

บทความวิจัย

การคัดกรองเส้นทางที่ปลอดภัยและการตรวจสอบเสถียรภาพเชิงเรขาคณิตสำหรับการนำทางหุ่นยนต์อัตโนมัติบนภูมิประเทศขรุขระ

ธนกร กุลศรี* คมกมล จงบุญวัฒนา และ อรัญ แบล็กเลอร์

ภาควิชาวิศวกรรมการผลิตและหุ่นยนต์ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ

* ผู้นิพนธ์ประสานงาน โทรศัพท์ 06 5225 1395 อีเมล: tanakornkulsri@gmail.com DOI: 10.14416/j.kmutnb.2026.10.003

รับเมื่อ 12 กรกฎาคม 2569 แก้ไขเมื่อ 10 สิงหาคม 2569 ตอรับเมื่อ 1 กันยายน 2569 เผยแพร่ออนไลน์ 9 ตุลาคม 2569

© 2026 King Mongkut's University of Technology North Bangkok. All Rights Reserved.

บทคัดย่อ

บทความนี้นำเสนอวิธีการคัดกรองการชนและการตรวจสอบเสถียรภาพเชิงเรขาคณิตสำหรับการนำทางอัตโนมัติของหุ่นยนต์ที่ขับเคลื่อนด้วยล้อบนภูมิประเทศขรุขระ แนวทางที่นำเสนอผสานตัวกรองการชนแบบสองขั้นตอนและการตรวจสอบเสถียรภาพโดยอาศัยจุดศูนย์กลาง (Center of Gravity; CoG) เข้ากับตัววางแผนเส้นทางเฉพาะที่แบบมอชันพริมิทีฟ (Motion-primitive Local Planner) ในขั้นตอนแรก ตัวอย่างสิ่งกีดขวางจากแผนที่ระดับความสูงแบบ 2.5 มิติ จะถูกเก็บรวบรวมภายในรัศมีการค้นหาล้อมรอบตัวหุ่นยนต์ ในขั้นตอนที่สอง ตัวอย่างแต่ละจุดจะถูกฉายเข้าสู่เฟรมพิกัดของพริมิทีฟตัวเลือกแล้วตรวจสอบกับตารางความสอดคล้องของวอกเซล (Voxel Correspondence Table) ที่คำนวณล่วงหน้าเพื่อคัดพริมิทีฟที่ถูกกีดขวางเกินค่าขีดแบ่งออก พริมิทีฟที่เหลือจะเข้าสู่การตรวจสอบเสถียรภาพเชิงสถิติแบบจุดต่อจุดตามเส้นทาง โดยประมาณระนาบรองรับได้มุมทั้งสี่ของหุ่นยนต์และปฏิเสธพริมิทีฟหากระนาบเอียงเกินขีดจำกัด หรือระยะห่างระหว่างจุดศูนย์กลางกับระนาบรองรับเกินค่าขีดแบ่งการตกหล่น วิธีการนี้ได้รับการตรวจสอบในสถานการณ์การหลบหลีกหลุม ทั้งในการจำลองด้วย Gazebo และในห้องปฏิบัติการจริง โดยใช้หุ่นยนต์ขับเคลื่อนน้ำหนัก 77 กิโลกรัม ติดตั้งไลดาร์ Livox MID-360 สองตัว อัตราความสำเร็จเพิ่มขึ้นจาก 80% เป็น 100% ในการจำลอง และ ได้ 100% ในการทดลองจริง อัตราส่วนความปลอดภัยต่อผนังเพิ่มขึ้นจาก 92.55% เป็น 99.50% ในการจำลอง และ 98.94% ในสภาพแวดล้อมจริง โดยมีต้นทุนเพียงเล็กน้อย คือ ความยาวเส้นทางเพิ่มขึ้น 2.4% และเวลาเดินทางเพิ่มขึ้น 2.2%

คำสำคัญ: การคัดกรองการชน การตรวจสอบเสถียรภาพเชิงสถิติ การนำทางอัตโนมัติ หุ่นยนต์ที่ขับเคลื่อนด้วยล้อ

การอ้างอิงบทความ: ธนกร กุลศรี, คมกมล จงบุญวัฒนา และ อรัญ แบล็กเลอร์, “การคัดกรองเส้นทางที่ปลอดภัยและการตรวจสอบเสถียรภาพเชิงเรขาคณิตสำหรับการนำทางหุ่นยนต์อัตโนมัติบนภูมิประเทศขรุขระ,” วารสารวิชาการพระจอมเกล้าพระนครเหนือ, ปีที่ 37, ฉบับที่ 2, หน้า 1–15, เม.ย.-มิ.ย. 2570. เลขที่บทความ 272-018363, doi: 10.14416/j.kmutnb.2026.10.003.



