2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

Active Collision-Based Navigation for Wheeled Robots

Jingjing Li, Jialin Ji, Qianhao Wang, Huan Yu, Yu Pan, and Fei Gao

***Abstract*— Collision is typically avoided in robot navigation for safety guarantee. However, when a robot's exteroceptive sensors fail, which means it becomes "blind", collision can actually be leveraged to improve localization performance. Our research demonstrates the informative nature of collisions in this context. Moreover, we show that a robot is able to navigate in a known environment with only proprioceptive sensors by actively colliding with its surroundings for more reliable localization. Firstly, we design a collision-based observation model, which is differentiable and can be easily applied to various estimators. Secondly, we integrate this model into a collision-aided localization framework and implement it in two widely used estimators, the Kalman filter and the particle filter. Thirdly, we propose an active collision path planning method, which effectively reduces localization uncertainty.**

I. Introduction

Localization is a fundamental requirement in robot navigation. In order to perceive and estimate their own states, mobile robots commonly rely on sensors such as cameras [1, 2] and LiDARs [3]–[5] to observe the surrounding environment. However, these sensors, known as *exteroceptive* sensors, are susceptible to failures caused by low light conditions, presence of dust or fog, or even signal jamming. In such challenging scenarios, the information available for localization is limited to data provided by *proprioceptive* sensors, such as wheel odometers or inertial measurement units (IMU). But the integration of measurements from these proprioceptive sensors can lead to accumulation of errors, resulting in unreliable odometry estimates. Is there any approach to navigate when robots become "blind"?

One intuitive approach is to draw inspiration from how humans navigate in darkness. Suppose a person returns home in darkness, he can successfully find his way by tactually exploring the environment to locate the light switch, even if it is positioned in the farthest corner of the room. Analysis of this common scenario reveals two key principles: 1) humans possess a familiarity with the layout of their own room; 2) they rely on tactile exploration to sense the surroundings and locate themselves. In robot navigation, for the former, spatial familiarity can be included in a known or partially known map, which is available in many scenarios. For the latter, tactile exploration presents challenges for most mobile robots since physical contact with the environment typically implies collisions with obstacles, which is usually expected to be avoided for safety guarantee. Nevertheless, if we enable robots to cautiously collide with their surroundings, akin to human behavior, valuable "measurements" can be acquired and then effectively combined with prior knowledge of the environment to improve robot navigation performance.

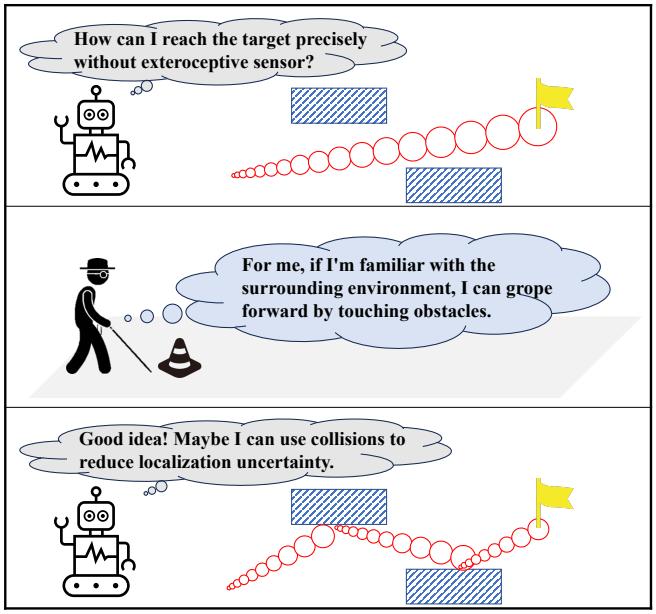

Fig. 1. A robot is considering how to navigate to its goal in a "blind" environment like a human groping forward in the dark. The goal is shown as a yellow flag. Red circles represent the localization uncertainty.

In this paper, we propose an active collision-based navigation method to tackle the problem of navigating in a known environment with only proprioceptive sensors. Firstly, we employ the Euclidean Signed Distance Field (ESDF), which contains surface gradient information, to model the localization information obtained during collisions. The differentiable nature of this modeling approach allows for convenient integration with various estimation methods. In this paper, we provide implementations of this collision observation within particle filtering and Kalman filtering frameworks, which leverage the information acquired from collisions to reduce localization uncertainty. Furthermore, we propose an active collision path planning method that deliberately plans collisions with obstacles in different directions to minimize the localization uncertainty of the robot during navigation towards the goal. In detail, we generate a connected graph by

This work was supported by the National Natural Science Foundation of China under Grant 62322314.

Jingjing Li, Jialin Ji, Qianhao Wang, Yu Pan and Fei Gao are with the Institute of Cyber-Systems and Control, College of Control Science and Engineering, Zhejiang University, Hangzhou 310027, China.

