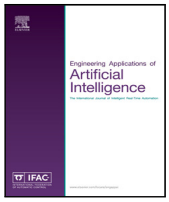
Engineering Applications of Artificial Intelligence 121 (2023) 105976

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai



DEM-AIA: Asymmetric inclination-aware trajectory planner for off-road vehicles with digital elevation models

Manuel Toscano-Moreno *, Anthony Mandow, María Alcázar Martínez, Alfonso García-Cerezo

Robotics and Mechatronics Group, Institute for Mechatronics Engineering & Cyber-Physical Systems (IMECH.UMA), Universidad de Málaga, EII, C/ Doctor Ortiz Ramos, 29071 Málaga, Spain

ARTICLE INFO

Dataset link: https://github.com/mToscanoMoreno/DEM-AIA

Keywords:
Path planning
Unmanned ground vehicle
Digital elevation model
Inclination awareness

ABSTRACT

Planning safe and effective trajectories for off-road unmanned ground vehicles (UGV) is a critical Artificial Intelligence (AI) challenge that can benefit from recent advances in digital elevation models (DEM) for readily capturing accurate terrain geometry. Considering path slopes is crucial to preserve stability and assess terrain traversability at feasible speeds to optimize travel time, which is highly dependent on direction (i.e., pitch and roll). In this article, we propose a new DEM-based asymmetric inclination-aware (DEM-AIA) trajectory planner for ground vehicles. The planner is an any-angle variant of the A* algorithm that computes pitch and roll estimations for each segment crossing cell triangles in the line-of-sight. Furthermore, we define a non-linear velocity constraints function that integrates information about tip-over safety limitations, maximum uphill and downhill slopes for the vehicle, and asymmetric modulation of nominal flat-ground velocity for all pitch and roll combinations. The planner produces a time sub-optimal trajectory with feasible speed references for each segment crossing a cell triangle. Moreover, we provide an extensive experimental analysis of inclination-aware performance on simulated and real-world DEMs as well as a comparison with state-of-the-art path planners adapted to travel-time optimization. An executable version of the planner with parameterizable variations is publicly available.

1. Introduction

The reliability and safety of unmanned ground vehicles (UGV), such as disaster robots, agricultural machinery and planetary rovers, requires novel Artificial Intelligence (AI) techniques that specifically consider the challenges of off-road navigation (Muñoz et al., 2017; Sánchez-Ibáñez et al., 2019). Recently, the widespread use of drones, satellite imagery, and the integration of photogrammetry and lidar have favored readily available and reliable digital elevation models (DEM) for path planning (Delmerico et al., 2017; Lauterbach et al., 2019; Gabrlik et al., 2021). This accurate terrain information can be useful to consider safety and travel cost optimization with novel graph-search variants of A* (Hart et al., 1968), a family of algorithms that is effective for path-finding in discretized maps (Muñoz et al., 2017).

In practice, three-dimensional (3D) terrain relief affects both traversability and path cost. Traversability depends on both the robot and the terrain (Hedrick et al., 2020), including inclination aspects such as tip-over stability, vehicle tractive power or slippage (Sánchez-Ibañez et al., 2021). As for cost function definition, distance optimization on 2D projections is not realistic since paths on the DEM surface are actually longer (Shum et al., 2015; Muñoz et al., 2017). Furthermore, uniform velocity assumptions can be impractical because traversing hills or hollows imposes vehicle speed limitations with respect to nominal flat-ground velocity for reasons such as energy consumption (Choi et al., 2012), terrain slope and traversability (Hedrick et al., 2020), or mission-specific requirements, e.g., victim evacuation in disaster robotics (Toscano-Moreno et al., 2022). Alternatively, trajectory planning produces path segments with corresponding attainable safe velocities. In this sense, minimization of travel time instead of path length can produce more feasible terrain plans with realistic expected execution times, which is crucial for time-critical applications on challenging terrain such as disaster robotics (Murphy et al., 2016) and planetary exploration (Ono et al., 2018).

Inclination-aware path planning requires computing DEM-based slope estimations for path edges. Besides, any-angle path planners avoid zig-zag paths by not limiting search toward 8-neighbor cells (Daniel et al., 2010; Choi and Yu, 2011), producing line-of-sight edges that pass through different cells. Thus, estimating the inclination of constituent segments is required for partial cost and traversability assessment. In the any-angle DEM-based 3Dana path planner, Muñoz et al. (2017) proposed computing the maximum slope for each of the four constituent isosceles right triangles of cells. Nevertheless, using this symmetric upper bound estimation for segment inclination can discard segments with

* Corresponding author.
E-mail address: m.toscano@uma.es (M. Toscano-Moreno).

https://doi.org/10.1016/j.engappai.2023.105976
Received 5 August 2022; Received in revised form 13 January 2023; Accepted 7 February 2023
Available online 15 February 2023

admissible pitch and roll combinations. Precisely, on uneven terrain, traversability and actual path costs are not only inclination-dependent (i.e., conditioned by pitch and roll angles) but also inherently asymmetric (i.e., they differ for uphill and downhill) (Choi et al., 2012; Garrido et al., 2019).

Our motivation for this work arises from actual needs of robotic missions. Our aim is to exploit the growing availability of accurate DEM data to plan effective (i.e., safe, feasible and not over-conservative) trajectories for UGVs on uneven terrain that explicitly consider asymmetric behavior regarding inclination and velocity constraints. Moreover, optimizing travel-time and obtaining feasible execution times is crucial for challenging and time-critical search and rescue (SAR) robotics, where DEM data is obtained from unmanned aerial robots (UAV) (Bravo-Arrabal et al., 2021; Toscano-Moreno et al., 2022). In particular, the major contribution of this work is DEM-AIA, an extension of the 3Dana any-angle path planner (Muñoz et al., 2017) that introduces the following novel features:

1. Travel-time optimization with non-uniform velocity for trajectory planning, where a different feasible velocity is appended to every path segment crossing cell triangles in the any-angle line-of-sight.
2. A novel formulation to estimate inclination for any-angle planning that computes both pitch and roll angles for the actual segments crossing cell triangles. This formulation uses cell-triangle geometry and considers motion direction.
3. The definition of a vehicle-dependent non-linear function of velocity constraints to integrate knowledge about tip-over limitations, maximum uphill and downhill slopes, and the asymmetric modulation of nominal flat-ground velocity for all pitch and roll combinations. This function allows straightforward implementation as a look-up table.

Moreover, we provide an extensive experimental analysis of inclination-aware performance on simulated and real-world DEMs. The analysis includes a comparison with A* and 3Dana versions that have been adapted to travel-time optimization, as well as three limited DEM-AIA configurations to assess the effects of any-angle search and asymmetric traversability combinations. The executable DEM-AIA code as well as the synthetic and real DEMs have been made publicly available for the research community.

The remainder of the article is organized as follows. Section 2 reviews related work. Section 3 provides definitions for the DEM and the graph search-based trajectory planning. Section 4 proposes inclination-aware computations for building the velocity constraints function. Section 5 describes the DEM-AIA algorithm. Section 6 discusses experimental results. Section 7 describes the public repository. Finally, Section 8 offers conclusions and ideas for future work.

2. Related work

A recent review of AI approaches for path planning for autonomous ground vehicles can be found in Sánchez-Ibañez et al. (2021). However, the challenging combination of terrain elevation models and vehicle-dependent variables has not yet been sufficiently studied (Hu et al., 2022). Some recent works have adopted sampling based planners like probabilistic roadmaps (PRM) (Yang et al., 2021) or rapidly-exploring random trees (RRT) using terrain point clouds captured by UAVs (Fedorenko et al., 2018) and considering vehicle kinematics (Ji et al., 2019), machine learning techniques (Hu et al., 2021), metaheuristic algorithms such as ant colony for finding energy optimal paths (Xiaodong et al., 2022; Jing et al., 2022), and optimal control frameworks using vehicle dynamics (Singh et al., 2016) or fast marching methods (FMM) (Garrido et al., 2013). In what follows, we focus on works specifically related to graph-search planning on DEM.

In general, a DEM consists of a dense regular grid of nodes. Limiting search for successors to the eight neighboring cells can reduce computational cost but results in unnatural zig-zag paths (Choi et al., 2012). Some authors have applied splines or other polynomial approximations to the resulting path to improve smoothness (Ishigami et al., 2011; Borges et al., 2019). Alternatively, *any-angle* path planners search toward non-neighboring cells to produce smooth paths (Daniel et al., 2010; Choi and Yu, 2011), which implies computing partial costs for the DEM cells crossed by the line-of-sight (Muñoz et al., 2017).

As for the consideration of *terrain inclination*, some works have processed the DEM to generate threshold-based binary 2D grids of navigable space. Borges et al. (2019) adopted computer vision techniques to associate traversability costs to nodes in grid-based graphs built from orthorectified 2D aerial images. Gabrlik et al. (2021) considered the limitations of the UGV with respect to the maximum slope and the height of the obstacles over a UAV photogrammetry model, resulting on a binary map of reachable areas where the standard A* algorithm could search for UGV paths in a radiation surveillance scenario. Other works have exploited DEM data to estimate actual inclinations. Thus, Muñoz et al. (2017) proposed an estimation of the maximum inclinations of the constituent triangles of DEM cells in order incorporate UGV pitch limitations in the 3Dana any-angle path planner for planetary exploration. Moreover, Huang et al. (2021) computed gradient values by applying the four-direction Sobel operator to the grayscale DEM image for 8-neighbor search. Similarly, Yu et al. (2020) estimated cell slope from the elevations of the 8-neighbor window.

Prevention of *tip-over* risks on uneven or unstructured terrains is a major traversability issue for DEM planning (Kubota et al., 2001; Colas et al., 2013). In many cases, using full dynamics physics simulation of the vehicle and the terrain is not feasible, so simple vehicle information (such as mass and geometry) is pragmatic for predicting pose and stability in DEM-based path planning (Fabian et al., 2020). In this sense, Hines et al. (2021) proposed attitude-aware planning by identifying non-traversable cells based on pitch and roll estimations and steepness thresholds, which were also checked during path tracking by an orientation correction behavior.

On uneven terrain, traversability, energy consumption and travel time can be affected significantly not only by slope but also by heading direction (Choi et al., 2012). A few works have considered the *asymmetric behavior* of ground vehicles for inclination-aware planning. In this sense, FMM can incorporate anisotropy in a natural way for energy-aware motion planing (Garrido et al., 2013), which has been applied with DEMs to minimize UGV height changes on planetary exploration (Garrido et al., 2019) or to compute feasible optimal paths for biped robots (Kumar and Dutta, 2022). As for graph search methods, Santos et al. (2020) proposed an orientation-aware extension of A* for tip-over avoidance on vineyards with steep slopes where possible orientations to neighbor cells were considered as multi-layered grids.

The effect of slopes on vehicle speed, and therefore on total *travel time*, is crucial for time-critical applications like planetary exploration (Ono et al., 2018) and SAR (Delmerico et al., 2017), but travel time optimization for DEM-based graph search has received scarce attention. Instead, some works have proposed a post-processing refinement for distance-optimized paths. Dolgov et al. (2010) used a constant speed for path length optimization by setting a velocity profile based on curvature or the proximity of obstacles. Similarly, Overbye and Saripalli (2021) appended the maximum speed to the path as a function of steering angles. Moreover, Miranda et al. (2019) proposed a motion control method for path tracking that considered the anisotropic behavior of ground agricultural vehicles on slopes and different terrain types. Explicit time-optimal route planning was addressed by Ono et al. (2018) for the Mars 2020 Rover Mission, where each DEM cell was assigned a speed rate based on a classification of slope, soil type and obstacle density. Delmerico et al. (2017) applied the D* algorithm to optimize traversal time based on velocities estimated from the terrain class and a function of distance and gradient between neighbor cells. Adámek et al. (2022) adapted Hybrid A* for travel-time optimization on a DEM based on simplified dynamics and bicycle kinematics, where pitch was

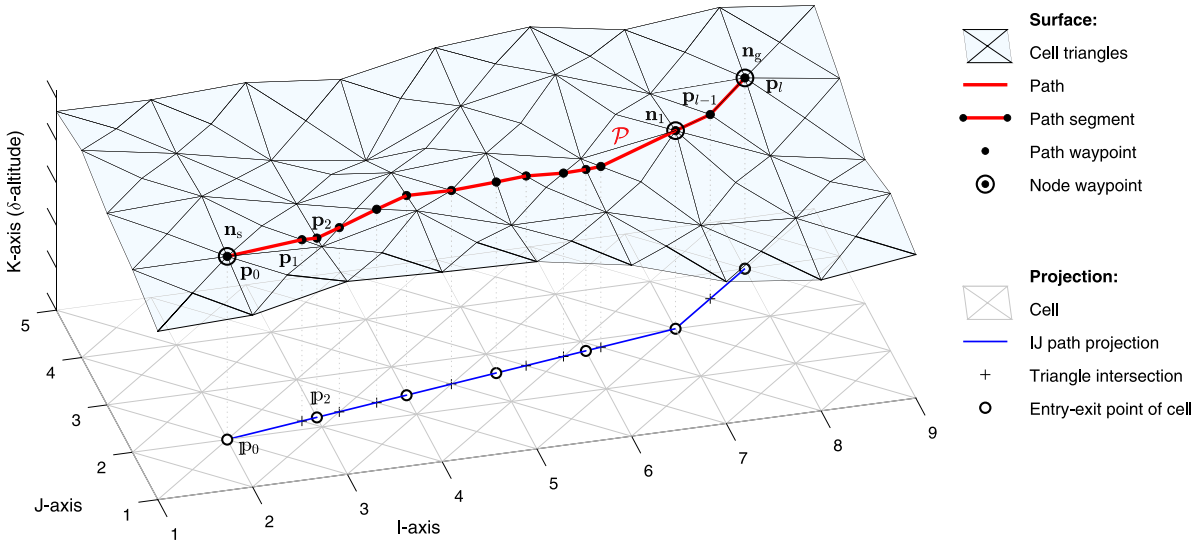


Fig. 1. Illustration of digital elevation model and path $\mathcal{P}$ from a start node $\mathbf{n}_s$ to a goal node $\mathbf{n}_g$. In this example, the any-angle path follows a line-of-sight between $\mathbf{n}_s$ and $\mathbf{n}_1$ and a graph edge between neighbors $\mathbf{n}_1$ and $\mathbf{n}_g$. The inclination-aware path splits segments in the intersections with cell-triangle sides. This path is a sequence of $\ell = 14$ segments defined by 15 waypoints, which are both triangle intersections and nodes.

estimated from the elevation difference between grid cells without considering tip-over safety. Hua et al. (2022) combined a terramechanics model with elevation data to optimize distance by defining fuzzy mobility cost function that depends on elevation, slope and attainable velocities.