Research Article

Geometric-Based Collision Filtering and Stability Check for Autonomous Robot Navigation in Uneven Terrain

Tanakorn Kulsri*, Komkamol Chongbunwatana and Aran Blattler

Department of Production and Robotic Engineering, Faculty of Engineering, King Mongkut's University of Technology North Bangkok, Bangkok, Thailand

* Corresponding Author, Tel. 06 5225 1395, E-mail: tanakornkulsri@gmail.com DOI: 10.14416/j.kmutnb.2026.10.003

Received 12 July 2026; Revised 10 August 2026; Accepted 1 September 2026; Published online: 9 October 2026

© 2026 King Mongkut's University of Technology North Bangkok. All Rights Reserved.

Abstract

This paper presents a geometric-based collision-filtering and stability-checking method for autonomous navigation of a tracked rescue robot over uneven terrain. The proposed approach integrates a two-phase collision filter and a Center-of-Gravity (CoG)-based stability check into a motion-primitive local planner. In the first phase, obstacle samples from a 2.5D elevation map are collected within a search radius around the robot. In the second phase, each obstacle sample is projected into each candidate primitive's body frame and tested against a precomputed voxel correspondence table; primitives whose blocked-cell count exceeds a threshold are discarded. The surviving primitives then undergo waypoint-by-waypoint static stability verification: the support plane beneath each of the robot's four corners is estimated, and a primitive is rejected if the plane's tilt exceeds a tilt limit or the gap between the CoG and the support plane exceeds a fall threshold. The method was validated in a hole-avoidance scenario in both Gazebo simulation and a real-world laboratory environment, using a 77 kg tracked robot equipped with dual Livox MID-360 LiDAR sensors. With the collision and stability checks enabled, the success rate improved from 80% to 100% in simulation and reached 100% in real-world trials. The wall safety ratio increased from 92.55% to 99.50% in simulation and reached 98.94% in the real-world environment, at a marginal efficiency cost of 2.4% in path length and 2.2% in travel time.

Keywords: Autonomous Navigation, Collision Filtering, Static Stability Verification, Tracked Rescue Robot

Please cite this article as: T. Kulsri, K. Chongbunwatana, and A. Blattler, "Geometric-based collision filtering and stability check for autonomous robot navigation in uneven terrain," *The Journal of KMUTNB*, vol. 37, no. 2, pp. 1–15, Apr.–Jun. 2027 (in Thai), Art. no. 272-018363, doi: 10.14416/j.kmutnb.2026.10.003.

1. Introduction

Autonomous mobile-robot systems are deployed more broadly across industrial and service domains, including coordinated multi-robot material handling [1]. The iRAP Robot research group at King Mongkut's University of Technology North Bangkok (KMUTNB) competes regularly in the RoboCup Rescue Robot League. The team's tracked platform (Figure 1) carries active front and rear flippers and dual LiDAR sensors for 3D perception. The competition arena consists of elevated wooden pallets separated by gaps emulating holes, crevices, and drop-offs. This terrain is dangerous to the robot if its tracks lose ground contact. Geometric configuration-space methods have been applied to tracked rescue robots to plan coordinated body and flipper motions over such terrain [2].

The global planner uses Jump Point Search (JPS), which preserves grid-optimality while expanding far fewer nodes than A* [3] and, unlike sampling-based planners [4], exploits the uniform-cost grid; graph-based planners have similarly been used for aerial platforms in hazardous environments [5]. At the local level, motion-primitive planners sample candidate trajectories and select the lowest-cost collision-free option. The local planner here builds on FALCO [6], adapted from an aerial vehicle to a

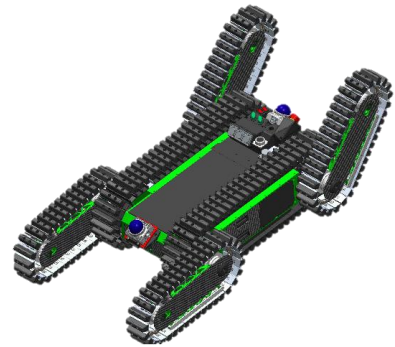


Figure 1 The tracked rescue robot developed by the iRAP team.

tracked platform, the details of which are given in Sections 2.3 and 2.4; FASTER [7] solves a mixed-integer program over convex free-space polyhedra. Both rely on geometric collision checks but do not verify that the robot's body remains statically stable along the path. Traversability-aware approaches instead assess whether terrain is navigable, not merely obstacle-free: PUTN [8], TNT [9], IN-Sight [10], and elevation-map planners for legged robots [11] all treat traversability as a soft cost that biases the planner rather than a hard safety constraint. Classical tip-over measures such as the force-angle stability margin [12] evaluate the center of mass against the support polygon but require accurate contact-point and inertial estimates. None perform explicit waypoint-level, CoG-based stability verification, so a geometrically collision-free path can still cause tip-over on holes and sharp edges.