Jingjing Li, Jialin Ji, Qianhao Wang and Fei Gao are with the Huzhou Institute, Zhejiang University, Huzhou 313000, China.

Huan Yu is with the School of Mechanical Engineering, Zhejiang University, Hangzhou 310027, China.

Email: {alan_lee, fgaoaa}@zju.edu.cn

Corresponding Author: Fei Gao.

intelligently sampling states on the surface and in the vicinity of obstacles. This graph construction approach enables effective representation of the environment and potential collision scenarios. Then by incorporating uncertainty constraints, we effectively manage the collisions associated with uncertainty and improve the reliability of robot localization during the navigation. Moreover, we deploy our method on a wheeled robot and conduct experiments to validate its effectiveness.

Summarizing the main contributions of this paper:

1) We design a collision-based observation model that is differentiable and easily applied to various estimators.
2) We integrate the above model into a collision-aided localization framework and implement both particle filtering and Kalman filtering for a wheeled robot.
3) We propose a path planning method based on the above collision-based estimation to reduce the localization uncertainty during navigation by active collisions.

II. Related Work

A. Collision-based Robotic Navigation

Various work is dedicated to design collision-resilient robots. Most of them design an external frame to protect the inner body. The Tiercel [6] uses a carbon fiber frame, a collision bumper and landing gears to suffer from collision. The RMF-Owl [7] designs a frame using carbon-foam sandwich material for prolonged endurance and collision-tolerance.The collision-resilient flying robot [8] uses an outer frame in the shape of a spherical polyhedron to protect the inner frame, where a gimbal system allows the protective frame to rotate independently from the inner frame. The flourishing development of collision-tolerant robots makes it possible for exploiting collisions.

The role of collisions has been developed in various special navigation tasks. Mote et al. [9] show that collision-inclusive trajectory could reduce the thrust needed for a free-flyer spacecraft. Zha et al. [10] show collision-inclusive trajectory achieves shorter time in tunnel like scenario. Mulgaonkar et al. [11] demonstrate that collisions can be used to fly through small apertures, quickly change the moving direction, and simply deploy payloads without additional complex mechanisms. These studies demonstrate the advantages of collisions in robot motion planning. Collisions are capable of quickly changing the robot's motion direction to saving energy or time, and expanding the robot's state space to potentially obtain better solutions.

Additionally, collision can also provide rich information in estimation and localization. Buchanan et al. [12] propose a haptic localization method for quadrupedal robots based on Sequential Monte Carlo methods, which use the touch-down information of the feet to do terrain classification and localization. Mulgaonkar et al. [6] show how a quadrotor use collision against transparent obstacles to build the map of the environment. Similarly, Wu et al. [13] break a long duration flight into multiple short duration hops to reduce the estimation error, using the stationary states (on the ground) between flights to get zero-velocity pseudo measurements for an extended Kalman filter (EKF). Liu et al. [14] design a collision protection system which functions as a touch sensor, demonstrating the potential of exploiting collision impacts for localization through particle filtering. Lew et al. [15] propose a novel contact-based inertial odometry, verifying that it is possible to navigate without any exteroceptive sensors exploiting collisions. However, these works merely utilize collisions passively instead of actively planning them to reduce the uncertainty of localization, which is precisely the problem we aim to address.

B. Observability-aware Planning

Consideration of observations in planning to improve localization accuracy has been well developed in recent years. For conventional sensors, a straightforward approach is to prioritize observations that offer greater localization benefits during the planning process. For instance, selecting areas with a higher density of features for visual sensors [16], and opting for more structured scenes with well-defined objects and surfaces for lidar sensors [17]. However, these task-specialized methods are not generalized for use in navigation using collisions as observations.

Most general approaches [18]–[20] primarily aim to avoid observations that may degrade localization during planning. These approaches typically analyze the information matrix associated with the observation, using the minimum eigenvalue as an indicator to identify potential degradation along the corresponding eigenvector. However, such methods are not applicable to collision observations since collisions inherently lack the ability to provide localization constraints in all directions. To effectively reduce localization uncertainty, it is necessary to consider multiple collisions from different directions. Furthermore, these methods only considers the localization impact of a single-frame observation, which is suitable for continuous observations. Nevertheless, collision observations are intermittent and discontinuous by nature.

In addressing intermittent observations of landmarks, Penin et al. [21] propose a covariance-based path planning algorithm to reduce the localization uncertainty during navigation. Similarly, Napolitano et al. [22] propose a metric based on Constructability Gramian, which aids in guiding planning. However, compared to the comprehensive reduction of localization uncertainty achieved by utilizing observations such as visual or landmark-based measurements in above works, a single collision observation can only provide information in one direction. Consequently, developing an observation-aware path planning algorithm based on collision observations remains a pressing and unresolved issue.

III. Collision-aided Localization

In this section, we introduce the proposed collision-aided localization framework. We design a collision-based observation model and integrated it into two typical estimators: the Kalman filter and the particle filter.