To the best of our knowledge, no previous work has addressed global DEM-based trajectory planning for UGVs travel-time optimization by estimating the pitch and roll of path segments for considering asymmetric vehicle behavior and tip-over avoidance. Our proposed solution for this problem is an extension of 3Dana, an algorithm that exploits the effectiveness of the A* family for graph search in discretized DEM maps with any-angle smooth path generation.

3. Digital elevation model for trajectory planning

This section defines the DEM and reviews graph-search any-angle path planning concepts.

3.1. Digital elevation model

A geodesic DEM model can be defined as matrix $\mathbf{Z} \in \mathcal{M}_{M\times N}(\mathbb{R})$ of altitudes $\mathbf{Z}(i,j) = z_{ij}$ for a set of node points $\mathbf{n} = (x_i, y_j, z_{ij})$ over a two-dimensional square grid of spatial resolution δ.

To simplify planning computations, a normalized DEM can be defined as $\mathbf{K} = \mathbf{Z}/\delta$, where nodes $\mathbf{n} = (i, j, k_{ij})$ have integer coordinates (see Fig. 1). In this work, graph search is performed on the IJ plane node projections, where $\mathtt{n}$ denotes the projection of $\mathbf{n}$, and $\mathbf{K}$ is used to compute inclination-aware costs.

We adopt a corner-node representation where a grid cell $\mathbf{c} = (i, j)$ is defined by IJ nodes (i, j), $(i+1, j)$, $(i+1, j+1)$ and $(i, j+1)$. Moreover, cell diagonals define four constituent isosceles right triangles that share the center point $\mathbf{p}_\mathbf{c} = (i + 1/2, j + 1/2, k_\mathbf{c})$ whose normalized altitude $k_\mathbf{c}$ is estimated as the average of the four cell nodes. Triangles allow estimating the normalized altitude k_a of any arbitrary DEM surface point $\mathbf{p}_a$ (Muñoz et al., 2017). In our approach, the formulation is simplified because only points belonging to the cell-triangle sides need to be interpolated. Thus, for a point $\mathbf{p}_a = (i_a, j_a, k_a)$ in the triangle side defined by vertices $\mathbf{p}_1 = (i_1, j_1, k_1)$ and $\mathbf{p}_2 = (i_2, j_2, k_2)$, altitude k_a can be computed as:

$$k_a = k_1 + (k_2 - k_1)\sqrt{\frac{(i_a - i_1)^2 + (j_a - j_1)^2}{(i_2 - i_1)^2 + (j_2 - j_1)^2}}. \tag{1}$$

3.2. Graph search-based planning concepts

Let search space $\mathcal{G}$ be a weighted and directed graph defined as the set $\mathcal{N}$ of $M \times N$ normalized DEM nodes and the set $\mathcal{E}$ of edges representing the node connections, which we define as 8-neighbors. In this work, we focus on slope-based traversability. Therefore, we do not consider removing from $\mathcal{G}$ nodes defined *a priori* as explicit obstacles, such as walls and rocks, or blocked areas (e.g., as in a binary occupancy map), since obstacles can be implicit in DEMs as insurmountable steep elevations.

Path planners of the A* family search for a globally optimal cost path connecting a start node $\mathbf{n}_s$ with a goal node $\mathbf{n}_g$. This is an iterative node expansion process from $\mathbf{n}_s$ that uses a queue of nodes to expand (OPEN) sorted by an evaluation function $f(\mathbf{n}) = g(\mathbf{n}) + h(\mathbf{n})$ that estimates the global cost from $\mathbf{n}_s$ to $\mathbf{n}_g$ passing through $\mathbf{n}$. The cost from $\mathbf{n}_s$ to $\mathbf{n}$ is $g(\mathbf{n}) = g(\mathtt{parent}(\mathbf{n})) + w(\mathtt{parent}(\mathbf{n}), \mathbf{n})$, with $g(\mathbf{n}_s) = 0$, and the heuristic $h(\mathbf{n})$ estimates the cheapest cost from $\mathbf{n}$ to $\mathbf{n}_g$. The resulting graph path is a sequence of edges connecting pairs of nodes (i.e., parent to child nodes), where the global cost $g(\mathbf{n}_g)$ is the sum of edge costs. The edge cost between nodes $\mathbf{n}$ and $\mathbf{n}'$, depends on the optimization criterion, which can be a metric such as path length, path smoothness, path safety, total execution time and total energy consumption (Atiyah et al., 2021). In this work, total travel time is defined as the optimization criterion.

Particularly, inclination-aware path planning requires computing DEM-based slope estimations for path edges. Furthermore, in any-angle search (Daniel et al., 2010; Choi and Yu, 2011), line-of-sight edges pass through different cells, so computing partial costs is necessary. We adopt subdivision of edges into segments that cross different cell triangles (Muñoz et al., 2017), as illustrated in Fig. 1. Then, the path $\mathcal{P}$ can be defined as a sequence of ℓ straight segments traversing cell triangles from a start node $\mathbf{n}_s$ to a goal node $\mathbf{n}_g$:

$$\mathcal{P} = \left\{\overline{\mathbf{p}_0\mathbf{p}_1}, \overline{\mathbf{p}_1\mathbf{p}_2}, \ldots \overline{\mathbf{p}_{\ell-1}\mathbf{p}_\ell}\right\},$$

where $\mathbf{p}_\imath$, for $0 \le \imath \le \ell$ and $\imath \in \mathbb{N}$, are path waypoints in normalized DEM coordinates with $\mathbf{p}_0 = \mathbf{n}_s$ and $\mathbf{p}_\ell = \mathbf{n}_g$. These waypoints need to be transformed into geodesic coordinates for real-world application.

4. Inclination-aware computations

This section formulates the estimation of pitch and roll angles for computing the inclination-aware cost of path segments. Besides, we

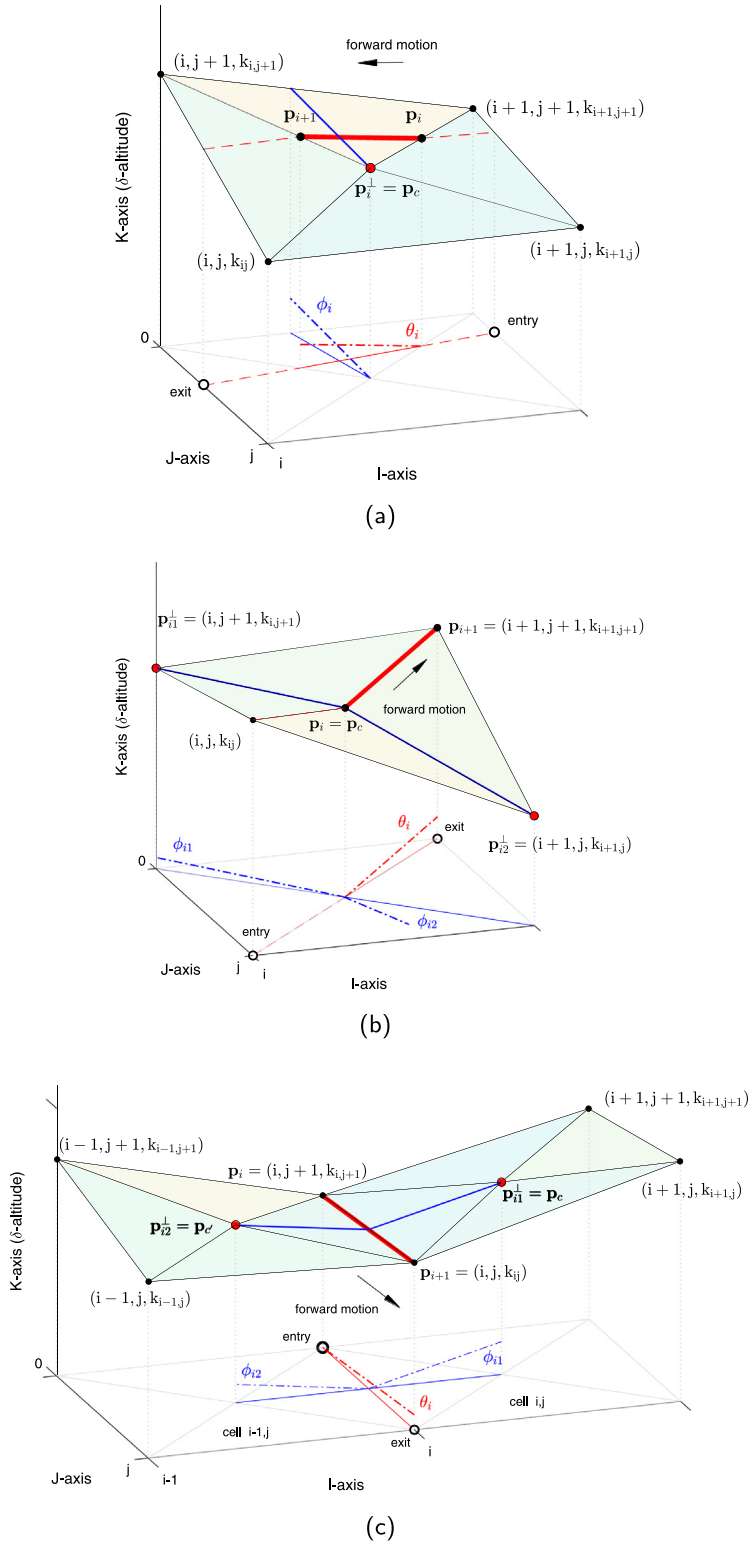


Fig. 2. Illustration of path segments on cell triangles with the representation of inclination angles on motion direction θ_i and lateral ϕ_i: (a) crossing a cell triangle; (b) cell diagonal; (c) cell side.

propose an application of the support polygon concept for defining pitch and roll limits for tip-over avoidance as well as a function to assign asymmetric reference velocities to segments based on slope and heading direction. These constraints are incorporated in a vehicle-dependent velocity function $v_{\text{ref}}(\theta, \phi)$ for the DEM-AIA trajectory planner.

4.1. Pitch and roll estimations

The proposed inclination-aware trajectory planning algorithm appends pitch θ_i and roll ϕ_i angle estimations (i.e., the forward and lateral inclinations with respect to the IJ plane, respectively) to the ith path

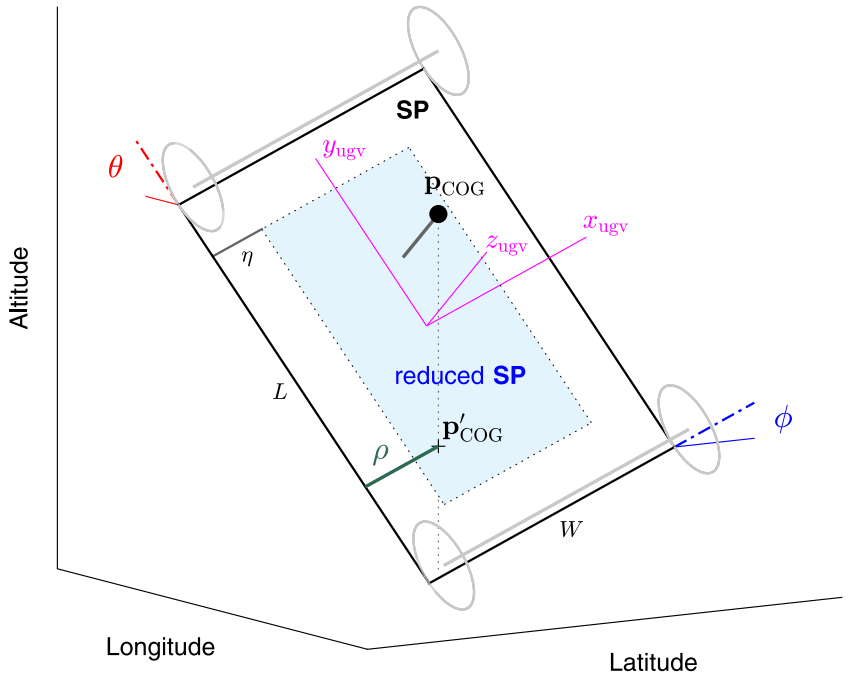


Fig. 3. Illustration of tip-over stability analysis based on the supporting polygon (SP) for a 4-wheeled vehicle. Pitch and roll angles affect the vertical projection of the COG ($\mathbf{p}'_{COG}$) and the tip-over distance stability margin ρ. A reduced SP provides a margin for DEM uncertainties.