This paper addresses that gap with an autonomous navigation framework that integrates three components: a JPS global planner that produces a coarse waypoint path, a motion-primitive local planner that generates candidate trajectories between waypoints, and a waypoint-level CoG stability verification that accepts or rejects each candidate. The central idea is to combine two approaches that are usually traded off against each other. Motion-primitive collision filtering is fast because it treats the robot as a particle against an obstacle map, but that same abstraction ignores the robot's physical extent, so a path can be collision-free yet leave part of the body unsupported. The proposed method retains this speed while adding an explicit rigid-body stability check at each waypoint, made tractable in real time through a precomputed voxel-correspondence table. The result enforces stability as a hard accept/reject constraint rather than a soft cost, gaining full-footprint safety without sacrificing the real-time performance of primitive-based planning.

2. Materials and Methods

2.1 System Overview

The navigation system follows a three-layer architecture, illustrated in Figure 2. The global layer converts the on-board 3D LiDAR

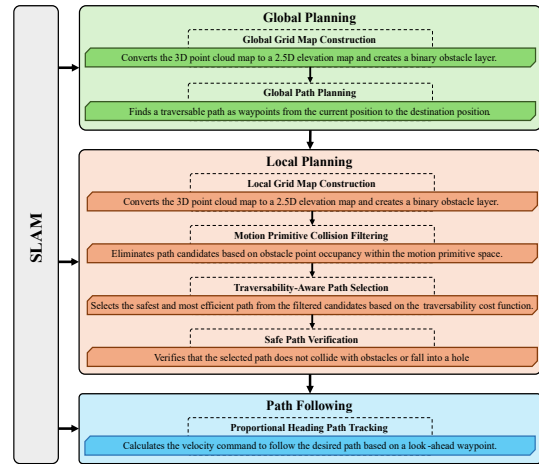


Figure 2 Three-layer system architecture.

point cloud into a 2.5D elevation grid with a binary obstacle layer and uses a Jump Point Search planner to compute a coarse waypoint path from start to goal. The local layer receives the next global waypoint, rebuilds the terrain at greater detail within a window around the robot, and evaluates a library of 2,744 motion primitives (343 base shapes $\times$ 8 heading rotations) to select a collision-free, statically stable local path. The path-following layer executes the selected primitive with a proportional-heading controller at a maximum speed of 0.3 m/s. Localization and mapping are assumed and treated as out of scope. Odometry is provided by Point-LIO tightly-coupled LiDAR-IMU fusion. The collision filtering and stability verification described in Sections 2.4 and 2.5 are the core contributions of this work and reside in the local planning layer.

2.2 Terrain Representation

The terrain is represented as a 2.5D elevation grid [13] with fixed cell resolution Δ , updated from dual Livox MID-360 scans. Each cell takes the maximum height of the points falling inside it; empty cells are filled from their valid neighbors, and a median filter suppresses sensor noise.

A binary obstacle layer is then derived cell by cell: for each cell C , the detector flags it as occupied if any of its eight neighbors $h_1, \dots, h_n$ within radius r_n satisfies Equation (1).

$$\begin{aligned} O(i, j) &= \text{occupied}; \\ \text{if any } |C - h_k| &> h_{\text{Thre}}, \\ k &\in \{1, \dots, n\} \end{aligned} \quad (1)$$

Where h_{Thre} is a configurable height-difference threshold and r_n is one cell width, so the eight neighbors form the immediate ring around C . The test catches the top edges of walls and pillars. Cells are further classified as Traversable, Hole, or Wall against absolute height thresholds, supporting the point classification used in Section 3.

2.3 Motion-Primitive Library

The local planner does not search free space directly; it selects from a fixed library of motion primitives generated offline. Each primitive is a cubic spline defined by four control

points in radius-shift coordinates spaced by an interval d_{cp} along the radius axis. Seven discrete shift values at each of three control points give $7^3 = 343$ base paths. The shift range narrows by a proportional scale $s < 1$ at each successive point $(\pm\theta_1, \pm s\theta_1, \pm s^2\theta_1)$ to produce the library's flower-petal taper (Figure 3). The angular spread of the paths is set by the robot's maximum yaw rate and their length by its maximum linear speed, so the library reflects the platform's motion capability. At lookup the library is rotated by eight 45° steps to span 360° , expanding it to $343 \times 8 = 2,744$ primitives, applied programmatically rather than stored.

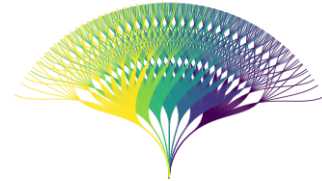


Figure 3 Motion primitives, one panel per direction group.

2.4 Two-Phase Collision Filtering

Collision filtering relies on a voxel-correspondence table computed offline. The robot's local space is discretized at a fixed voxel resolution Δv , extending W_v cells in width and L_v cells in length ($N_v = W_v \times L_v$). The resolution is set as fine as the onboard real-time budget allows and to match the resolution of the SLAM map that feeds it; in this work $\Delta v = 0.05$ m. The

region size is not tuned but follows directly from the primitive library: W_v spans the lateral angular spread of the paths, and L_v their forward reach, so the table covers exactly the space the primitives can occupy. The table is built by iterating over every vertex of every primitive: for each vertex the routine finds the nearest voxel and appends the primitive's index to that voxel's correspondence list, so that after construction each voxel holds the primitives whose path passes through it (Figure 4).