A. Collision-based Observation Model

Compared to *continuous* measurements provided by typical exteroceptive sensors like cameras, LiDARs, and Global

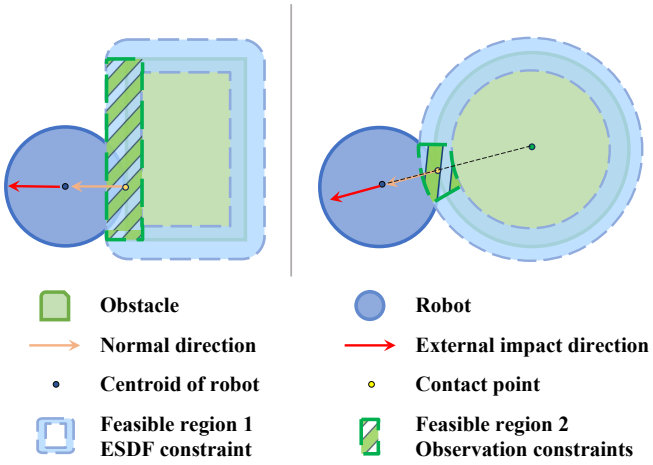

Fig. 2. The feasible region of the contact point after collision observation constraints for a circular obstacle and a polygon obstacle.

Navigation Satellite Systems (GNSS), collisions occur *intermittently*. When the robot physically collides with obstacles, an intuitive measurement equation can be written as $\Phi(x) = 0$, where $\Phi(x)$ represents the signed distance from the nearest obstacles obtained from the ESDF [23] map. Additionally, the gradient of the ESDF map, as shown in Fig. 2, provides information about the normal direction of the nearest obstacle. This information is valuable for updating the estimated states. In summary, our collision-based observation model combines two parts: the ESDF value and its gradient.

Except for the above-mentioned situation, we find another circumstance to leverage "collisionless". In cases where the position prediction from the proprioceptive sensors occurs inside the obstacles but the robot remains physically collision-free, we still update the state using $\Phi(x) = 0$. This strategy is able to prevent further wheel odometer drift until the robot actually collides with the obstacles, shown in Fig. 3(b).

B. Collision-aided Particle Filter (Collision-PF)

Particle filter is a common solution for estimation problems. It uses a number of particles to simulate the state distribution. Each particle represents a possible state and has a weight indicating how similar it is to the true state. The weighted sum of the particles indicates the estimation result.

The process contains the following five steps:

1) Initialization: If the initial position is unknown, uniform distribution can be reasonable. Since we have known the initial state, it's better to sample the particles' initial states in normal distribution. The standard deviations on the position are σ_x and σ_y. The total number of all particles is N.

2) Prediction: The state of particles can be predicted using the linear velocity v_{k-1} from wheel odometers:

$$p_k{}^i = p_{k-1}^i + v_{k-1}\Delta t + n_{k-1}, \tag{1}$$

where Δt is the time difference between two adjacent messages, and n_{k-1} is the prediction noise. This noise simulates the accumulated error of wheel odometers.

3) Update: When a collision occurs, the robot must stay at the surface of the obstacle. So the distance between the robot and the obstacle should be zero, i.e., $d_k = 0$. For the i-th particle, we query the ESDF map to find its distance to the nearest obstacle d_k^i. The direction of the external force f_k can be acquired using IMU's acceleration. According to the assumption that there is no relative motion along the obstacle surface, the tangential friction can be ignored. That is to say, the direction of f_k is the normal direction of the obstacle surface, pointing to the outside. For the i-th particle at position p_k^i, we use the direction of the gradient ∇d_k^i to simulate the direction of the external force.

To calculate the weight w_k^i of each particle, we use the product of two normal distributions. We assume that the distance to the nearest obstacle $\mathrm{D} \sim \mathcal{N}(d_k, {\sigma_d}^2)$, the direction of external force $\mathrm{F} \sim \mathcal{N}(f_k, {\sigma_f}^2)$.

$$\begin{aligned} w_k^i &= f_D(d_k^i) f_F(\nabla d_k^i) \\ &= \frac{1}{\sqrt{2\pi}\sigma_d} e^{-\frac{(d_k^i - d_k)^2}{2{\sigma_d}^2}} \frac{1}{\sqrt{2\pi}\sigma_f} e^{-\frac{(\nabla d_k^i - f_k)^2}{2{\sigma_f}^2}}, \end{aligned} \tag{2}$$

where σ_d and σ_f are standard deviations of the distance and the direction respectively. Then normalize the weights to ensure that the sum of the weights is equal to 1:

$$w_k^i = \frac{w_k^i}{\sum_{i=1}^n w_k^i}. \tag{3}$$

4) Resampling: To avoid the situation that only a handful of particles have large weights while others contribute little, we resample the particles after each update step. We adopt the systematic resampling strategy, which ensures sampling from the whole particle space and particles with larger weights are resampled proportionally more often. It divides the cumulative sum space into N parts, choosing each sample in every $1/N$ part by adding the same random offset.