segment $\overline{\mathbf{p}_i\mathbf{p}_{i+1}}$. With this purpose, we define function Θ as:

$$\langle \overline{\mathbf{p}_i\mathbf{p}_{i+1}}, \theta_i, \phi_i \rangle = \Theta\left(\overline{\mathbf{p}_i\mathbf{p}_{i+1}}\right). \tag{2}$$

The estimation of θ_i in Eq. (2) is computed as:

$$\theta_i = \tan^{-1}(A/B), \tag{3}$$

with

$$A = \left(k_{i+1} - k_i\right),$$
$$B = \sqrt{\left(i_{i+1} - i_i\right)^2 + \left(j_{i+1} - j_i\right)^2}.$$

As for ϕ_i, this is computed from the cell triangle that contains the path segment (see Fig. 2). In particular, ϕ_i can be obtained from the relation between areas of equivalent parallelograms and the distance from a point $\mathbf{p}_i^{\perp}$ to a line as:

$$\phi_i = \cos^{-1}\left(C_k \sqrt{\frac{A^2 + B^2}{B^2\left(C_i^2 + C_j^2 + C_k^2\right)}}\right), \tag{4}$$

with

$$C_i = \left(j_i^{\perp} - j_i\right)\left(k_{i+1} - k_i\right) - \left(k_i^{\perp} - k_i\right)\left(j_{i+1} - j_i\right),$$
$$C_j = \left(i_i^{\perp} - i_i\right)\left(k_{i+1} - k_i\right) - \left(k_i^{\perp} - k_i\right)\left(i_{i+1} - i_i\right),$$
$$C_k = \left(i_i^{\perp} - i_i\right)\left(j_{i+1} - j_i\right) - \left(j_i^{\perp} - j_i\right)\left(i_{i+1} - i_i\right),$$

where $\mathbf{p}_i^{\perp} = \left(i_i^{\perp}, j_i^{\perp}, k_i^{\perp}\right)$ can be any point of the cell triangle that does not belong to $\overline{\mathbf{p}_i\mathbf{p}_{i+1}}$.

Two estimation approaches are proposed depending on whether the segment crosses a triangle or coincides with a triangle side. When the segment crosses a triangle, roll and pitch depends only on triangle inclination and segment direction (see Fig. 2(a)), so only one auxiliary point $\mathbf{p}_i^{\perp}$ is needed. For simplicity, $\mathbf{p}_i^{\perp}$ is chosen as the cell center point $\mathbf{p_c}$.

When the segment coincides with a triangle side, i.e., connects a cell node with the cell center or two cell nodes, it is shared by two neighboring triangles with their corresponding inclinations (see Fig. 2(b–c)). In this case, two auxiliary points are required to compute ϕ_i as the average value of Eq. (4) for both triangles:

$$\phi_i = \frac{1}{2}\left(\phi_{i1} + \phi_{i2}\right).$$

For simplicity, the two points are chosen as the vertices of the both neighboring triangles that do not belong to the segment. These points are the nodes of the opposite cell diagonal if the segment contains the cell center (see Fig. 2(b)) or the adjacent cell centers if the segment connects two cell nodes (see Fig. 2(c)).

4.2. UGV pitch and roll constraints

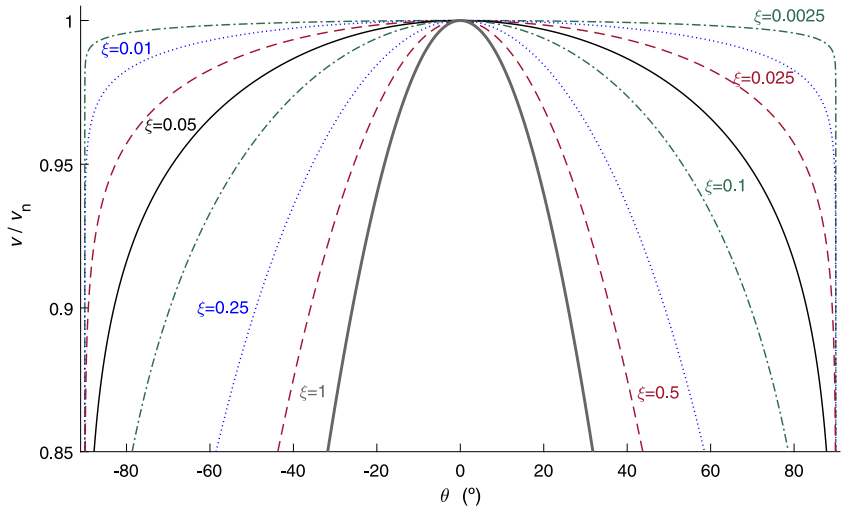
The slope-based traversability limitations for a ground vehicle can be due to aspects such as maximum traction power, slippage or tip-over prevention (Isher et al., 2022). In this work, we define asymmetric pitch $[\theta_{min}, \theta_{max}]$ and roll $[\phi_{min}, \phi_{max}]$ constraints for tip-over avoidance. With this purpose, we consider wheeled or tracked UGVs whose local reference frame $\{x_{ugv}, y_{ugv}, z_{ugv}\}$ is defined at the center of the rectangular supporting polygon (SP), with length L and width W, defined by ground contact points. The vehicle travel speed and acceleration are assumed to be moderate, so external and inertial forces other than the weight force can be neglected (Vidoni et al., 2015). Therefore, a static stability analysis can be adopted.

Fig. 3 illustrates tip-over static stability analysis based on the SP (Safar et al., 2012; Morales et al., 2013), where the vehicle's center of gravity (COG) is $\mathbf{p}_{COG} = \left(x_{COG}, y_{COG}, z_{COG}\right)$ in the local frame coordinates. In actual vehicles, this point is not necessarily centered with respect to the longitudinal or transversal axes. For static stability, the vertical projection of $\mathbf{p}_{COG}$ over the ground plane ($\mathbf{p}'_{COG}$) must fall within the SP. Then, a tip-over distance stability margin Papadakis (2013) can be defined as the shortest distance ρ from $\mathbf{p}'_{COG}$ to the sides of the SP.

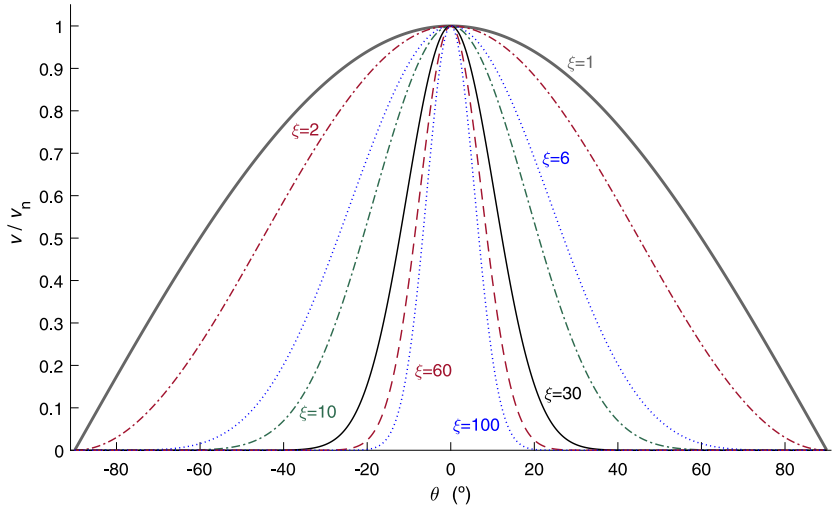
In order to add a margin for DEM uncertainties, a reduced SP could be defined at a distance $\eta = \rho_{tol}\min\{L, W\}/2$ within original SP limits, where $0 \le \rho_{tol} \le 1$. Then, the following expressions:

$$D_x = x_{COG} \pm \frac{1}{2}W\left(1 - \rho_{tol}\right),$$
$$D_y = y_{COG} \pm \frac{1}{2}\left(L - W \cdot \rho_{tol}\right)$$

yield the pair of distances from the COG to the left and right sides of the reduced SP and the pair of distances to the front and rear sides, respectively.



(a) Weak pitch dependence ($0 < \xi \leq 1$).



(b) Stronger pitch dependence ($\xi \geq 1$).

Fig. 4. Normalized representation of v/v_n (Eq. (7)) for different ξ values.

From this tip-over stability analysis, the pitch thresholds $\theta_{\min}, \theta_{\max}$ can be obtained as:

$$\begin{pmatrix} \theta_{\min} \\ \theta_{\max} \end{pmatrix} = \tan^{-1}\left(\frac{D_y}{z_{\text{COG}}} \right), \tag{5}$$

and roll thresholds $\phi_{\min}, \phi_{\max}$ are a function of the estimated pitch θ_ι (Eq. (3)):

$$\begin{pmatrix} \phi_{\min} \\ \phi_{\max} \end{pmatrix} = \tan^{-1}\left(\frac{-D_x}{D(\theta_\iota)\sin\theta_\iota + z_{\text{COG}}\cos\theta_\iota} \right), \tag{6}$$

where

$$D(\theta_\iota) = \begin{cases} \min\{D_y\} & \text{if} \quad \theta_\iota < 0 \text{ and } y_{\text{COG}} > 0 \text{ or} \\ & \quad \theta_\iota > 0 \text{ and } y_{\text{COG}} < 0, \\ \max\{D_y\} & \text{otherwise.} \end{cases}$$

4.3. UGV locomotion forward speed constraints

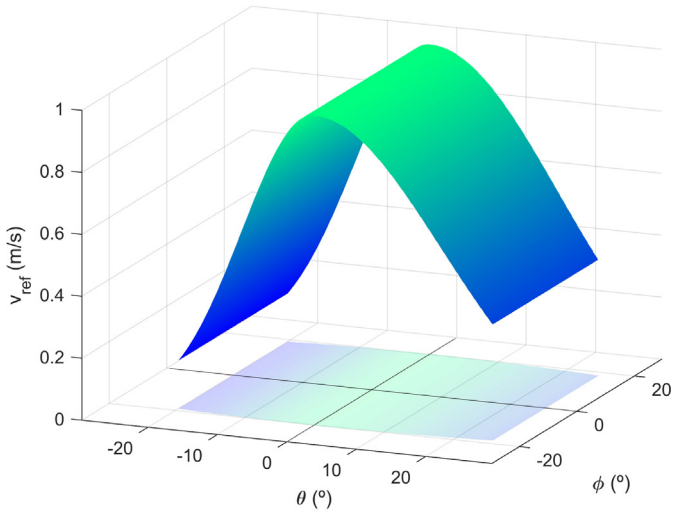
On uneven terrain, the optimization of travel time is not equivalent to path length because the constant velocity assumption does not necessarily hold for different slopes and heading directions (Choi et al., 2012). The effect of terrain inclination on the reference velocity can be defined by a vehicle-dependent function $v(\theta, \phi)$, which modulates the constant nominal velocity v_n, i.e., that for flat terrain, into a reference velocity for the ιth segment based on θ_ι and ϕ_ι.

Without loss of generality, in this work we define $v(\theta, \phi) = v(\theta)$, where the effect of ϕ for forward speed is considered negligible, for simplicity, under the stability conditions established in Section 4.2. In particular, the proposed forward speed constraint function is defined as:

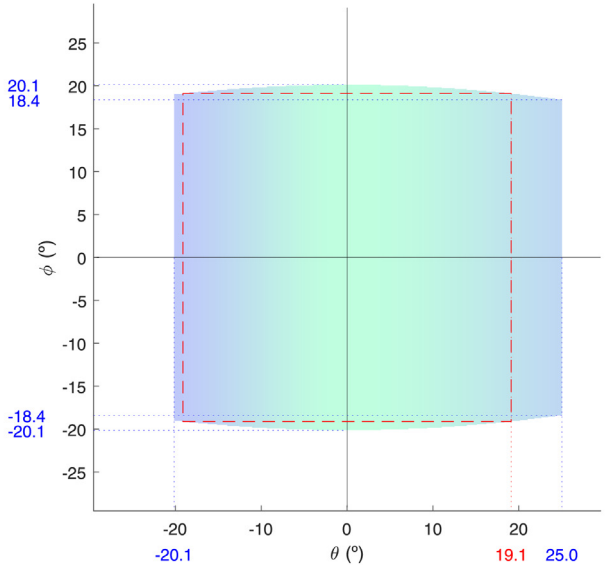
$$v(\theta) = v_n \cdot \cos^{\xi}\theta\,, \quad \begin{cases} \xi = \xi_\downarrow & \text{if} \quad \theta < 0 \\ \xi = \xi_\uparrow & \text{if} \quad \theta \geq 0 \end{cases}, \tag{7}$$

where $\xi_\downarrow, \xi_\uparrow \in \mathbb{R}_{\geq 0}$ are coefficients defining the asymmetric dependence on θ. Thus, when $\xi = 0$, the reference $v = v_n$ is constant. On the other hand, when $\xi > 0$, v is reduced depending on slope, as shown in Fig. 4. This figure represents the normalized relation v/v_n for different ξ values, where Fig. 4(a) and 4(b) show curves for applying with weaker ($0 < \xi \leq 1$) or stronger ($\xi \geq 1$) velocity dependence degrees, respectively.