The runtime filter operates in two phases. In Phase A (obstacle-sample collection), the planner scans the local elevation grid within a search radius r_s centered on the robot and collects all obstacle cells into a sample list. The radius r_s acts like the inflation layer of a conventional planner, but is deliberately kept small, set near the robot's radius and then reduced so that it does not block narrow passages; because the surviving primitives are re-checked by the stability module (2.5), the collision filter need not be conservative on its own. In Phase B (voxel-correspondence check), each obstacle sample is rotated into the body frame of every candidate primitive, the corresponding voxel is looked up, and a blocked counter is incremented for every primitive in that voxel's list, avoiding per-primitive point-cloud tests.

Cells with no valid height are added to the voxel-occupancy list but not to the obstacle-sample list, so unobserved cells do not affect the blocking decision. A primitive whose blocked count exceeds a threshold N_{block} is then discarded. Setting $N_{block} = 0$ is the most conservative choice: a primitive is retained only if no obstacle sample falls within any of its occupied voxels; a value above 0 mainly absorbs mapping or SLAM noise, and is unnecessary when that noise is negligible. Figure 5 shows the candidate set before and after Phase B pruning.

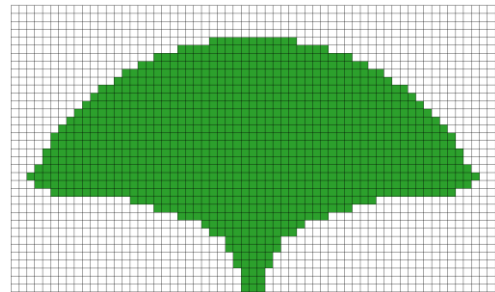


Figure 4 Voxel correspondence table.

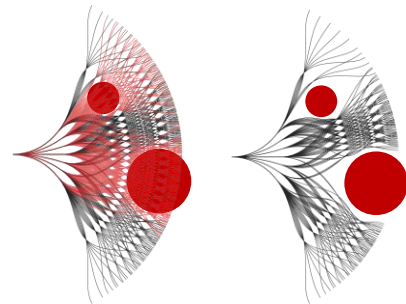


Figure 5 Phase B collision filtering, before and after.

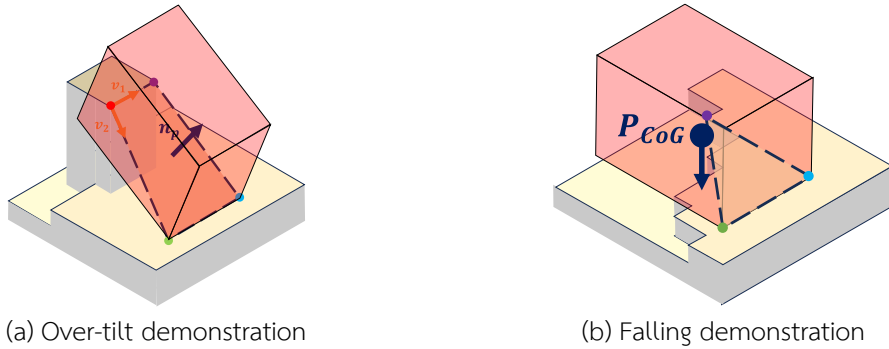


Figure 6 Over-tilt and fall rejection cases, isometric view.

2.5 Static-Stability Verification

Primitives that survive the collision filter undergo a waypoint-by-waypoint static-stability check. At each waypoint, the robot's rectangular footprint ($L_b \times W_b$) is divided into four corner regions (front-left, front-right, rear-left, rear-right). For each region, the system queries the elevation grid and selects the highest terrain sample as that region's contact point. A support plane is fitted to the three highest of the four contact points (the lowest is discarded) by computing the normal vector of Equation (2).

$$\hat{n}_p = \frac{\mathbf{v}_1 \times \mathbf{v}_2}{\|\mathbf{v}_1 \times \mathbf{v}_2\|} \quad (2)$$

Where $\mathbf{v}_1$ and $\mathbf{v}_2$ are vectors from the lowest of the three retained contact points to the other two. The plane is written as $\hat{n}_p \cdot (\mathbf{x} - \mathbf{c}_a) = 0$, where $\mathbf{c}_a$ is a point on the plane.

A primitive is rejected at a waypoint if either of two conditions is met. First, if the tilt

of any corner's support plane exceeds a threshold T_{ilt} equivalently, if the cosine of $\hat{n}_p$ with the world-up vector falls below $\cos T_{ilt}$ the terrain is judged too steep for stable traversal. Second, the robot's CoG, taken as the geometric center of the bounding box, is projected vertically onto the support plane and the terrain height beneath the projected point is looked up; if the vertical gap between the CoG projection and the terrain exceeds a fall threshold T_{fall} , the robot would lose ground contact at that corner, indicating a hole or drop-off. The tilt limit reflects the maximum slope the tracks can grip, and the fall threshold is set to the maximum height drop the robot can bridge without its tracks lifting off. Figure 6 illustrates the over-tilt rejection case (a) and the fall rejection case (b).