5) State estimation: Calculate the weighted sum to get the position estimation p_k^* for the collision-PF at time k ,

$$p_k^* = \sum_{i=1}^{n} w_k^i p_k^i. \tag{4}$$

C. Collision-aided Extended Kalman Filter (Collision-EKF)

Here we present the collision-aided Kalman filter method. The state variable we choose is the robot position p_k at time k. The measurement z_k is the ESDF value of the robot centroid. The input of the system is the velocity v_k at time k. The state transition and observation modes are as follows:

$$p_k = f(p_{k-1}, v_k) + \zeta_k = p_{k-1} + v_k \Delta t + \zeta_k, \tag{5a}$$

$$z_k = h(p_k) + \epsilon_k, \tag{5b}$$

where ζ_k is the process noise, $\boldsymbol{\zeta}_k \sim \mathcal{N}(0, \mathbf{Q}_k)$, ϵ_k is the observation noise, $\epsilon_k \sim \mathcal{N}(0, R_k)$, both $\mathbf{Q}_k$ and $R_k > 0$.

The prediction and update process is the same with the standard EKF.

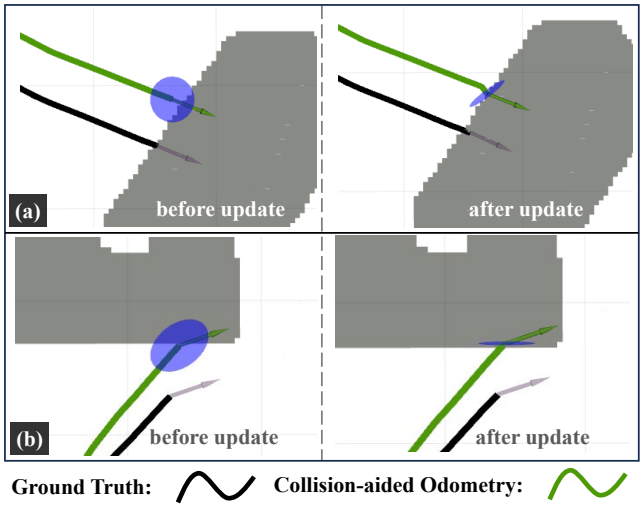

Fig. 3. Two circumstances for collision-based state updating. (a) The robot physically collides with obstacles. (b) The robot is physically collision-free, but the preintegration of wheel odometer is inside obstacles.

Prediction:

$$\hat{p}_{k|k-1} = f(\hat{p}_{k-1|k-1}, v_k), \tag{6a}$$

$$\boldsymbol{P}_{k|k-1} = \boldsymbol{F}_k \boldsymbol{P}_{k-1|k-1} \boldsymbol{F}_k^T + \boldsymbol{Q}_k. \tag{6b}$$

Update:

$$\tilde{y}_k = \boldsymbol{z}_k - h(\hat{p}_{k|k-1}), \tag{7a}$$

$$S_k = \boldsymbol{H}_k \boldsymbol{P}_{k|k-1} \boldsymbol{H}_k^T + R_k, \tag{7b}$$

$$\boldsymbol{K}_k = \boldsymbol{P}_{k|k-1} \boldsymbol{H}_k^T S_k^{-1}, \tag{7c}$$

$$\hat{p}_{k|k} = \hat{p}_{k|k-1} + \boldsymbol{K}_k \tilde{y}_k, \tag{7d}$$

$$\boldsymbol{P}_{k|k} = (\boldsymbol{I} - \boldsymbol{K}_k \boldsymbol{H}_k) \boldsymbol{P}_{k|k-1}, \tag{7e}$$

where $\hat{p}_{k|k-1}$ and $\hat{p}_{k|k}$ are predicted and updated state respectively, $\boldsymbol{P}_{k|k-1}$ and $\boldsymbol{P}_{k|k}$ are predicted and updated covariance of state respectively. $\boldsymbol{K}_k$ is the calculated Kalman gain. $h(p_k)$ is the function to calculate the ESDF value, $\boldsymbol{H}_k$ is the Jacobian of $h(p_k)$ at $\hat{p}_{k|k-1}$, $\boldsymbol{F}_k$ is the Jacobian of $f(\hat{p}_{k-1|k-1}, v_k)$ at $(\hat{p}_{k-1|k-1}, v_k)$.

When the robot collides with an obstacle, an observation that $z_k = 0$ can be obtained. This method will give the updated state. If no collision, we use the prediction result as the estimated result.

Notice, the covariance matrix contains the information of position uncertainty, which can be represented as an ellipse. The two eigenvectors represent the direction of the major-axis and the minor-axis. The arithmetic square roots of the two eigenvalues represent half the length of each axis.