Given a UGV, the ξ coefficients could be set by experimentally determining the velocities v that show acceptable performance for different terrain inclinations and then adjusting this data to the appropriate left and right curves in Fig. 4. In case of asymmetric constraints (e.g., descending requires less speed), then the left side ($\xi_\downarrow$) and the right side ($\xi_\uparrow$) of v in Fig. 4 would be different ξ values.



(a) Three-dimensional surface plot.



(b) Two-dimensional representation. The red dashed square indicates the largest symmetric stability region.

Fig. 5. Illustration of the inclination-aware velocity constraints function v_{ref} for asymmetric vehicle-dependent coefficients $\xi_{\downarrow} = 30$ and $\xi_{\uparrow} = 10$. For clarity, only non-zero (i.e., stable) values are shown.

4.4. Inclination-aware velocity constraints function

The DEM-AIA trajectory planner uses vehicle-dependent inclination-aware velocity constraints defined as a non-linear function $v_{\text{ref}}(\theta, \phi)$ that assigns the reference velocity for a path segment based on slope and heading direction. This function can be implemented as a 2D look-up table built by incorporating UGV locomotion forward speed constraints as given by Eq. (7) and then setting zero values to all pitch and roll combinations over the thresholds computed in Eqs. (5) and (6).

Fig. 5 illustrates the inclination-aware velocity constraints function for the case-study vehicle in Section 6. The function $v_{\text{ref}}(\theta, \phi)$ incorporates tip-over avoidance (Section 4.2) and the forward speed constraints (Section 4.3). For the sake of clarity, only non-zero values are shown. Zero values (i.e., those outside of the shaded projection on the $\theta\phi$ plane, shown in Fig. 5(b)), indicate the unstable inclination combinations. In this case, the stable region is asymmetric with respect to pitch due to the advanced position of the COG. Moreover, the curved sides of this region indicate that the maximum roll angle supported by the vehicle decreases with increasing pitch angle (on both uphill and downhill slopes). The figure also shows the largest stability region (red dashed square) that could be obtained for identical symmetric pitch and roll thresholds, which is more conservative (i.e., restrictive) than the proposed function.

5. DEM-AIA algorithm

This section proposes an any-angle DEM-based asymmetric inclination-aware planner (DEM-AIA) for ground vehicles that computes a sub-optimal travel-time trajectory by using a vehicle-dependent velocity constraints function and pitch and roll estimations for each segment crossing cell triangles. First, we present an overview of the algorithm. Then, we describe the inner functions.

5.1. Overview of the DEM-AIA algorithm

Algorithm 1 presents the basic structure for the DEM-AIA trajectory planner. This algorithm is a variant of A* for DEM-based search that admits parameterization of different inclination-aware strategies for comparison and analysis.

Algorithm 1: DEM-AIA trajectory planner

```
Function ⟨P, V⟩ = DEM-AIA (K, n_s, n_g, v_ref, any_angle, heuristic):
    ▷ initialization:
1   ⟨P, V⟩ ← ⟨∅, ∅⟩
2   for n ∈ K do
3       parent(n) ← null
4       g(n) ← ∞
5       switch heuristic do
6           case Euclidean_time do h(n) ← h_e(n)        ▷ Eq. (9)
7           case Octile_time do h(n) ← h_o(n)           ▷ Eq. (10)
8   g(n_s) ← 0
9   OPEN ← {n_s}
10  if not any_angle then CLOSED ← ∅
    ▷ iterative node expansion:
11  while OPEN ≠ ∅ do
12      n ← argmin_{n ∈ OPEN} {f(n)}
13      OPEN ← OPEN − {n}
14      if not any_angle then CLOSED ← CLOSED ∪ {n}
15      if n = n_g then
            ▷ obtain resulting trajectory, backtracking from n_g to n_s:
16          while n ≠ ∅ do
17              P ← segments(n) ∪ P
18              n ← parent(n)
            ▷ assign reference velocity for every path segment:
19          for ⟨p_i p_{i+1}, θ_i, φ_i⟩ ∈ P do V ← V ∪ v_ref(θ_i, φ_i)   ▷ Eq. (7)
20          return ⟨P, V⟩
21      for n′ ∈ successors(n) do
22          if any_angle or n′ ∉ CLOSED then
23              UpdateNode(n, n′, K, v_ref, any_angle)   ▷ Algorithm 2
24  return fail
```

The algorithm returns a tuple $\langle \mathcal{P}, \mathcal{V} \rangle$, where $\mathcal{P}$ is a sequence of path segments (see Section 3.2) and $\mathcal{V}$ the corresponding reference velocities. The inputs for the algorithm are the normalized DEM matrix $\mathbf{K}$, the start $\mathbf{n}_s$ and goal $\mathbf{n}_g$ nodes, a normalized version (i.e., in cells/second) of the inclination-aware velocity function $v_{ref}(\theta, \phi)$, and two search configuration parameters (`any_angle` and `heuristic`) to allow fair comparison against other planners.

The vehicle-dependent inclination-aware velocity constraints are given as a non-linear function $v_{ref}(\theta, \phi)$ that assigns the reference velocity for a path segment based on slope and heading direction. The methodology for computing $v_{ref}(\theta, \phi)$ is presented in Section 4.

As for configuration parameters, first, the Boolean `any_angle` enables line-of-sight search towards non-neighboring nodes, which does not use the CLOSED set because expanded nodes are still eligible for re-expansion. Second, the `heuristic` parameter identifies the type of heuristic function $h(\mathbf{n})$. For optimizing total travel time, $h(\mathbf{n})$ can be defined as:

$$h(\mathbf{n}) = \frac{\texttt{estimated\_distance}\left(\mathbf{n}, \mathbf{n}_g\right)}{v_n}, \tag{8}$$

where v_n is the maximum speed considered for the vehicle in the absence of terrain restrictions. Commonly used distance estimations (Choi and Yu, 2011) for the numerator of Eq. (8) are 3D Euclidean distance, which gives an Euclidean time heuristic:

$$h_e(\mathbf{n}) = \frac{\sqrt{\Delta_i^2 + \Delta_j^2 + \Delta_k^2}}{v_n}, \tag{9}$$

or the 3D octile distance:

$$h_o(\mathbf{n}) = \frac{\sqrt{\left(\left|\Delta_i - \Delta_j\right| + \sqrt{2}\min\left\{\Delta_i, \Delta_j\right\}\right)^2 + \Delta_k^2}}{v_n}, \tag{10}$$

with $\Delta_i = |i_g - i|$, $\Delta_j = |j_g - j|$, $\Delta_k = |k_{i_g j_g} - k_{ij}|$, for an octile time estimation.

DEM-AIA consists of two phases (see Algorithm 1): initialization and iterative node expansion. First, data structures are initialized, the heuristic function $h(\mathbf{n})$ is evaluated for every node $\mathbf{n}$ in $\mathbf{K}$ (lines 5 to 7), and $\mathbf{n}_s$ is inserted into OPEN (line 9). Then, while there are nodes in OPEN, the iterative node expansion process is performed. The node $\mathbf{n}$ in OPEN with the lowest $f(\mathbf{n})$ is extracted (lines 12 and 13). If $\mathbf{n}$ is $\mathbf{n}_g$, the algorithm returns the trajectory $\langle \mathcal{P}, \mathcal{V} \rangle$ by backtracking of the path segments and the parent nodes from $\mathbf{n}_g$ to $\mathbf{n}_s$. Else, successor nodes of $\mathbf{n}$ are updated or inserted into OPEN by the `UpdateNode` function (line 23), which computes the evaluation function with any-angle considerations. If OPEN becomes empty, $\mathbf{n}_g$ is unreachable. `UpdateNode` is described below along with rest of inner functions.

5.2. DEM-AIA inner functions

Function `UpdateNode` (see Algorithm 2) updates or inserts successor nodes into OPEN (line 23 of Algorithm 1). This algorithm calls functions `EntryExitPoints` and `Segments`, shown in Algorithms 3 and 4, respectively. These algorithms define the inner functions of the DEM-AIA trajectory planner.

Algorithm 2: Computes the evaluation function $f(\mathbf{n}')$, updates $segments(\mathbf{n}')$ and $parents(\mathbf{n}')$, and inserts the appropriate node $\mathbf{n}'$ into OPEN.

```
Function UpdateNode (n, n′, K, v_ref, any_angle):
    ▷ Compute g_n = g(n) + cost (n, n′)
1   g_n ← g(n)
2   segments_n ← Segments(n, n′)                        ▷ Algorithm 4
3   for ⟨p_i p_{i+1}, θ_i, φ_i⟩ ∈ segments_n do
4       g_n ← g_n + SegmentCost(⟨p_i p_{i+1}, θ_i, φ_i⟩, v_ref, δ)   ▷ Eq. (11)
    ▷ Compute g_p = g(parent(n)) + cost (parent(n), n′)
5   g_p ← −∞
6   segments_p ← ∅
7   if any_angle and n ≠ n_s then
8       g_p ← g(parent(n))
9       for (P_j, P_{j+1}) ∈ EntryExitPoints(parent(n), n′) do   ▷ Alg. 3
10          partialSegments ← Segments(P_j, P_{j+1})
11          for ⟨p_i p_{i+1}, θ_i, φ_i⟩ ∈ partialSegments do
12              g_p ← g_p + SegmentCost(⟨p_i p_{i+1}, θ_i, φ_i⟩, v_ref, δ)
13          segments_p ← segments_p ∪ partialSegments
    ▷ Select the graph edge with the lowest cost g(n′) and then update parent
      of n′, segments of path edge to n′, and insert n′ into OPEN
14  if g_p ≠ ∞ and g_p ≤ g_n and g_p < g(n′) then
15      parent(n′) ← parent(n)
16      segments(n′) ← segments_p
17      g(n′) ← g_p
18      f(n′) ← g_p + h(n′)
19      OPEN ← OPEN ∪ {n′}
20  else if g_n ≠ ∞ and g_n < g(n′) then
21      parent(n′) ← n
22      segments(n′) ← segments_n
23      g(n′) ← g_n
24      f(n′) ← g_n + h(n′)
25      OPEN ← OPEN ∪ {n′}
```

5.2.1. UpdateNode function

Function `UpdateNode` (Algorithm 2) is similar to the one used by 3Dana (Muñoz et al., 2017) but we incorporate the vehicle-dependent inclination-aware velocity constraints function v_{ref} as well as the travel-time optimization criterion.

As common in any-angle methods (Daniel et al., 2010; Muñoz et al., 2017), the selection of $parent(\mathbf{n}')$, where $\mathbf{n}'$ is an 8-neighbor successor

of $\mathbf{n}$, requires computing two alternative tentative costs: the total cost passing through n, which is $g_n = g(\mathbf{n}) + \texttt{Cost}\left(\mathbf{n}, \mathbf{n}'\right)$ (lines 1 to 4), and the total cost using a line-of-sight between $parent(\mathbf{n})$ and $\mathbf{n}'$, defined as $g_p = g(parent(\mathbf{n})) + \texttt{Cost}\left(parent(\mathbf{n}), \mathbf{n}'\right)$ (lines 5 to 13), where Cost() indicates the partial cost between two nodes:

$$\texttt{Cost}(\mathbf{n}, \mathbf{n}') = \sum_{i=0}^{S_{\mathbf{n},\mathbf{n}'}-1} \texttt{SegmentCost}\left(\overline{\mathbf{p}_i\mathbf{p}_{i+1}}, v_{\text{ref}}(\theta_i, \phi_i), \delta\right)$$

where $S_{\mathbf{n},\mathbf{n}'}$ is the number of triangle segments between $\mathbf{n}$ and $\mathbf{n}'$, with $p_0 = \mathbf{n}$ and $p_{S_{\mathbf{n},\mathbf{n}'}} = \mathbf{n}'$. Moreover, the segment cost follows a travel-time optimization criterion based on $v_{\text{ref}}(\theta, \phi)$:

$$\texttt{SegmentCost}\left(\overline{\mathbf{p}_i\mathbf{p}_{i+1}}, v_{\text{ref}}(\theta_i, \phi_i)\right) = \frac{\sqrt{\left(i_{i+1} - i_i\right)^2 + \left(j_{i+1} - j_i\right)^2 + \left(k_{i+1} - k_i\right)^2}}{v_{\text{ref}}(\theta_i, \phi_i)}, \tag{11}$$

which is infinite for non-traversable segments, i.e., $v_{\text{ref}}(\theta_i, \phi_i)$ is zero. Finally, the lowest finite tentative cost is used to insert or update $\mathbf{n}'$ in *parent*, *segment* and OPEN if it improves the current cost $g(\mathbf{n}')$ (lines 14 to 25).

To calculate path segments, the function UpdateNode obtains the segment waypoints by two inner functions: EntryExitPoints (line 9) and Segments (lines 10 and 2), described below.

5.2.2. EntryExitPoints function

The function EntryExitPoints, shown in Algorithm 3, calculates an ordered list of two-dimensional points at which the IJ projection of the line-of-sight between two non-neighboring nodes intersects the grid lines. This work refers to these points as cell entry/exit points (see Fig. 1).

Algorithm 3: Returns the list of cell entry/exit points for the line-of-sight between nodes $\mathbf{n}_1 = (i_1, j_1, k_1)$ and $\mathbf{n}_2 = (i_2, j_2, k_2)$.