After both the collision filter and the stability check, the surviving primitives are ranked by directional cost toward the goal waypoint,



and the lowest-cost primitive is forwarded to the path-following controller.

3. Results and Discussion

3.1 Experimental Setup

Experiments were conducted in a Hole-Avoidance scenario in both Gazebo simulation and a real-world laboratory, using the tracked robot specified in Table 1. Across all trials, the flippers were kept in a fixed configuration, so the reported results reflect body-motion planning alone rather than active flipper articulation. The Hole-Avoidance scenario was selected because it simultaneously stresses both mechanisms the planner contributes: collision filtering against wall-like obstacles and stability verification against unsupported edges and drops. Terrains such as slopes, loose gravel, and staircases primarily test locomotion over uneven ground, which for this platform depends on active flipper articulation [14]; with the flippers held fixed, evaluating them would measure a capability outside the scope of this work. Planning decisions are geometric and static by design, made on the elevation-grid representation without modelling dynamic effects such as friction; physics-engine parameters therefore lie outside the method's assumptions.

The simulation ran in Gazebo on a desktop

workstation, with track friction and flipper joints modelled through the SimpleTrackedVehiclePlugin and a mimic-joint plugin. The simulation arena measured approximately 18×12 m with elevated surfaces separated by rectangular holes (Figure 7). The real-world arena measured 12×7 m and was constructed from nine wooden pallets ($1.1 \times 1.1 \times 0.12$ m each) arranged to create traversable surfaces separated by gaps (Figure 8); odometry was provided by Point-LIO rather than track encoders.

Each experiment consisted of 10 forward-and-backward repetitions (20 legs) between fixed start and goal positions. A leg was classified as failed on a 120 s timeout, an over-tilt exceeding 30° , or, in the real-world trials, safety-operator intervention. In simulation, three conditions were evaluated by enabling or disabling the two safety mechanisms independently. The first (Enable Collision, Disable Stability) selects the lowest-cost primitive passing the collision filter without any stability check; this corresponds to FALCO with a 2.5D gridmap input adapted for a grounded robot, in which unknown cells are treated as obstacles so paths stay within traversable area. The second (Disable Collision, Enable Stability) isolates the stability check without geometric pre-filtering. The third (Enable Collision, Enable

Stability) is the proposed method. All three share identical start and goal positions, primitive library, and terrain (Figure 7).

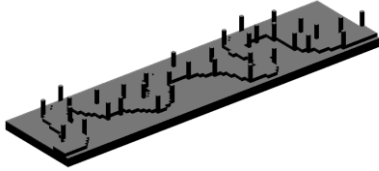


Figure 7 Gazebo simulation arena for the Hole-Avoidance scenario.



Figure 8 Real-world laboratory arena for the Hole-Avoidance scenario.

Table 1 Hardware specifications of the iRAP tracked rescue robot.

Component	Specification
Maximum dimensions	1.0 × 0.6 × 0.6 m
Weight	77 kg
Locomotion	Tracked drive with 2 active flippers
Maximum commanded speed	0.3 m/s
3D LiDAR sensors	2 × Livox MID-360
IMU	Integrated within the Livox MID-360

Because the baseline shares the same platform, sensor configuration, primitive library,

and arena as the proposed method, the FALCO-3D condition serves as a controlled geometry-only benchmark that isolates the contribution of the stability check. Rather than re-implementing planners designed for different platforms, such as FALCO [6] and FASTER [7] for aerial vehicles and PUTN [8] for wheeled robots on outdoor slopes, which would confound the comparison with platform and tuning differences, we adapt the representative geometry-only strategy onto the identical tracked platform, so that any performance difference is attributable to the safety mechanisms rather than the underlying planner.

Three Safety Ratio variants are reported. The Hole and Wall Safety Ratios give the proportion of footprint poses free of hole cells and wall cells, respectively. The Total Safety Ratio sums the hole-unsafe and wall-unsafe pose counts over all footprint poses; a pose violating both conditions contributes to both terms, which is why the Total may fall below either component. An unsafe pose does not itself constitute a failed leg: a leg fails only on timeout, over-tilt, or safety-operator intervention.

3.2 Simulation Results

Table 2 summarizes the three conditions. The proposed method (Enable Collision, Enable Stability) reached 100% versus 80% for the

FALCO-3D baseline and 15% for the ablation with collision filtering disabled. To confirm these rates are not artefacts of the sample size, the 95% confidence intervals [83.9%, 100%] for the proposed method, [58.4%, 91.9%] for FALCO-3D, and [5.2%, 36.0%] for the ablation do not overlap between the proposed method and the ablation. The FALCO-3D failures were collisions into holes or tilts beyond 30° on unsupported edges; the ablation failed far more often because, without collision pre-filtering, the stability check must search a much deeper candidate list each cycle and loses real-time responsiveness. The proposed method achieves this safety gain at only a modest cost in path length and travel time (Table 2). Figure 9 compares the footprint traces across the three conditions: the FALCO-3D baseline (Figure 9(a)) and the ablation (Figure 9(b)) both cross unsupported terrain at the hole edges, whereas the proposed method (Figure 9(c)) stays clear of those edges throughout and routes around them.