IV. Active Collision Planning

To further exploit collisions to facilitate localization, we propose a collision-oriented planning algorithm *bounce planner*, to actively trigger collisions while the robot is moving. This algorithm aims to find a path from a starting position to a goal position, with additional constraints on the estimated position uncertainty. With this algorithm, once the prediction of position uncertainty is nearly to exceed a given threshold, active collisions are planned to generate collision observations by building a collision roadmap probabilistically and then searching for a path on it. The proposed planning method is presented in detail in Alg. 1.

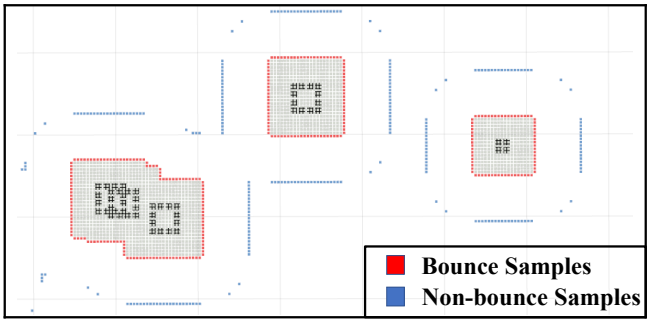

Fig. 4. Visualization of the collision roadmap after batch sampling .

A. Batch Sampling for Collision Roadmap

Our planner is required to accurately generate collision points on the boundary of obstacles, thus highly relies on a fine map resolution, which may consume unaffordable onboard computational resources. To cut off the calculation time for real-world usage, we firstly sample the ESDF map to build a simplified connected graph $\mathcal{G}$ consisting of bounce samples and non-bounce samples, as shown in Fig. 4. Here bounce samples are extracted from the surfaces of obstacles, which are classified by zero ESDF value. Non-bounce samples in free space are necessary to connect waypoints and form a path. To select proper coordinates, we build the map skeleton by grouping samples which have a certain distance margin value to obstacles. In our implementation, we set this margin value to 0.6m to balance sample number and path completeness. These samples compose a nodes set, along with the start and goal coordinates. Finally, nodes with distances lower than a range threshold are connected, and are determined at the following search step.

B. Roadmap Pruning under Uncertainty Constraint

The above-mentioned roadmap connects sampling rigorously, based on a range threshold. We then prune the roadmap based on safety and uncertainty criteria. Firstly, the safety of connections is checked to ensure path feasibility, thus all connections that intersect with obstacles are deleted from the roadmap. Moreover, to facilitate the collision-aided state estimation, the roadmap requires further pruning considering the uncertainty of localization. To this end, we need to calculate the uncertainty magnitude of each node to quantify the localization quality when the robot arrives at this node. As illustrated in Sec. III-C, we use the maximum eigenvalue λ_0 of the propagated state covariance matrix to quantify the position uncertainty. Here, we assume that the robot moves at a constant velocity in a straight line l between two nodes to approximate an arrival time for state propagation. We perform the prediction as Eq. 6b at a fixed rate and get the propagated covariance matrix. Since no collision updates along this movement, the position uncertainty along l keeps growing. The propagated position

Algorithm 1: Bounce Planner

Input: start state s, goal state g, uncertainty threshold λ, ESDF map $\mathcal{M}$, search range r
Output: a path $\mathcal{P}$

initialize the graph $\mathcal{G}$, build kd-tree $tree$;
initialize the priority queue $\mathcal{Q} \leftarrow \varnothing$, $\mathcal{Q}.pushback(s)$;
while $!\mathcal{Q}.empty()$ **do**
  $n_{cur} \leftarrow \mathcal{Q}.top(), \mathcal{Q}.pop()$;
  $n_{cur}.state \leftarrow EXPANDED$;
  if $findGoal(n_{cur})$ **then**
    $\mathcal{P} \leftarrow retracePath(n_{cur})$;
    return $\mathcal{P}$;
  $neighbours \leftarrow tree.SearchInRange(n_{cur}, r)$;
  foreach *node* n *in* $neighbours$ **do**
    if *node* n *has not been expanded* **then**
      `// safety check`
      if $!EdgeValid(n_{cur}, n)$ **then** Continue ;
      $Un_{temp} \leftarrow$ propagated uncertainty as no collision;
      `// uncertainty constraint`
      if $Un_{temp} \geq \lambda$ **then** Continue ;
      if n *is bounceSample* **then**
        `// collision update`
        $Un_{temp} \leftarrow bounceUpdate(n)$;
      if n *is not in* $\mathcal{Q}$ **then**
        $n.parent \leftarrow n_{cur}$;
        $n.uncertainty \leftarrow Un_{temp}$;
        $\mathcal{Q}.pushback(n)$;
      else
        $gScore_{temp} \leftarrow n_{cur}.gScore + cost(n_{cur}, node)$;
        if $gScore_{temp} < n.gScore$ **then**
          $n.changeParent(n_{cur})$;
          $n.uncertainty \leftarrow Un_{temp}$;
return $\mathcal{P}$

uncertainty at the end of l is checked given an uncertainty threshold λ, and connections with unacceptable uncertainty are deleted from the roadmap. Finally, if the end node of l is a bounce sample, we update its uncertainty according to Eq. 7a to Eq. 7e. After these steps a collision roadmap that quantifies our collision-oriented estimator is established, and a shortest path can be easily searched by typical path searching methods such as A*.