Function LIST of $\mathbb{p}_{\mathcal{J}}$ = EntryExitPoints $(\mathbf{n}_1, \mathbf{n}_2)$:

▷ initialization
1 LIST $\leftarrow \{(i_1, j_1)\}$
2 $\Delta_i \leftarrow$ **if** $i_2 > i_1$ **then** 1 **else if** $i_2 < i_1$ **then** −1 **else** 0
3 $\Delta_j \leftarrow$ **if** $j_2 > j_1$ **then** 1 **else if** $j_2 < j_1$ **then** −1 **else** 0
4 $m \leftarrow (j_2 - j_1) / (i_2 - i_1)$
5 $\mathbb{p}_a \overset{\text{def}}{=} (i_a, j_a) \leftarrow (i_1 + \Delta_i,\ j_1 + m \cdot \Delta_i)$
6 $\mathbb{p}_b \overset{\text{def}}{=} (i_b, j_b) \leftarrow (i_1 + \Delta_j / m,\ j_1 + \Delta_j)$
7 **if** $m = 0$ **then** $i_b \leftarrow \Delta_i \cdot \infty$ **else if** $m = \infty$ **then** $j_a \leftarrow \Delta_i \cdot \infty$
8 **if** $m = 1$ **then** $\mathbb{p}_b \leftarrow (i_2, j_2)$
▷ creation of cell entry/exit points list
9 **while not** $(i_a = i_2$ **and** $j_b = j_2)$ **do**
10 **if** $(i_a \le i_b$ **and** $\Delta_i > 0)$ **or** $(i_a \ge i_b$ **and** $\Delta_i < 0)$ **then**
11 LIST $\leftarrow$ LIST $\cup \{\mathbb{p}_a\}$
12 $i_a \leftarrow i_a + \Delta_i$
13 **if** $m \neq \infty$ **then** $j_a \leftarrow j_1 + m\,(i_a - i_1)$
14 **else**
15 LIST $\leftarrow$ LIST $\cup \{\mathbb{p}_b\}$
16 $j_b \leftarrow j_b + \Delta_j$
17 **if** $m \neq 0$ **then** $i_b \leftarrow i_1 + (j_b - j_1) / m$
18 LIST $\leftarrow$ LIST $\cup \{(i_2, j_2)\}$
19 **return** LIST

This function is based on the clipping line algorithm (Dimri, 2015) and uses the parametric form of a two-dimensional line equation to calculate the intersection points of a line segment with the boundaries of a clip window. Unlike the algorithm used by 3Dana, Algorithm 3 computes the intersection points for any pair of line-of-sight nodes, including pairs with the same vertical or horizontal coordinates.

5.2.3. Segments function

The function Segments, shown in Algorithm 4, computes an ordered list of path segments $\langle \overline{\mathbf{p}_i\mathbf{p}_{i+1}}, \theta_i, \phi_i \rangle$ between one cell's pair of entry/exit points, i.e. one segment for every triangle crossed within the cell. This function distinguishes two cases of cell entry/exit points: (i) when both correspond to nodes, and (ii) when one or both do not correspond with nodes.

Algorithm 4: Returns the list of segments with pitch and roll angles between a cell's entry $\mathbb{p}_1 = (i_1, j_1)$ and exit $\mathbb{p}_2 = (i_2, j_2)$ points.

Function LIST of $\langle \overline{\mathbf{p}_i\mathbf{p}_{i+1}}, \theta_i, \phi_i \rangle$ = Segments $(\mathbb{p}_1, \mathbb{p}_2, \mathbf{K})$:

▷ compute cell and its center:
1 $\mathbf{c} \overset{\text{def}}{=} (i, j) \leftarrow \lfloor (\mathbb{p}_1 + \mathbb{p}_2) / 2 \rfloor$
2 $\mathbf{p}_c \leftarrow \texttt{CellCenter}(\mathbf{c})$
▷ both entry/exit points correspond to nodes:
3 **if** $\mathbb{p}_1 = \lfloor \mathbb{p}_1 \rfloor$ **and** $\mathbb{p}_2 = \lfloor \mathbb{p}_2 \rfloor$ **then**
4 $\mathbf{p}_1 \leftarrow (\mathbb{p}_1, \mathbf{K}(i_1, j_1))$; $\mathbf{p}_2 \leftarrow (\mathbb{p}_2, \mathbf{K}(i_2, j_2))$
5 **if** $i_1 = i_2$ **or** $j_1 = j_2$ **then** ▷ 4-adjacent nodes
6 LIST $\leftarrow \{\Theta(\overline{\mathbf{p}_1\mathbf{p}_2})\}$ ▷ Eq. (2)
else ▷ opposite nodes in the cell diagonal
7 LIST $\leftarrow \{\Theta(\overline{\mathbf{p}_1\mathbf{p}_c}), \Theta(\overline{\mathbf{p}_c\mathbf{p}_2})\}$
else ▷ any of entry/exit points does not correspond to node:
▷ look up or estimate altitude corresponding to both entry/exit points:
8 $\mathbf{p}_{e1} \leftarrow (\lfloor \mathbb{p}_1 \rfloor, \mathbf{K}\lfloor (i_1, j_1) \rfloor)$; $\mathbf{p}_{e2} \leftarrow (\lfloor \mathbb{p}_2 \rfloor, \mathbf{K}\lfloor (i_2, j_2) \rfloor)$
9 **if** $\mathbb{p}_1 = \lfloor \mathbb{p}_1 \rfloor$ **then** $\mathbf{p}_1 \leftarrow (\mathbb{p}_1, \mathbf{K}(i_1, j_1))$ **else**
10 **if** $i_1 = \lfloor i_1 \rfloor$ **then** $\mathbf{p}_{e1'} \leftarrow (\lfloor i_1, j_1 + 1 \rfloor, \mathbf{K}\lfloor (i_1, j_1 + 1) \rfloor)$
11 **else** $\mathbf{p}_{e1'} \leftarrow (\lfloor i_1 + 1, j_1 \rfloor, \mathbf{K}\lfloor (i_1 + 1, j_1) \rfloor)$
12 $\mathbf{p}_1 \leftarrow \left(\mathbb{p}_1, \texttt{EstimateK}(\mathbb{p}_1, \mathbf{p}_{e1}, \mathbf{p}_{e1'})\right)$ ▷ Eq. (1)
13 **if** $\mathbb{p}_2 = \lfloor \mathbb{p}_2 \rfloor$ **then** $\mathbf{p}_2 \leftarrow (\mathbb{p}_2, \mathbf{K}(i_2, j_2))$ **else**
14 **if** $i_2 = \lfloor i_2 \rfloor$ **then** $\mathbf{p}_{e2'} \leftarrow (\lfloor i_2, j_2 + 1 \rfloor, \mathbf{K}\lfloor (i_2, j_2 + 1) \rfloor)$
15 **else** $\mathbf{p}_{e2'} \leftarrow (\lfloor i_2 + 1, j_2 \rfloor, \mathbf{K}\lfloor (i_2 + 1, j_2) \rfloor)$
16 $\mathbf{p}_2 \leftarrow \left(\mathbb{p}_2, \texttt{EstimateK}(\mathbb{p}_2, \mathbf{p}_{e2}, \mathbf{p}_{e2'})\right)$
▷ compute intersections of line segment $\overline{\mathbb{p}_1\mathbb{p}_2}$ with cell diagonals:
17 $\mathbb{p}_a \overset{\text{def}}{=} (i_a, j_a) \leftarrow \texttt{line}(\mathbb{p}_1, \mathbb{p}_2) \cap \texttt{line}((i, j), (i + 1, j + 1))$
18 $\mathbb{p}_b \overset{\text{def}}{=} (i_b, j_b) \leftarrow \texttt{line}(\mathbb{p}_1, \mathbb{p}_2) \cap \texttt{line}((i, j + 1), (i + 1, j))$
19 **if** $\mathbb{p}_a \in \mathbf{c}$ **and** $\mathbb{p}_a \neq \mathbb{p}_1$ **and** $\mathbb{p}_a \neq \mathbb{p}_2$ **then**
20 **if** $i_a < i + 1/2$ **then** $\mathbf{p}_{ea} \leftarrow (i, j, \mathbf{K}(i, j))$
21 **else** $\mathbf{p}_{ea} \leftarrow (i + 1, j + 1, \mathbf{K}(i + 1, j + 1))$
22 $\mathbf{p}_a \leftarrow \left(i_a, j_a, \texttt{EstimateK}(\mathbb{p}_a, \mathbf{p}_{ea}, \mathbf{p}_c)\right)$
23 **if** $\mathbf{p}_a = \mathbf{p}_c$ **then** LIST $\leftarrow \{\Theta(\overline{\mathbf{p}_1\mathbf{p}_c}), \Theta(\overline{\mathbf{p}_c\mathbf{p}_2})\}$
24 **else if** $\nexists \mathbf{p}_b$ **then** LIST $\leftarrow \{\Theta(\overline{\mathbf{p}_1\mathbf{p}_a}), \Theta(\overline{\mathbf{p}_a\mathbf{p}_2})\}$
25 **if** $\mathbb{p}_b \in \mathbf{c}$ **and** $\mathbb{p}_a \neq \mathbb{p}_b$ **and** $\mathbb{p}_b \neq \mathbb{p}_1$ **and** $\mathbb{p}_b \neq \mathbb{p}_2$ **then**
26 **if** $i_b < i + 1/2$ **then** $\mathbf{p}_{eb} \leftarrow (i, j + 1, \mathbf{K}(i, j + 1))$
27 **else** $\mathbf{p}_{eb} \leftarrow (i + 1, j, \mathbf{K}(i + 1, j))$
28 $\mathbf{p}_b \leftarrow \left(i_b, j_b, \texttt{EstimateK}(\mathbb{p}_b, \mathbf{p}_{eb}, \mathbf{p}_c)\right)$
29 **if** $\exists \mathbf{p}_a$ **then**
30 **if** $(i_a - i_1)^2 + (j_a - j_1)^2 \le (i_b - i_1)^2 + (j_b - j_1)^2$ **then**
31 LIST $\leftarrow \{\Theta(\overline{\mathbf{p}_1\mathbf{p}_a}), \Theta(\overline{\mathbf{p}_a\mathbf{p}_b}), \Theta(\overline{\mathbf{p}_b\mathbf{p}_2})\}$
32 **else**
33 LIST $\leftarrow \{\Theta(\overline{\mathbf{p}_1\mathbf{p}_b}), \Theta(\overline{\mathbf{p}_b\mathbf{p}_a}), \Theta(\overline{\mathbf{p}_a\mathbf{p}_2})\}$
34 **else** LIST $\leftarrow \{\Theta(\overline{\mathbf{p}_1\mathbf{p}_b}), \Theta(\overline{\mathbf{p}_b\mathbf{p}_2})\}$
35 **return** LIST

When both points correspond to nodes, there are two possible cases. First (line 6), if the pair of cell entry/exit points correspond with 4-adjacent nodes, the path segment coincides with cell side $\overline{\mathbf{p}_i\mathbf{p}_{i+1}}$ (see Fig. 2(c)). In this case, the elevation of the points is already known (line 4). Second (line 7), if both cell entry/exit points correspond with opposite nodes in the cell diagonal, this produces two path segments split by the central point of the cell (see Fig. 2(b)).

When one or both cell entry/exit points do not correspond with nodes (lines 8 to 34), the partial path is composed of two or three path

Table 1
Path planning configurations used for comparative analysis.

Planner	Inclination	Any-angle	Traversability
Modified A*	$\|\alpha_z\|$	no	symmetric
Modified 3Dana	$\|\alpha_z\|$	yes	symmetric
DEM-AIA(n, s)	θ_t, ϕ_t	no	symmetric
DEM-AIA(s)	θ_t, ϕ_t	yes	symmetric
DEM-AIA(n)	θ_t, ϕ_t	no	asymmetric
DEM-AIA	θ_t, ϕ_t	yes	asymmetric

Table 2
Case-study vehicles parameters for the synthetic experiments.

Parameter	Symbol	Value
SP dimensions	$L \times W$	0.68×0.62 m
Reduced SP coefficient	ρ_{tol}	0.29
COG	$\mathbf{p}_{\text{cog}}$	(0.00, 0.03, 0.60) m
Velocity for flat terrain	v_{n}	1 m/s
Vehicle type:		
$v^{\text{u}}_{\text{ref}}$	$(\xi_{\downarrow}, \xi_{\uparrow})$	(0, 0)
$v^{\text{d}}_{\text{ref}}$	$(\xi_{\downarrow}, \xi_{\uparrow})$	(6, 2)
$v^{\text{sd}}_{\text{ref}}$	$(\xi_{\downarrow}, \xi_{\uparrow})$	(30, 10)

segments depending of the number of intersections with cell diagonals. For instance, the example in Fig. 2(a) produces three segments because the partial path intersects both cell diagonals. In any case, for non-node points, elevation is estimated from the two nodes that define the entry (lines 10 to 12) and exit (lines 14 to 16) cell sides. A similar computation is proposed for the intersections with the cell diagonals (lines 20 to 22 and lines 26 to 28) when the intersection point is not the cell center $\mathbf{p}_c$.