Figure 10 reports per-cycle computation time. The FALCO-3D baseline is fastest but unsafe, the stability-only ablation is far slower and loses real-time control, and the proposed method stays within the real-time range while achieving the highest safety, confirming that collision pre-filtering is what makes full- footprint stability checking viable online.

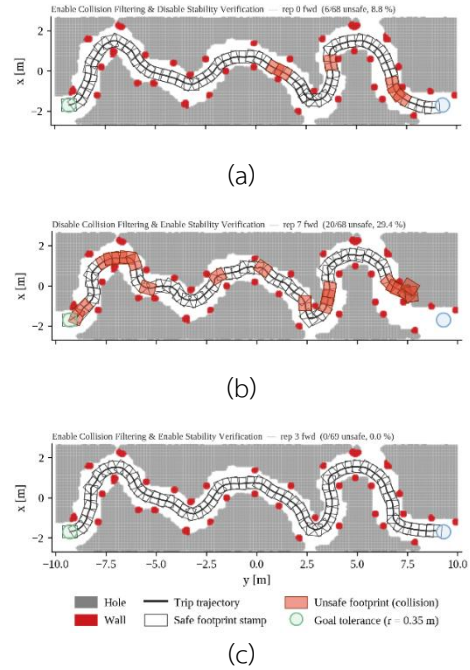


Figure 9 Footprint traces in simulation for the three conditions: (a) FALCO-3D baseline, (b) ablation (stability only), and (c) the proposed method.

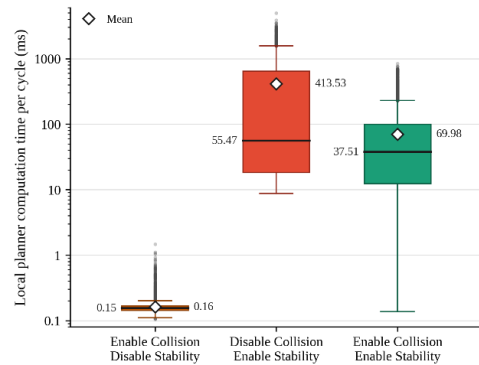


Figure 10 Local-planner computation time per cycle for the three conditions (log scale; box = median and IQR). Measured on the onboard Jetson Orin NX.

Table 2 Simulation results for the Hole-Avoidance scenario (20 legs, mean $\pm$ SD).

	FALCO-3D	Ablation	Proposed
Collision Filtering	✓		✓
Stability Verification		✓	✓
Success rate	16/20 (80%)	3/20 (15%)	20/20 (100%)
Path length (m)	26.43 $\pm$ 0.41	28.45 $\pm$ 0.21	27.07 $\pm$ 0.29
Travel time (s)	63.90 $\pm$ 1.80	93.20 $\pm$ 3.90	65.30 $\pm$ 2.70
Hole Safety (%)	99.99	80.73	99.95
Wall Safety (%)	92.55	78.26	99.50
Total Safety (%)	92.55	58.98	99.46

Table 3 Real-world laboratory results for the Hole-Avoidance scenario (20 legs, mean $\pm$ SD). This FALCO-3D row is a sim-replay reference, not a hardware run.

	FALCO-3D	Proposed
Collision Filtering	✓	✓
Stability Verification		✓
Success rate	-	20/20 (100%)
Path length (m)	-	7.91 $\pm$ 0.33
Travel time (s)	-	35.4 $\pm$ 3.0
Hole Safety Ratio (%)	90.47	91.04
Wall Safety Ratio (%)	72.76	98.94
Total Safety Ratio (%)	63.23	89.98

3.3 Real-World Laboratory Results

The same Hole-Avoidance scenario was replicated on the physical robot in the 12 $\times$ 7 m laboratory. Only the Proposed condition was run on hardware; the FALCO-3D figures in Table 3 are a sim-replay reference computed on the recorded real-world point cloud, because the unsafe condition could not be run on the physical robot without risking damage. This distinction must be kept in mind: the real-world comparison is not a like-for-like hardware experiment.

The robot completed all 20 legs (100% success), consistent with simulation. The Total Safety Ratio of 89.98% is lower than the simulation value of 99.46%, but the two are not comparable in magnitude: the much shorter real-world path yields far fewer footprint samples, so a brief unsafe excursion lowers the ratio more sharply. The meaningful signal is the Wall Safety Ratio. It reached 98.94% for the hardware-proposed run against 72.76% for the sim-replay FALCO-3D reference, a 26-percentage-point gap.