V. Experiments

A. Implementation Details

Simulation and real-world experiments are conducted to validate the effectiveness and efficiency of our methods. We deploy our localization method on an omnidirectional McNamee-based wheeled robotAll computations are performed by an onboard computer NUC with an i5-1135G7 CPU. An ouster LiDAR OS0-64-U is used to build the prior point cloud map. After filtering the noise, we transform the point cloud to a 2D voxel map. Then we inflate obstacles based on the radius of the robot to get the feasible region of the robot's centroid, from which we build the ESDF map. We use the NOKOV motion capture system to provide the ground truth of localization. The localization of our methods only uses IMU and wheel odometers.

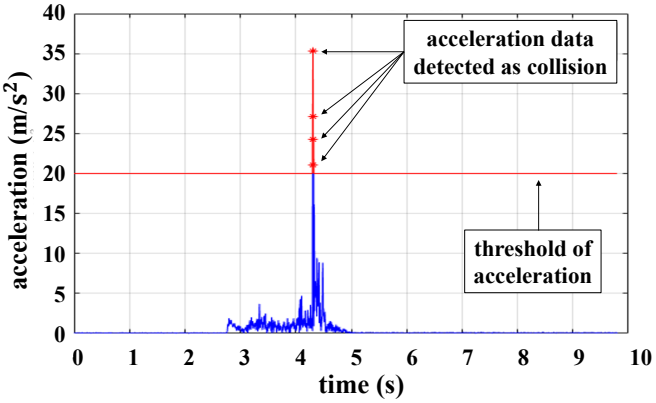

Fig. 5. A typical process of acceleration magnitude changes at XY-plane during a collision. This figure shows the robot colliding with an obstacle and then coming to a stop.

B. Collision Detection

An accurate collision detection strategy is vital for the proposed collision-based estimation algorithm. Since IMU is very sensitive to collision events, in this paper, we use the IMU on the robot to detect collision, which is convenient and effective. A typical collision process is shown in Fig. 5. Without collision, the robot runs normally and the acceleration stays under a certain threshold. Once collides, the acceleration increases rapidly and soon breaks the threshold. Then it decreases greatly in a very short time, after several oscillations it recovers to the normal state. The whole collision process lasts about 20ms.

C. Real World Localization Experiments

The size of the prior voxel map is 17.6 x $12.4m$. Then we manually remote the robot to randomly collide with surrounding obstacles in the environment. The robot starts at a fixed position with a predefined orientation to ensure the initial pose is known. At the end of the trajectory, we remote the robot back to the start position. The whole trajectory is of length of $47.8m$ and lasts $108s$, including 7 collision events. We make comparisons with Collision-PF, Collision-EKF, the wheel odometry and the ground truth.

As Fig. 6 shows, the position error accumulates with time growing before collision for all three trajectories. Once collision happens, both Collision-EKF and Collision-PF exploit the collision information to update the current position, correcting the position to the surface of obstacles instead of dashing into them. For Collision-PF, the first collision is not detected since the acceleration magnitude is lower than the given threshold. Without a collision update, its trajectory