6. Experiments

This section describes the experimental methodology and presents an analysis of the DEM-AIA trajectory planner with respect to two comparable versions of state-of-the-art planners: A* and 3Dana. Furthermore, the effects of any-angle search and asymmetric inclination awareness are assessed by comparing limited DEM-AIA configurations (see Table 1). Finally, the application to a real-world DEMs is also discussed.

6.1. Methodology

The methodology adopted for validation and comparative analysis is consistent with related works in the AI literature. In particular, as in Muñoz et al. (2017), we compare the proposed planner with the most related antecedent method, which in our case is 3Dana, as well as with a comparable version of A*. Both algorithms have had to be modified to incorporate inclination-dependent velocity and travel time as the optimization criterion. For A*, we have adapted the DEM variant proposed by Muñoz et al. (2017). Hereinafter, we will refer to A* and 3Dana meaning our modified versions of these algorithms.

Moreover, in the comparative analysis, we assess the effect of three distinctive characteristics of DEM-based planning with variable velocity and travel-time optimization (see Table 1):

- *Segment pitch and roll estimation* versus *triangle's maximum slope*, which are needed for traversability assessment and segment velocity computation. We propose the estimation of θ_t and ϕ_t angles from DEM information using Eqs. (3) and (4). Alternatively, 3Dana uses the absolute value of the angle α_z formed by the normal vector of the cell triangle and the global Z axis, which is constant for any segment in the triangle.
- *Any-angle* versus *8-adjacency* graph-search method.
- *Asymmetric* versus *symmetric* inclination awareness for traversability. Asymmetric roll and pitch vehicle constraints yield irregular traversability combinations, like the shaded area in Fig. 5(b). Conversely, symmetric inclination constraints (i.e., the same for θ and ϕ independently of the inclination sign) produce a square region (see the red dashed line in the figure). For A* and 3Dana, only symmetric inclination is considered due to the constant $|\alpha_z|$.

As shown in Table 1, in addition to full DEM-AIA, three limited versions of the algorithm have been considered for combinations of any-angle and asymmetry characteristics: "n" denotes that search is limited to neighbor cells (i.e., `any_angle = FALSE`), and "s" that traversability limitations are symmetric. Besides, each configuration has been tested with both `heuristic = Euclidean_time` (h_e) and `heuristic = Octile_time` (h_o).

As for the UGV, we have considered three cases with the same constructive parameters but different inclination-dependence degrees for velocity (see Table 2). For constructive parameters, we have used representative values for wheeled or tracked robotic vehicles. In particular, the COG is centered with respect to the local x_{ugv} axis but has a positive y_{COG} (i.e., slightly on the forward side). Parameters L, W, $\mathbf{p}_{\text{COG}}$ and ρ_{tol} are used with Eqs. (5) and (6) to compute the asymmetric pitch and roll tip-over avoidance constraints. The resulting pitch thresholds are $\theta_{\min} = -20.1°$ and $\theta_{\max} = 25.0°$, and roll limits vary within $[-20.1°, -18.4°]$ for $\phi_{\min}$, and $[18.4°, 20.1°]$ for $\phi_{\max}$ (see Fig. 5). The symmetric configurations use $|\theta_{\min}| = |\theta_{\max}| = |\phi_{\min}| = |\phi_{\max}|$=19.1° (see Fig. 5(b)).

Furthermore, parameters v_{n}, $\xi_{\downarrow}$ and $\xi_{\uparrow}$ are used with Eq. (7) and the pitch and roll limits to compute the vehicle-dependent inclination-aware velocity constraints function $v_{\text{ref}}(\theta, \phi)$. First, the nominal velocity v_{n} has been set at 1 m/s, a representative moderate value for mobile robotics applications on natural terrain. Then, three $(\xi_{\downarrow}, \xi_{\uparrow})$ sets are considered to define $v_{\text{ref}}(\theta, \phi) = \{v^{\text{u}}_{\text{ref}}, v^{\text{d}}_{\text{ref}}, v^{\text{sd}}_{\text{ref}}\}$ functions for vehicles with different inclination-dependence degrees:

- *Uniform velocity* ($v^{\text{u}}_{\text{ref}}$). Velocity is not dependent of terrain inclination: $(\xi_{\downarrow}, \xi_{\uparrow} = 0)$, so $v = v_{\text{n}}$. This case is analogous to the path length as optimization criterion.
- *Dependence* ($v^{\text{d}}_{\text{ref}}$). Velocity is moderately conditioned by terrain inclination. We have tested asymmetric values $\xi_{\downarrow} = 6$ and $\xi_{\uparrow} = 2$.
- *Stronger dependence* ($v^{\text{sd}}_{\text{ref}}$), with $\xi_{\downarrow} = 30$ and $\xi_{\uparrow} = 10$.

The example in Fig. 5 corresponds to $v^{\text{sd}}_{\text{ref}}(\theta, \phi)$. Moreover, Fig. 6 compares the three $v_{\text{ref}}(\theta, \phi)$ functions for $\phi = 0°$. In the cases of inclination-dependence, the asymmetry yields faster velocities for ascending than for descending because we have found empirically that velocity variations (e.g., slowing down or turning) in downwards motion are more critical for traction loss due to the effect of gravity.

Regarding terrain, we generated synthetic DEMs of size 500 × 500 cells with the diamond square algorithm (Fink et al., 2019), a fractal-based terrain generation technique, using elevations in the range of 0 to 50 m and spatial resolution $\delta = 1$ m. From these random DEMs, we have chosen three illustrative ones with at least 50% of traversable cells, and distinctive and realistic terrain profiles, which are shown in Fig. 7.

For each DEM, we have generated 500 $(\mathbf{n}_{\text{s}}, \mathbf{n}_{\text{g}})$ pairs of start-goal nodes. To make path lengths comparable, all $\mathbf{n}_{\text{s}}$ nodes have been randomly chosen in the lower fifth of the Y coordinate and all $\mathbf{n}_{\text{g}}$ in the higher fifth (see the south and north areas marked in each DEM of Fig. 7). All in all, the total number of path planning experiments performed in the comparative analysis has been 54000 (i.e., 3 synthetic DEMs × 500 $(\mathbf{n}_{\text{s}}, \mathbf{n}_{\text{g}})$ pairs × 6 path planning configurations × 2 heuristic functions × 3 vehicles with different inclination-dependence degrees for velocity).

Path planning has been executed on a 3.6 GHz Intel© Octa-Core i7-9700K processor and 16 GB RAM running Windows 10. In particular, MathWorks® MATLAB R2022a with the MATLAB Coder application has been used to generate executable code (MEX files). For a fair comparison, all algorithms have been implemented with the same software.

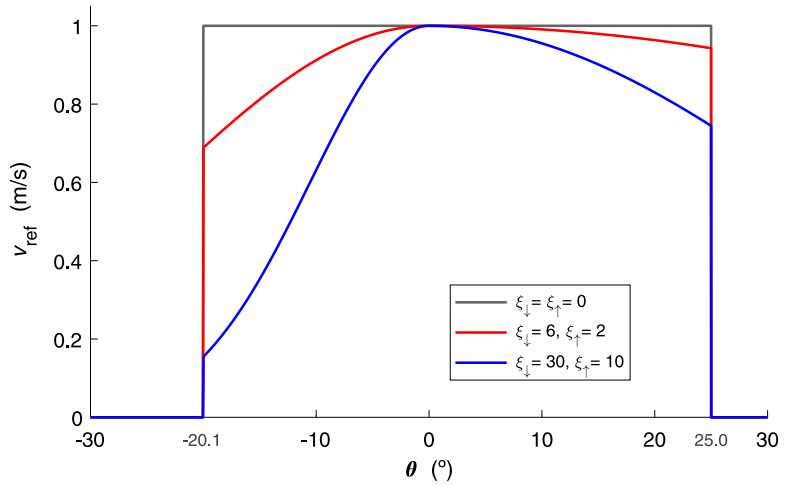


Fig. 6. Representation of $v_{ref}(\theta, \phi)$ functions for $\phi = 0°$: v^{u}_{ref} (black), v^{d}_{ref} (red) and v^{sd}_{ref} (blue).

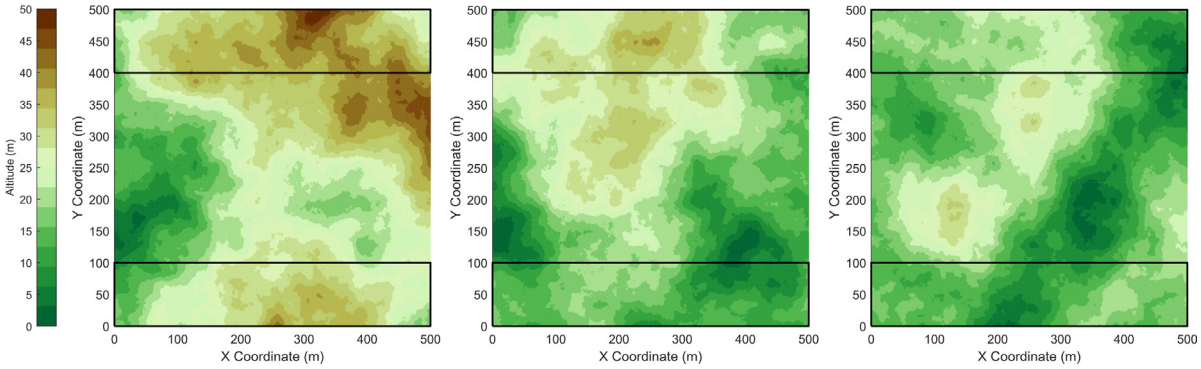


Fig. 7. The three 500 × 500 cells synthetic DEMs used for the experimental analysis. 1500 $(\mathbf{n}_s, \mathbf{n}_g)$ pairs of start-goal nodes are defined by randomly generating 500 pairs from south and north areas marked, respectively, in each DEM.

6.2. Analysis and discussion

For the comparative analysis, the following performance indices have been used:

- Number of iterations in the search algorithm.
- Number of explored nodes.
- Number of re-expanded nodes.
- Computing time of the planning algorithm.
- Total travel time which is the optimization criterion.
- Total absolute heading turn.

Table 3 summarizes comparative results for the three vehicles by presenting the median of three relevant performance indices (computing time, total travel time and total turn) for each planner. Values are shown for both heuristic functions (h_e and h_o). In addition, the table shows the best mark obtained by the planners for each performance index, which is the percentage of experiments where the planner has achieved the best score for an index.

Moreover, Figs. 8 and 9 offer box-and-whiskers plots that represent the quartiles of data distribution for the 1500 $(\mathbf{n}_s, \mathbf{n}_g)$ pairs of analyzed start-goal nodes, where the line of a center-box represents the median value (second quartile) of dataset. For every index, graphs are given for the three vehicles with different inclination-dependence degrees for $v_{ref}(\theta, \phi)$. For each planner (indicated under the abscissa axis), two boxes are shown: blue for Euclidean-time and red for octile-time.

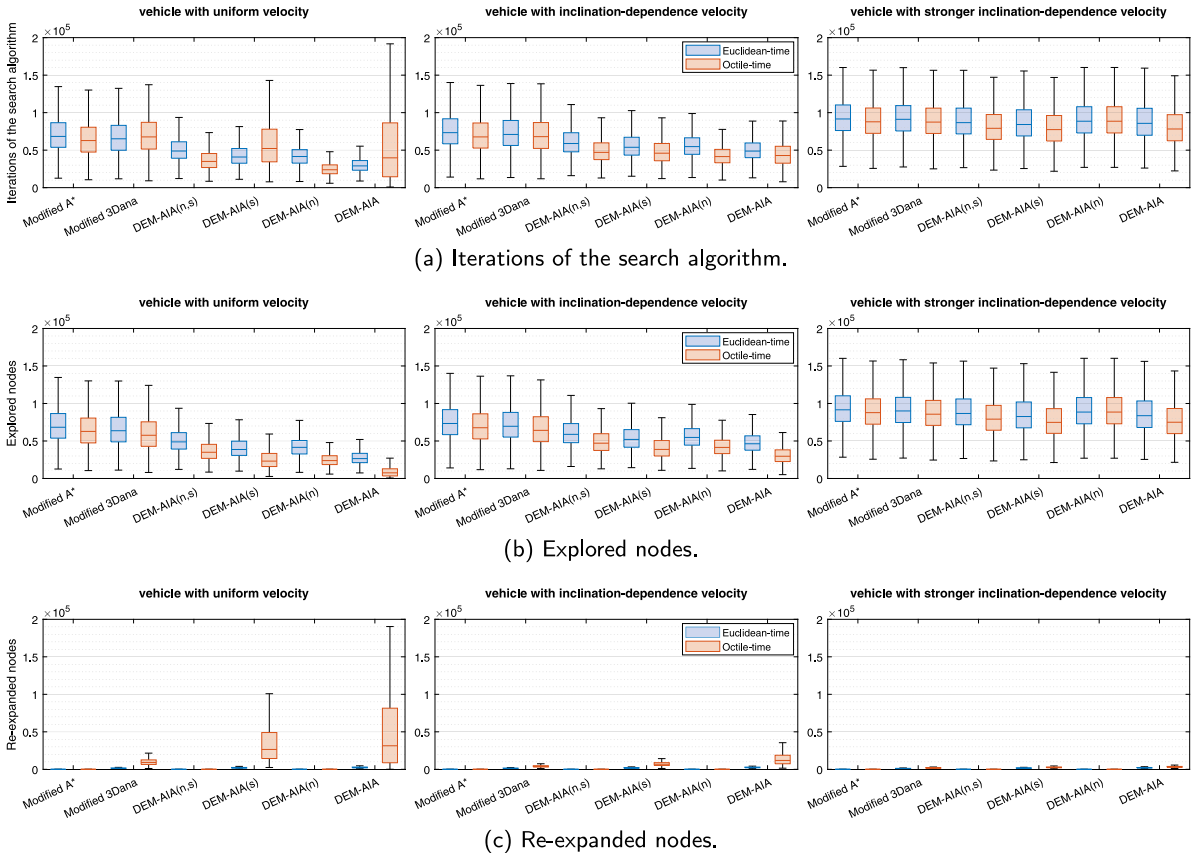
6.2.1. Total number of iterations, explored nodes and re-expanded nodes

Fig. 8(a) shows the number of iterations performed during the node expansion process for each vehicle type and planner. The number of explored (see Fig. 8(b)) and re-expanded (see Fig. 8(c)) nodes determines the total number of iterations. Only any-angle planners support re-expansion of previously explored nodes.