Figure 11(a) shows the Proposed run keeping the footprint clear of the risk regions (hole and wall) for most of the path, with about 9.5% of poses flagged unsafe, whereas the FALCO-3D run in Figure 11(b) leaves roughly 33% of poses unsafe, with contacts occurring frequently along the trajectory.

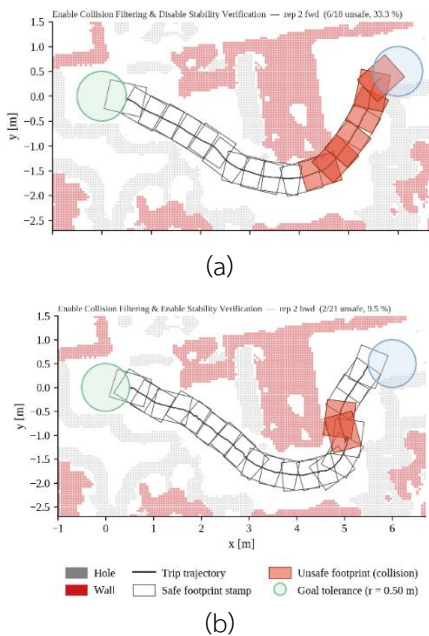


Figure 11 Footprint comparison in the real-world laboratory: (a) Proposed and (b) FALCO-3D (sim-replay reference).

3.4 Discussion

These results can be placed in context against recent uneven-terrain navigation work. The method in [15] solves a nonlinear program over a planar robot-terrain contact model for articulated tracked robots, reporting computation

within 150 ms at 5 Hz. That work adopts the same static assumption used here, arguing that a heavy tracked platform with a large contact area provides sufficient friction for static equilibrium.

The cost of full-body reasoning is the central issue. The planner in [16] operates on a 2.5D grid and notes that the vehicle cannot be treated as a mass point, yet that simulating tip-over per segment is computationally expensive; a gradability threshold is substituted instead. The present work reaches the same conclusion about cost but resolves it by pruning the candidate set first, after which the four-corner support plane and CoG are evaluated directly at every waypoint, at a median cycle of 69.98 ms on the onboard Jetson Orin NX (Figure 10). Without that prefilter, the median cycle rose to 413.53 ms and the success rate fell to 15% (Table 2).

Conceptually closest is CAP [17], which also derives its condition from tip-over analysis on a 2.5D grid. Two differences matter here: CAP assumes the wheels stay in ground contact, whereas losing that contact at a hole edge is the failure mode addressed here; and its capsizing constraint is relaxed into a weighted quadratic pull toward the traversable range, so an unsafe orientation can still be admitted when other terms dominate.

Across [15]–[17] and soft-cost planners such as PUTN [8], the safety condition is either a term inside an optimization or a cheaper proxy, and in both cases an unsafe configuration can survive. For a hole edge, this is insufficient: the tracks either bridge the gap or lose contact. The wall safety ratio rose from 92.55–99.50% because the condition is non-negotiable, at a cost of 2.4% in path length and 2.2% in travel time.

4. Conclusions

This study presented a geometric collision-filtering and static-stability verification method for the autonomous navigation of a tracked rescue robot over uneven terrain. A precomputed voxel-correspondence table prunes infeasible motion primitives, and a waypoint-level check fits support planes to the four footprint corners and rejects primitives that exceed tilt or fall thresholds, all within a motion-primitive local planner. By converting the classical tip-over stability margin into a discrete per-waypoint accept/reject test and adapting the support-plane idea from legged robots to a tracked footprint, the method eliminates tip-over and edge failures that the soft-cost and optimization-embedded formulations leave unaddressed. In a Hole-Avoidance scenario, it raised the success rate

from 80–100% in simulation and completed all 20 legs on a 77 kg physical robot, at an efficiency cost of only 2.4% in path length and 2.2% in travel time; the hardware evaluation ran the full method only, with the disabled-safety baseline obtained by simulation replay rather than a physical run. These results confirm that explicit collision and stability verification can be integrated into a motion-primitive planner at low computational overhead while substantially improving safety.

Several limitations bound these results: the stability model is static and ignores inertial effects (acceptable at 0.3 m/s); only the Hole-Avoidance scenario is evaluated; the real-world disabled baseline is a sim-replay reference rather than a hardware run. Future work will add active flipper control under explicit safety constraints, extend the static stability model to dynamic stability with inertial effects, integrate SLAM-based mapping for larger environments, and evaluate the planner on incline, roughness, and dynamic-obstacle scenarios in addition to hole avoidance.

5. Acknowledgement

This research was supported by King Mongkut's University of Technology North Bangkok (KMUTNB). The authors thank the iRAP Robot team for technical support.