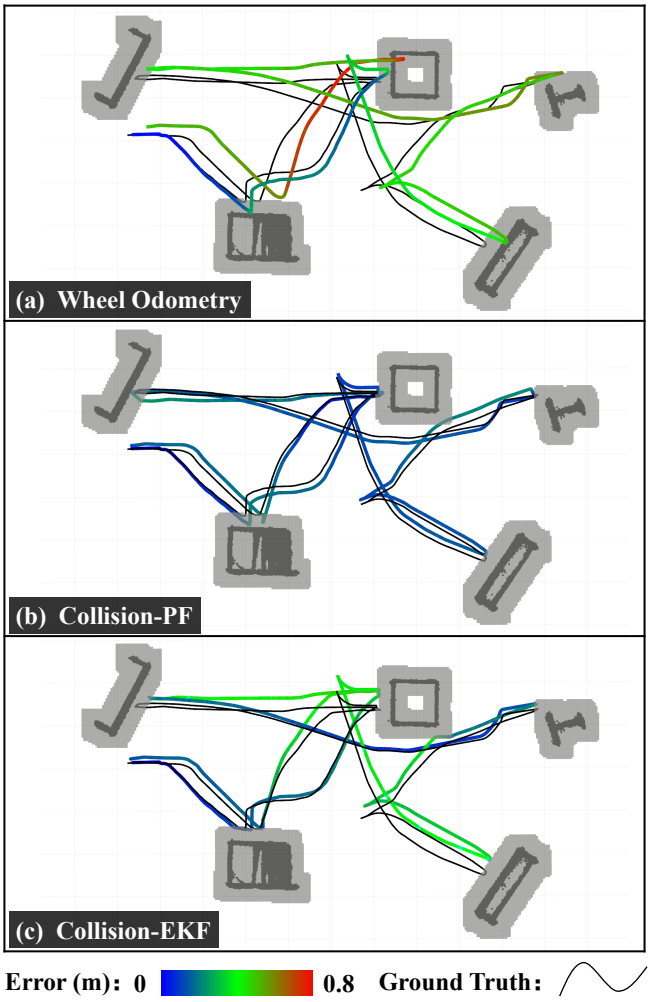

Fig. 6. Snapshots of trajectories in localization experiments. (a) The trajectory of wheel odometry without exploiting collision information. (b) The trajectory of collision-aided particle filter. (c) The trajectory of collision-aided extended Kalman filter.

dashes into the obstacle at first collision. However, the exploitation of latter collisions successfully reduces the position error, preventing the trajectory from huge divergence.

Tab. I shows that collision-aid localization methods have smaller absolute position error. Collision information truly improves the localization performance.

TABLE I
ABSOLUTE POSITION ERROR COMPARISON

Method	RMSE	Mean	STD	Min	Max
Wheel Odometry	0.431	0.391	0.181	0.000	0.779
Collision-PF	**0.150**	**0.137**	**0.063**	0.000	**0.308**
Collision-EKF	0.272	0.235	0.137	0.000	0.573

D. Verification of Bounce Planner in Simulation

To verify the effectiveness of the proposed active collision planning algorithm, we build a random map with cluttered obstacles in RViz. The map is 22 x 22m with a resolution of $0.04m$. We assume that the robot moves at a speed of $0.3m/s$. Red points in Fig. 7 are bounce samples at the

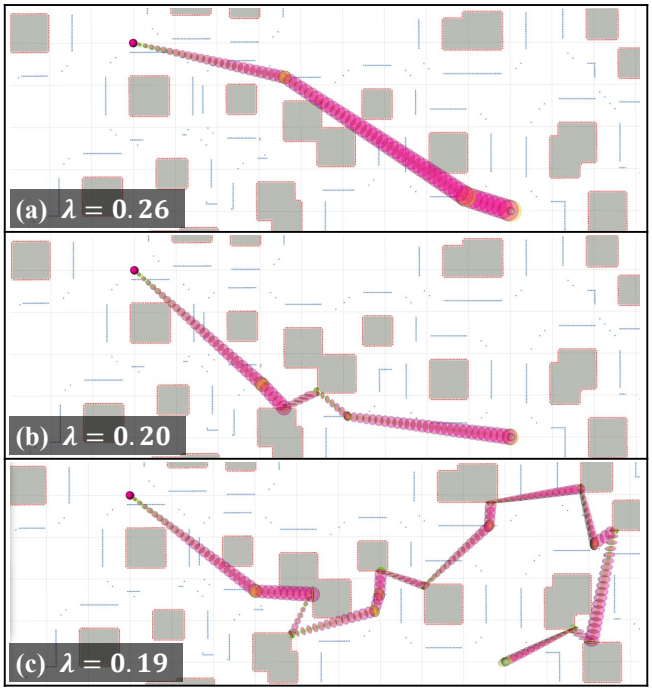

Fig. 7. Results of active collision planning with different uncertainty thresholds. Each grid in the map is 1 x 1m. (a) Planned path when $\lambda = 0.26m$ without collision. (b) Planned path when $\lambda = 0.20m$ which contains two collisions with obstacles. (c) Planned path when $\lambda = 0.19m$ which contains nine collisions with obstacles.

surface of obstacles while blue points are non-bounce samples which have an ESDF value of $0.6m$. We set uncertainty threshold λ with different values. The search range for kd-tree is $15m$. The path starts from $(-5m, 1m)$ to $(4m, -3m)$.

As is shown in Fig. 7, our active collision planning algorithm successfully finds a path from start to goal under the given position uncertainty threshold. For Fig. 7(a), $\lambda = 0.26m$ is large enough, so the path does not need collision. The position uncertainty at the goal position is $0.259m$. As for Fig. 7(b) when λ gets smaller, it is impossible to find a path without collision. So the result path contains two collisions to reduce the position uncertainty of intermediate nodes. The position uncertainty at the goal for the second path is $0.198m$. In Fig. 7 (c), λ reaches even smaller, the result has to add more collisions to satisfy the uncertainty constraint. The uncertainty of the goal is $0.176m$. These results prove that planned collision improves the position uncertainty and validates the effectiveness of our method. Besides, the smaller the uncertainty threshold is, the more the collision contained in the planned path.

