC3071, 2021JJ40696) and NUDT Research Grants (ZK19-30, ZK22-52).

6.4 Authors' contributions

Yuefeng Xi: Methodology, Writing Draft, Visualization, Results Analysis; **Chenyang Zhu:** Methodology, Supervision, Results Analysis; **Yao Duan:** Supervision, Results Analysis; **Renjiao Yi:** Supervision, Results Analysis; **Lintao Zheng:** Supervision, Results Analysis; **Hongjun He:** Supervision, Results Analysis; **Kai Xu:** Methodology, Supervision.

6.5 Acknowledgements

We thank the anonymous reviewers for their valuable comments. We are grateful to Jiazhao Zhang and Chenyi Liu for the fruitful discussions.

References

[1] Zeng R, Wen Y, Zhao W, Liu YJ. View planning in robot active vision: A survey of systems, algorithms, and applications. *Computational Visual Media*, 2020, 6(3): 225–245.

[2] Höller B, Mossel A, Kaufmann H. Automatic object annotation in streamed and remotely explored large 3D reconstructions. *Computational Visual Media*, 2021, 7(1): 71–86.

[3] Bourgault F, Makarenko A, Williams S, Grocholsky B, Durrant-Whyte H. Information based adaptive robotic exploration. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 1, 2002, 540–545 vol.1, doi: 10.1109/IRDS.2002.1041446.

[4] Umari H, Mukhopadhyay S. Autonomous robotic exploration based on multiple rapidly-exploring randomized trees. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, 1396–1402, doi:10.1109/IROS.2017.8202319.

[5] Maurović I, Petrović I, et al.. Autonomous exploration of large unknown indoor environments for dense 3D model building. *IFAC Proceedings Volumes*, 2014, 47(3): 10188–10193.

[6] Senarathne PGCN, Wang D. Towards autonomous 3D exploration using surface frontiers. In *2016 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, 2016, 34–41, doi:10.1109/SSRR.2016.7784274.

[7] Xu K, Zheng L, Yan Z, Yan G, Zhang E, Niessner M, Deussen O, Cohen-Or D, Huang H. Autonomous reconstruction of unknown indoor scenes guided by time-varying tensor fields. *ACM Transactions on Graphics (TOG)*, 2017, 36(6): 1–15.

[8] Zhang J, Hu C, Chadha RG, Singh S. Falco: Fast likelihood-based collision avoidance with extension to human-guided navigation. *Journal of Field Robotics*, 2020, 37(8): 1300–1313.

[9] Khatib O. Real-time obstacle avoidance for manipulators and mobile robots. In *Proceedings. 1985 IEEE International Conference on Robotics and Automation*, volume 2, 1985, 500–505, doi:10.1109/ROBOT.1985.1087247.

[10] Koren Y, Borenstein J, et al.. Potential field methods and their inherent limitations for mobile robot navigation. In *ICRA*, volume 2, 1991, 1398–1404.

[11] Ok K, Ansari S, Gallagher B, Sica W, Dellaert F, Stilman M. Path planning with uncertainty: Voronoi Uncertainty Fields. In *2013 IEEE International Conference on Robotics and Automation*, 2013, 4596–4601, doi:10.1109/ICRA.2013.6631230.

[12] Yamauchi B. A frontier-based approach for autonomous exploration. In *Proceedings 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation CIRA'97. 'Towards New Computational Principles for Robotics and Automation'*, 1997, 146–151, doi:10.1109/CIRA.1997.613851.

[13] Holz D, Basilico N, Amigoni F, Behnke S. Evaluating the Efficiency of Frontier-based Exploration Strategies. In *ISR 2010 (41st International Symposium on Robotics) and ROBOTIK 2010 (6th German Conference on Robotics)*, 2010, 1–8.

[14] Kulich M, Faigl J, Přeučil L. On distance utility in the exploration task. In *2011 IEEE International Conference on Robotics and Automation*, 2011, 4455–4460, doi:10.1109/ICRA.2011.5980221.

[15] Cao C, Zhu H, Choset H, Zhang J. TARE: A Hierarchical Framework for Efficiently Exploring Complex 3D Environments. In *Robotics: Science and Systems*, 2021.

[16] Shade R, Newman P. Choosing where to go: Complete 3D exploration with stereo. In *2011 IEEE International Conference on Robotics and Automation*, 2011, 2806–2811, doi:10.1109/ICRA.2011.5980121.

[17] Papadimitriou CH. The complexity of the Lin–Kernighan heuristic for the traveling salesman problem. *SIAM Journal on Computing*, 1992, 21(3): 450–465.

[18] Kulich M, Kubalík J, Přeučil L. An integrated approach to goal selection in mobile robot exploration. *Sensors*, 2019, 19(6): 1400.

[19] Norvig PR, Intelligence SA. A modern approach. *Prentice Hall Upper Saddle River, NJ, USA: Rani, M., Nayak, R., & Vyas, OP (2015). An ontology-based adaptive personalized e-learning system, assisted by software agents on cloud storage. Knowledge-Based Systems*, 2002, 90: 33–48.

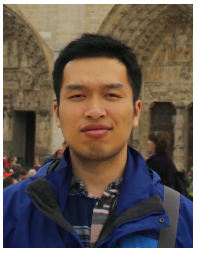
[20] Zhang E, Hays J, Turk G. Interactive Tensor Field Design and Visualization on Surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 2007, 13(1): 94–107, doi:10.1109/TVCG.2007.16.

[21] Chang A, Dai A, Funkhouser T, Halber M, Niebner M, Savva M, Song S, Zeng A, Zhang Y. Matterport3D: Learning from RGB-D Data in Indoor Environments. In *2017 International Conference on 3D Vision (3DV)*, 2017, 667–676, doi:10.1109/3DV.2017.00081.

[22] Bai S, Wang J, Chen F, Englot B. Information-theoretic exploration with Bayesian optimization. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, 1816–1822, doi:10.1109/IROS.2016.7759289.

Author biographies

Yuefeng Xi received his B.E. degree in software engineering from Hebei University of Science and Technology. He is now a master's student at the National University of Defense Technology (NUDT), China. His research interests cover robot perception.



Chenyang Zhu is an Assistant Professor at the School of Computing, NUDT. His current directions of interest include data-driven shape analysis and modeling, 3D vision, robot perception and navigation, etc.

Yao Duan is a Ph.D. candidate in the School of Computing, NUDT. Her current directions of interest include data-driven shape analysis and modeling, 3D vision and scene understanding, etc.

Renjiao Yi is an Assistant Professor in the School of Computing, NUDT. She is interested in 3D vision problems such as inverse rendering and image-based relighting.

Lintao Zheng is an Assistant Professor at the College of Meteorology and Oceanography, NUDT. He earned his Ph.D. in computer science from NUDT. His research interests focus on 3D vision and robot perception.

Hongjun He is a Professor in the School of Computing, NUDT, where he received his Ph.D. in 1998. His current research interests are file system security, machine learning, and software measurement, etc.

Kai Xu is a Professor in the School of Computing, NUDT, where he received his Ph.D. in 2011. He serves on the editorial boards of ACM Transactions on Graphics, Computer Graphics Forum, Computers & Graphics, etc.