Reference

- [1] P. Chanchaichujit and P. Smithmaitrie, "Management of a multi-robots system for industrial material handling," *The Journal of KMUTNB*, vol. 29, no. 3, pp. 454–464, Jul.–Sep. 2019 (in Thai), doi: 10.14416/j.kmutnb.2019.02.003.
- [2] Y. Yuan, L. Wang, and S. Schwertfeger, "Configuration-space flipper planning for rescue robots," in *Proceedings 2019 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, Würzburg, Germany, 2019, pp. 37–42, doi: 10.1109/SSRR.2019.8848978.
- [3] S. Liu, M. Watterson, K. Mohta, K. Sun, S. Bhattacharya, C. J. Taylor, and V. Kumar, "Planning dynamically feasible trajectories for quadrotors using safe flight corridors in 3-D complex environments," *IEEE Robotics and Automation Letters*, vol. 2, no. 3, pp. 1688–1695, Jul. 2017, doi: 10.1109/LRA.2017.2663526.
- [4] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *International Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, 2011, doi: 10.1177/0278364911406761.
- [5] R. Prasertlux and S. Jiriwibhakorn, "Design and development path planning of unmanned aerial vehicles for stealth missions in dangerous environments," *The Journal of KMUTNB*, vol. 35, no. 3, pp. 1–14, Jul.–Sep. 2025 (in Thai), Art. no. 253-087557, doi: 10.14416/j.kmutnb.2024.10.016.
- [6] J. Zhang, C. Hu, R. G. Chadha, and S. Singh, "Falco: Fast likelihood-based collision avoidance with extension to human-guided navigation," *Journal of Field Robotics*, vol. 37, no. 8, pp. 1300–1313, 2020, doi: 10.1002/rob.21952.
- [7] J. Tordesillas, B. Lopez, M. Everett, and J. How, "FASTER: Fast and safe trajectory planner for navigation in unknown environments," *IEEE Transactions on Robotics*, vol. 38, no. 2, pp. 922–938, Apr. 2022, doi: 10.1109/TRO.2021.3100142.
- [8] Z. Jian, Z. Lu, X. Zhou, B. Lan, A. Xiao, X. Wang, and B. Liang, "PUTN: A plane-fitting based uneven terrain navigation framework," in *Proceedings of the 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Kyoto, Japan, 2022, pp. 7160–7166, doi: 10.1109/IROS47612.2022.9981038.
- [9] C. Pan, A. Datar, A. Pokhrel, M. Choulas, M. Nazeri, and X. Xiao, "Traverse the non-traversable: Estimating traversability for wheeled mobility on vertically challenging terrain," *arXiv:2409.17479*, 2024, doi: 10.48550/arXiv.2409.17479.

- [10] P. Schoch, F. Yang, Y. Ma, S. Leutenegger, M. Hutter, and Q. Leboutet, "IN-Sight: Interactive navigation through sight," in *Proceedings of the 2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Abu Dhabi, United Arab Emirates, 2024*, pp. 7794–7800, doi: 10.1109/IROS58592.2024.10801928.
- [11] M. Wermelinger, P. Fankhauser, R. Diethelm, P. Krüsi, R. Siegwart, and M. Hutter, "Navigation planning for legged robots in challenging terrain," in *Proceedings of the 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Daejeon, Korea, 2016*, pp. 1184–1189, doi: 10.1109/IROS.2016.7759199.
- [12] E. G. Papadopoulos and D. A. Rey, "A new measure of tipover stability margin for mobile manipulators," in *Proceedings of the 1996 IEEE International Conference on Robotics and Automation (ICRA), USA, 1996*, pp. 3111–3116, doi: 10.1109/ROBOT.1996.509185.
- [13] P. Fankhauser, M. Bloesch, and M. Hutter, "Probabilistic terrain mapping for mobile robots with uncertain localization," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3019–3026, 2018, doi: 10.1109/LRA.2018.2849506.
- [14] M. Pecka, V. Šalanský, K. Zimmermann, and T. Svoboda, "Autonomous flipper control with safety constraints," in *Proceedings of the 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Daejeon, Korea, 2016*, pp. 2889–2894, doi: 10.1109/IROS.2016.77 59447.
- [15] Z. Xu, Y. Chen, Z. Jian, J. Tan, X. Wang, B. Liang, "Hybrid trajectory optimization for autonomous terrain traversal of articulated tracked robots," *IEEE Robotics and Automation Letters*, vol. 9, no. 1, pp. 755–762, Jan. 2024, doi: 10.1109/LRA.2023. 3337593.
- [16] Y. Wang, N. Du, Y. Qin, X. Zhang, R. Song, and C. Wang, "History-aware planning for risk-free autonomous navigation on unknown uneven terrain," in *Proceedings of the 2024 IEEE International Conference on Robotics and Automation (ICRA), Yokohama, Japan, 2024*, pp. 7583–7589, doi: 10.1109/ICRA57147.2024.10610488.
- [17] W. Zhang, Y. Wang, W. Lu, P. Zhang, X. Zhang, Y. Wang, and C. Wang, "Capsizing-guided trajectory optimization for autonomous navigation with rough terrain," in *Proceedings of the 2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Hangzhou, China, 2025*, pp. 21360–21367, doi: 10.1109/IROS60139.2025.11247440.