VI. CONCLUSION AND FUTURE WORK

In this paper, we show the possibility of reliable robot navigation equipped with only proprioceptive sensors leveraging the collision information. Besides, we propose an active collision path planning method, which reduces the localization uncertainty during navigation. In the future, we will apply our framework to aerial robots and design a collision-degeneration-degree evaluation method, which will make our framework more certifiable.

REFERENCES

[1] T. Qin, P. Li, and S. Shen, "Vins-mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.

[2] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint kalman filter for vision-aided inertial navigation," in *Proceedings 2007 IEEE international conference on robotics and automation*. IEEE, 2007, pp. 3565–3572.

[3] W. Xu and F. Zhang, "Fast-lio: A fast, robust lidar-inertial odometry package by tightly-coupled iterated kalman filter," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 3317–3324, 2021.

[4] W. Xu, Y. Cai, D. He, J. Lin, and F. Zhang, "Fast-lio2: Fast direct lidar-inertial odometry," *IEEE Transactions on Robotics*, vol. 38, no. 4, pp. 2053–2073, 2022.

[5] J. Zhang and S. Singh, "Loam: Lidar odometry and mapping in real-time." in *Robotics: Science and systems*, vol. 2, no. 9. Berkeley, CA, 2014, pp. 1–9.

[6] Y. Mulgaonkar, W. Liu, D. Thakur, K. Daniilidis, C. J. Taylor, and V. Kumar, "The tiercel: A novel autonomous micro aerial vehicle that can map the environment by flying into obstacles," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 7448–7454.

[7] P. De Petris, H. Nguyen, M. Dharmadhikari, M. Kulkarni, N. Khedekar, F. Mascarich, and K. Alexis, "Rmf-owl: A collision-tolerant flying robot for autonomous subterranean exploration," in *2022 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2022, pp. 536–543.

[8] A. Briod, P. Kornatowski, J.-C. Zufferey, and D. Floreano, "A collision-resilient flying robot," *Journal of Field Robotics*, vol. 31, no. 4, pp. 496–509, 2014.

[9] M. Mote, M. Egerstedt, E. Feron, A. Bylard, and M. Pavone, "Collision-inclusive trajectory optimization for free-flying spacecraft," *Journal of Guidance, Control, and Dynamics*, vol. 43, no. 7, pp. 1247–1258, 2020.

[10] J. Zha and M. W. Mueller, "Exploiting collisions for sampling-based multicopter motion planning," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 7943–7949.

[11] Y. Mulgaonkar, A. Makineni, L. Guerrero-Bonilla, and V. Kumar, "Robust aerial robot swarms without collision avoidance," *IEEE Robotics and Automation Letters*, vol. 3, no. 1, pp. 596–603, 2017.

[12] R. Buchanan, J. Bednarek, M. Camurri, M. R. Nowicki, K. Walas, and M. Fallon, "Navigating by touch: haptic monte carlo localization via geometric sensing and terrain classification," *Autonomous Robots*, vol. 45, no. 6, pp. 843–857, 2021.

[13] X. Wu and M. W. Mueller, "Using multiple short hops for multicopter navigation with only inertial sensors," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 8559–8565.

[14] C. Liu and R. Tron, "Sensing via collisions: a smart cage for quadrotors with applications to self-localization," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 4033–4039.

[15] T. Lew, T. Emmei, D. D. Fan, T. Bartlett, A. Santamaria-Navarro, R. Thakker, and A.-a. Agha-mohammadi, "Contact inertial odometry: Collisions are your friends," in *The International Symposium of Robotics Research*. Springer, 2019, pp. 938–958.

[16] S. A. Sadat, K. Chutskoff, D. Jungic, J. Wawerla, and R. Vaughan, "Feature-rich path planning for robust navigation of mavs with mono-slam," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 3870–3875.

[17] R. Takemura and G. Ishigami, "Perception-aware receding horizon path planning for uavs with lidar-based slam," in *2022 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*. IEEE, 2022, pp. 1–8.

[18] Z. Zhang and D. Scaramuzza, "Beyond point clouds: Fisher information field for active visual localization," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 5986–5992.

[19] J. Zhang, M. Kaess, and S. Singh, "On degeneracy of optimization-based state estimation problems," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 809–816.

[20] P. Salaris, R. Spica, P. R. Giordano, and P. Rives, "Online optimal active sensing control," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 672–678.

[21] B. Penin, P. R. Giordano, and F. Chaumette, "Minimum-time trajectory planning under intermittent measurements," *IEEE Robotics and Automation Letters*, vol. 4, no. 1, pp. 153–160, 2018.

[22] O. Napolitano, D. Fontanelli, L. Pallottino, and P. Salaris, "Gramian-based optimal active sensing control under intermittent measurements," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 9680–9686.

[23] P. F. Felzenszwalb and D. P. Huttenlocher, "Distance transforms of sampled functions," *Theory of computing*, vol. 8, no. 1, pp. 415–428, 2012.