The heuristic function influences the total number of iterations. The octile-time heuristic reduces the number of explored nodes (red boxes in Fig. 8(b)). This can be explained because the graph-search process is based on an 8-adjacency nodes connectivity and, therefore, the octile distance is an appropriate estimation of the total path length. For any-angle planners with the v^{u}_{ref} vehicle, the Euclidean-time heuristic leads to a smaller number of re-expanded nodes (blue boxes in Fig. 8(c)). This can be explained because the resulting trajectories tend to rectilinear paths and, therefore, Euclidean distance is a better estimation of total path length than octile in this case. Consequently, the Euclidean-time heuristic reveals a smaller number of iterations for any-angle planners with the v^{u}_{ref} vehicle (blue boxes in Fig. 8(a)(left)).

For the octile-time heuristic function (see red boxes in Fig. 8(c)), the number of re-expanded nodes decreases significantly for the stronger inclination-dependent vehicle (v^{sd}_{ref}). This could be explained because, relief inclination changes have more impact on vehicle velocity reductions, which favors the search for faster non-rectilinear trajectories.

Moreover, the double-valued estimation of segment inclination – i.e., pitch and roll angles – and the asymmetric inclination awareness lead to a marked decrease in the number of iterations of the DEM-AIA algorithm compared to A* and 3Dana planners. This difference is reduced for the v^{sd} vehicle, since the high velocity reductions that the natural terrain relief would imply, could lead to an increase of the search space.

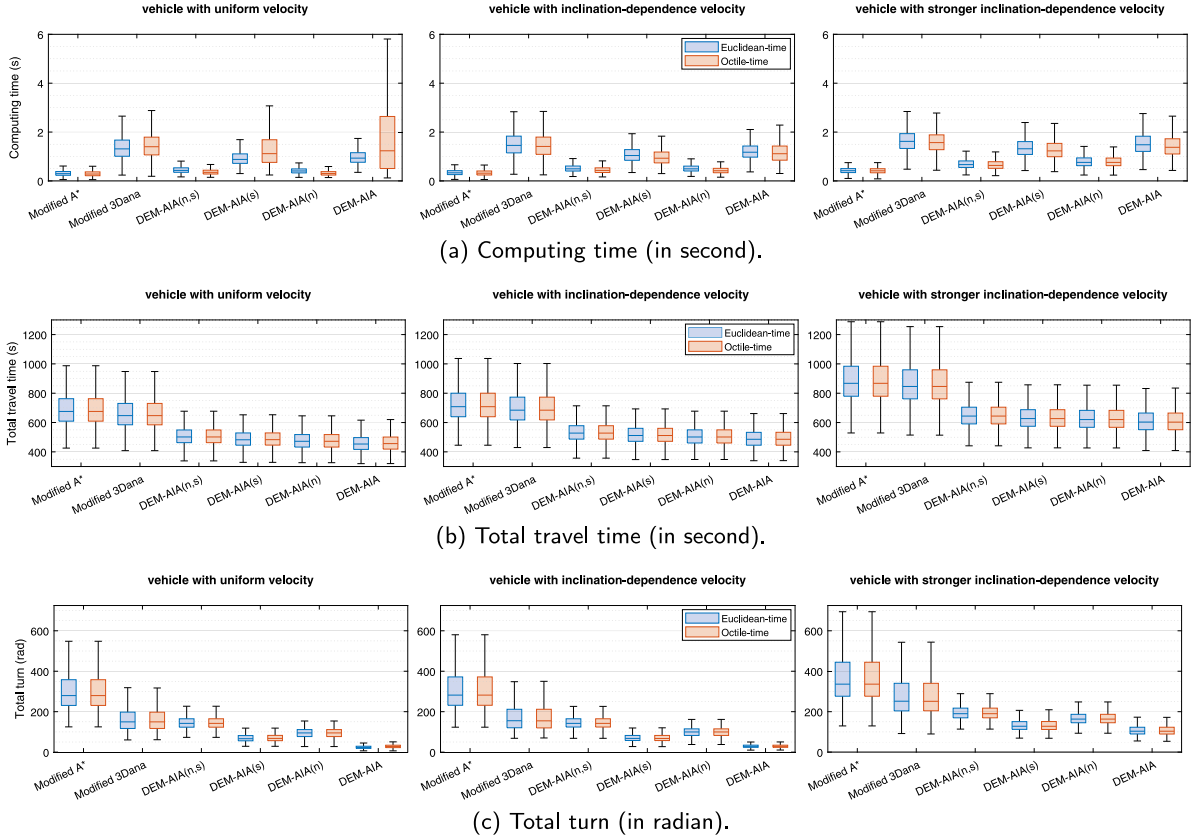


(a) Iterations of the search algorithm.

(b) Explored nodes.

(c) Re-expanded nodes.

Fig. 8. Box-and-whiskers plots of the analyzed performance indexes for path planning configurations: iterations of the search algorithm, and explored and re-expanded nodes.



(a) Computing time (in second).

(b) Total travel time (in second).

(c) Total turn (in radian).

Fig. 9. Box-and-whiskers plots of analyzed performance indexes for path planning configurations: computing time (in second), total travel time (in second) and total turn (in radian).

Table 3
Median value and best mark of the three most relevant performance indices for each path planning configuration, heuristic function and inclination-dependence degree for velocity. These indices are computing time, total travel time and total turn.

Path planning configuration	Computing time (ms) h_e/h_o	Best mark (%) h_e/h_o	Total travel time (s) h_e/h_o	Best mark (%) h_e/h_o	Total turn (rad) h_e/h_o	Best mark (%) h_e/h_o
Vehicle with uniform velocity (v^{u}_{ref})						
Modified A*	298/282	8.3/38.9	675.8	–	279.6	–
Modified 3Dana	1312/1500	–	647.3/647.2	–	149.6/149.6	–
DEM-AIA(n, s)	431/351	0.1/6.0	501.7	–	142.2	–
DEM-AIA(s)	881/1116	–	482.5/483.2	–	66.4/67.8	–
DEM-AIA(n)	408/302	–/35.5	472.1	–	95.0	–
DEM-AIA	933/1233	–/11.2	453.5/456.4	91.4/8.7	23.2/28.0	76.7/23.4
Vehicle with inclination-dependence velocity (v^{d}_{ref})						
Modified A*	332/316	15.3/58.9	708.2	–	282.0	–
Modified 3Dana	1454/1412	–	684.4/684.4	–	155.3/154.8	–
DEM-AIA(n, s)	495/431	0.1/9.3	528.0	–	142.2	–
DEM-AIA(s)	1040/929	–	511.7/511.6	–	67.8/67.5	–
DEM-AIA(n)	495/417	0.1/16.3	501.4	–	99.8	–
DEM-AIA	1175/1114	–	485.8/485.8	42.0/58.5	28.5/28.5	48.6/51.9
Vehicle with stronger inclination-dependence velocity (v^{sd}_{ref})						
Modified A*	419/414	30.7/62.7	867.5	–	336.2	–
Modified 3Dana	1619/1569	–	846.1/846.2	–	251.8/251.3	–
DEM-AIA(n, s)	668/634	0.2/6.3	644.2	–	190.1	–
DEM-AIA(s)	1316/1230	–	627.1/626.9	–	128.0/127.8	1.2/1.1
DEM-AIA(n)	759/758	–/0.1	620.3	–	163.4	0.1/0.1
DEM-AIA	1480/1377	–	603.3/602.9	40.9/72.1	103.4/103.4	54.4/56.5

6.2.2. Computing time

The experimental results for computing time are summarized in Fig. 9(a) and Table 3(left). Computing time is affected by the total number of iterations as well as the computation of the cost and heuristic functions in each iteration. Thus, evaluating the cost of every segment in the line-of-sight for any-angle search produces an average overhead of 176% (456 ms against 1261 ms). This average overhead reaches 330% when comparing A* with 3Dana. DEM-AIA configurations with disabled any-angle search have longer computing times than A* (between 30% and 60% longer), mainly due to the double estimation of the segment inclinations: pitch θ_t and roll ϕ_t. However, any-angle DEM-AIA configurations have shorter times than 3Dana (between 18% and 40% less), which could be explained because the pitch and roll estimation and the asymmetric inclination awareness allow considering a larger number of traversable cells, thus reducing the number of explored nodes.

In general, the octile-time heuristic produces similar but slightly shorter computing times with respect to Euclidean, with the exception of any-angle planners with the v^{u}_{ref} vehicle (i.e. equivalent to path length optimization). This is consistent with the results discussed for the same case regarding the number of iterations.

For all vehicles, A* with h_o scores the best mark (between approximately 39.7% and 60%) for computing time. With h_o, non any-angle DEM-AIA with asymmetric inclination awareness has a similar best mark (35.5%) and computing time (approximately 300 ms) as A* for uniform velocity. Nevertheless, full DEM-AIA achieves an average 15% (1287 ms against 1514 ms) improvement in computing time with respect to 3Dana for the two inclination dependent vehicles.

6.2.3. Total travel time and total turn

Total travel time and total turn indicate the quality of the resulting trajectories. Interestingly, the sum of best marks for both indices in Table 3 (middle and right) is always larger than 100%. The analysis of individual experiments reveals that in some cases (0.14% for v^{u}_{ref}, 0.54% for v^{d}_{ref}, and 13.14% for v^{sd}_{ref}), DEM-AIA finds the same trajectory for both h_e and h_o. The larger coincidence for v^{sd}_{ref} can be explained by a lower number of traversable (i.e., eligible) cells. Precisely, the differences between h_e and h_o indices indicate that any-angle planners may lead to different results depending on the heuristic function, since these algorithms do not fulfill the admissibility property and, therefore, an optimal solution is not guaranteed.

Table 4
Vehicle parameters for the real-world DEM experiments.

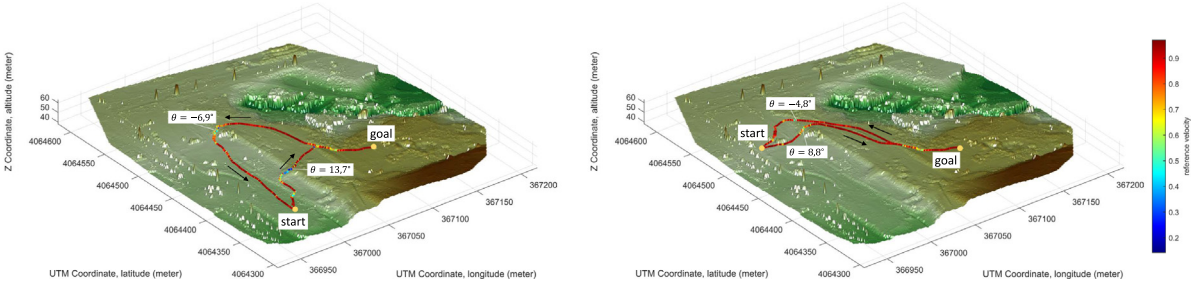
Parameter	Symbol	Value
SP dimensions	$L \times W$	2.025×1.210 m
Reduced SP coefficient	ρ_{tol}	0.8
COG	$\mathbf{p}_{cog}$	(0.00, 0.3, 0.97) m
Velocity for flat terrain	v_n	1 m/s
Inclination-dependence	$(\xi_\downarrow, \xi_\uparrow)$	(300, 100)

Figs. 9(b–c) reveal a similarity between total travel time and total turn performances. 3Dana presents a reduction in both indices with respect to A*. Nevertheless, all DEM-AIA configurations improve these results because pitch and roll estimations are less restrictive than the maximum slope of the triangles. Besides, DEM-AIA configurations with asymmetric inclination awareness further improve this results. All in all, full DEM-AIA achieves the lowest total travel times and total turn in all cases (see Table 3).

6.3. DEM-AIA applied to a real-world DEM

The DEM-AIA planner has been applied to real-world DEMs generated from drone photogrammetry using Pix4D-mapper with three different resolutions $\delta = \{0.5, 1, 2\}$m. The terrain corresponds to the Experimentation Area in New Technologies for Emergency Intervention of Universidad de Málaga (Spain) (UMA, 2022), which is used for realistic SAR exercises with disaster robots, where safe and effective off-road path planning is critical (Bravo-Arrabal et al., 2021; Toscano-Moreno et al., 2022). Moreover, UGV parameters (see Table 4) have been obtained empirically from skid-steer Rover J8, developed by Argo (Kitchener, Ontario, Canada). The resulting pitch thresholds $\theta_{min} = -13.3°$ and $\theta_{max} = 40.5°$ reveal highly asymmetric inclination constraints. For roll limits, these vary within $[-7.13°, -6.73°]$ for ϕ_{min}, and $[6.73°, 7.13°]$ for ϕ_{max}. For inclination dependence parameters, high values of $\xi_\downarrow$ and $\xi_\uparrow$ imply penalization of speed on steep slopes, especially downhill, as required in missions such as disaster victim evacuation (Toscano-Moreno et al., 2022).

Next, asymmetric behavior and the effect of DEM resolution on DEM-AIA planner is discussed. Moreover, a comparative with original 3Dana is evaluated with uniform velocity.



(a) Effect of asymmetric traversability limits (b) Effect of asymmetric velocities for travel-time optimization.

Fig. 10. Illustrations of the DEM-AIA planner applied to a real-world DEM ($\delta = 1$ m).

Table 5
DEM-AIA trajectory planning with different DEM resolutions (to goal/from goal).

δ (m)	0.5	1	2
Example shown in Fig. 10(a)			
Distance (m)	197.1/339.8	182.3/334.2	182.9/327.7
Travel time (s)	317/420	282/403	256/392
Runtime (ms)	1421/1722	444/597	188/251
Example shown in Fig. 10(b)			
Distance (m)	215.3/237.0	210.6/234.3	207.0/226.7
Travel time (s)	242/287	234/276	229/269
Runtime (ms)	2454/1105	740/363	213/178

Table 6
DEM-AIA versus 3Dana planners with uniform velocity and $\delta = 1$ m (to goal/from goal).

Planner	DEM-AIA	3Dana
Example shown in Fig. 11(a)		
Distance (m)	132.7/188.5	411.9/413.3
Runtime (ms)	183/899	6400/1473
Iterations	2031/10499	31680/12297
Explored nodes	1235/9852	18735/10844
Re-expanded nodes	800/647	13248/1458
Example shown in Fig. 11(b)		
Distance (m)	180.6/203.4	276.6/276.1
Runtime (ms)	3065/813	7940/1355
Iterations	12771/6784	33057/10317
Explored nodes	2802/5624	15919/9171
Re-expanded nodes	10387/1166	17440/1164

6.3.1. Asymmetric behavior and DEM resolution

Two representative round-trip trajectories for $\delta = 1$ m are illustrated in Fig. 10, where a color scale indicates planned segment velocities. These examples, where the maximum pitch value is highlighted for each trajectory, show how the planner reduces velocities on uphill and downhill slopes. Moreover, Table 5 summarizes results for the three DEM resolutions.

The examples are illustrative of two major effects of asymmetric inclination awareness provided by DEM-AIA. First, in Fig. 10(a), the maximum slope in the uphill trajectory to the goal point ($\theta = 13.7°$) would exceed the downhill limit $\theta_{\min}$, so a different return trajectory is found with maximum downhill slope $\theta = -6.9°$. Thus, asymmetric traversability achieves a faster admissible trajectory to the goal point (282 s) against the time that would have been required to travel the return path in reverse (355 s, computed by changing the signs of θ and ϕ in the trajectory segments). The second example (see Fig. 10(b)) illustrates asymmetric inclination-dependent velocities. In this case, the maximum slope of the uphill trajectory (8.8°) would be admissible for returning. However, the planner produces a longer return path (234.3 m against 210.6 m, as shown in Table 5) because travel time is shorter: 276 s against 310 s for the initial path in reverse. This is explained by vehicle parameters $\xi_{\downarrow} > \xi_{\uparrow}$ in Table 4, which yield lower velocities for steeper downhill slopes in Eq. (7).

As for the effect of DEM resolution (see Table 5), the average computing time (runtime) grows with higher resolution on account of the increase in the number of explored nodes. In these examples, average runtime approximately triples when δ is halved. Moreover, results show a slight increase of travel time with higher resolution, which can be explained by more slope variations due to a larger number of segments. The impact of resolution is also related to the quality of the photogrammetric process: we have found significant differences in travel time, and even in the resulting paths, for DEMs generated with extreme values (not given in the table) $\delta = 0.25$ m, which might indicate noisy data, and $\delta = 4$ m, which can be explained by excessive terrain filtering. Furthermore, for all resolutions in Table 5, there is a significant difference in computing times between the trajectories to and from the goal. In this case, shorter runtimes can be explained by reduced traversability conditions around the initial node (nodes not satisfying the UGV inclination constraints), which implies a more directed node expansion towards the resulting trajectory.

6.3.2. Application to distance optimization

A comparison of DEM-AIA and the original 3Dana is illustrated in Fig. 11 and Table 6. In this case, DEM-AIA has been limited to uniform velocity for geodetic distance optimization. In these examples, both planners achieve round-trip trajectories between the start and goal points. The paths obtained by DEM-AIA produce shorter distances than 3Dana because asymmetric inclination constraints allow exploring nodes that are discarded by a unique symmetric threshold. For both planners, computing time is affected by two factors: path length, which is related to the number of explored nodes, and the number of re-expanded nodes for sub-optimal search. Nevertheless, computing times are shorter for DEM-AIA even with the extra computational load of estimating θ and ϕ for each path segment against a single $|\alpha_z|$ for cell triangles. This can be explained because asymmetry can direct the node expansion process more quickly to the resulting path, which reduces the number of iterations. In 3Dana, the round-trip paths are slightly different, which is not due to symmetry constraints, but to the fact that both planners are sub-optimal and results can differ according to node re-expansion.

Furthermore, the comparison between Figs. 10 and 11 shows the differences between time- and distance-based optimization with DEM-AIA. Interestingly, even if the return path in Fig. 11(a) – 188.5 m – is notably shorter than that in Fig. 10(a) – 334.2 m –, steep slopes of up to 37.7° would impose strong speed limitations for travel time. Similarly, the initial path in Fig. 11(b) moves quite straight to the goal, but crosses two steep uphill slopes of up to 31.2°, whereas the maximum slope of 8.8° in Fig. 10(a) allows for faster travel time.

7. Public repository

The executable codes – i.e., MEX files compiled for MathWorks® MATLAB R2022a –, and four MAT files consisting of the three synthetic DEMs used for the comparative analysis and the real-world DEM with $\delta = 1$ m discussed in Section 6 are available in the GitHub repository: https://github.com/mToscanoMoreno/DEM-AIA. Besides, a main script `test_DEMAIA.m` is included to test and facilitate the use of the DEM-AIA planner for the community.

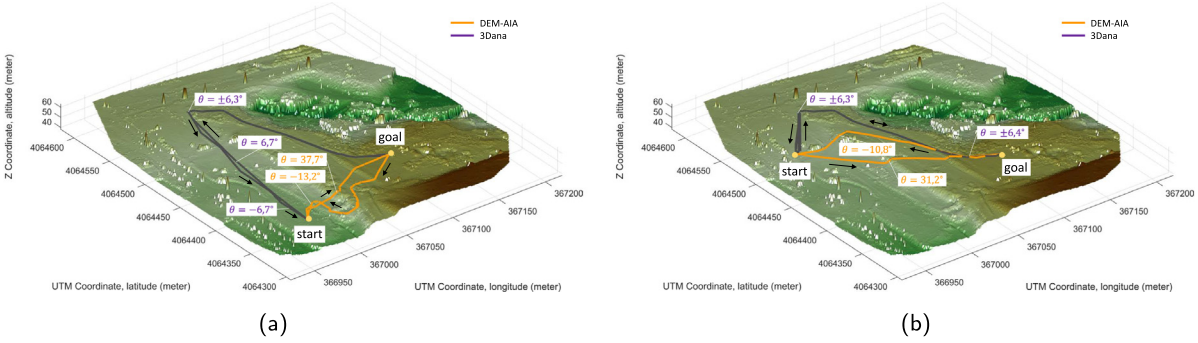


Fig. 11. Illustration with comparative of the DEM-AIA and 3Dana planners applied to a real-world DEM with uniform velocity and $\delta = 1$ m.

8. Conclusions

This work has proposed a DEM-based trajectory planner for ground vehicles (DEM-AIA) that accounts for path inclination (i.e., pitch and roll estimated from DEM data) and asymmetry (i.e., traversability, partial costs and segment speeds can be different for uphill and downhill). The new intelligent planner extends the 3Dana (Muñoz et al., 2017) any-angle graph search algorithm. The major distinctive features of our contribution are the following: (i) Our method computes a time sub-optimal trajectory between the start and goal points with non-uniform velocity by assigning a feasible velocity to path segments; (ii) a novel formulation that uses cell-triangle geometry and motion direction to estimate inclination for any-angle planning that computes both pitch and roll angles for the actual segments crossing cell triangles; and (iii) a vehicle-dependent non-linear function of pitch and roll that can be implemented as a look-up table and defines admissible velocity by incorporating asymmetric inclination-dependent forward speed limits, tip-over stability constraints, and feasible inclination-aware speed rates with respect to the maximum nominal speed for flat terrain.

The experimental analysis has been based on 54000 experiments where we have compared our algorithm with a DEM variant of A* and 3Dana (Muñoz et al., 2017), which have been adapted to travel-time optimization and inclination-dependent velocity. In the experimental analysis we have also evaluated the effect of any-angle search, consideration of asymmetry, and octile- vs. Euclidean-time heuristics.

Pitch and roll estimation of segment inclination and asymmetric inclination awareness have led to a significant reduction in the number of iterations of the DEM-AIA algorithm compared to A* and 3Dana planners. This reduction has been specially noticeable for vehicles with moderate inclination velocity constraints, where the number of traversable nodes is larger. Moreover, computing time has been slightly faster using the octile-time heuristic function, except for any-angle planners with uniform velocity vehicles (i.e. equivalent to path length optimization) due to larger number of re-expanded nodes produced by that heuristic. For all vehicles, A* with the octile-time heuristic scored the best mark for computing time. However, DEM-AIA has shown faster computing times than 3Dana, due to the reduction in the number of explored nodes.

Experimental analysis has revealed a correlation between travel-time and total-turn performance. The asymmetric inclination awareness feature of DEM-AIA has achieved the fastest total travel times and total turn in all cases, because pitch and roll estimations allow for realistic inclination constraints. These performance has also been observed in examples with real off-road DEMs.

Our future research will validate the proposed trajectory planning method for planning of a real skid steer UGV on off-road natural terrain. It will also be interesting to explore the specification of high-level missions on irregular terrain as sequences of task by means of modal temporal logic, such as linear temporal logic (LTL) (Fainekos et al., 2009), where tasks can be planned by DEM-AIA with inclination awareness and travel-time optimization. Furthermore, the inclination-aware velocity constraints functions defined in this work are vehicle-dependent but not consider different soil characteristics. The effect of characteristics such as humidity, rugosity and compactness can be considered by classifying soil types from visual data in digital terrain models and defining soil-wise velocity constraints functions for the vehicle.

CRediT authorship contribution statement

Manuel Toscano-Moreno: Investigation, Methodology, Software, Formal analysis, Validation, Visualization, Writing – original draft, Writing – review & editing. **Anthony Mandow:** Conceptualization, Funding acquisition, Investigation, Methodology, Supervision, Validation, Visualization, Writing – original draft, Writing – review & editing. **María Alcázar Martínez:** Validation, Writing – review & editing. **Alfonso García-Cerezo:** Funding acquisition, Conceptualization, Methodology, Project administration, Supervision.

Declaration of competing interest

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.

Data availability

Executable code (MEX files compiled for MathWorks MATLAB R2022a) and four MAT files with synthetic and real-world DEMs are available in GitHub: https://github.com/mToscanoMoreno/DEM-AIA.

Acknowledgments

This work has been partially funded by the Spanish projects RTI2018-093421-B-I00 and PID2021-122944OB-I00. The first author is partially supported by predoctoral grant BES-2016-077022 (Spanish Ministry of Science and Innovation, co-financed by the European Social Fund). Funding for open access charge: Universidad de Málaga / CBUA. The authors are grateful to ATyges Ingeniería (Málaga, Spain) for providing aerial photogrammetry. Finally, we acknowledge the support from our colleagues of the UMA Robotics and Mechatronics Group.

References

Adámek, R., Rajchl, M., Kriváneк, V., Grepl, R., 2022. A design of a global path planner for nonholonomic vehicle based on dynamic simulations. Lecture Notes in Comput. Sci. 13207 LNCS, 127–144. http://dx.doi.org/10.1007/978-3-030-98260-7_8.

Atiyah, A.N., Adzhar, N., Jaini, N.I., 2021. An overview: on path planning optimization criteria and mobile robot navigation. J. Phys. Conf. Ser. 1988 (1), 012036. http://dx.doi.org/10.1088/1742-6596/1988/1/012036.

Borges, C.D.B., Almeida, A.M.A., Paula Júnior, I.C., Sá Junior, J.J.M., 2019. A strategy and evaluation method for ground global path planning based on aerial images. Expert Syst. Appl. 137, 232–252. http://dx.doi.org/10.1016/j.eswa.2019.06.067.

Bravo-Arrabal, J., Toscano-Moreno, M., Fernandez-Lozano, J., Mandow, A., Gomez-Ruiz, J., García-Cerezo, A., 2021. The internet of cooperative agents architecture (X-IoCA) for robots, hybrid sensor networks, and MEC centers in complex environments: A search and rescue case study. Sensors 21 (23), 7843. http://dx.doi.org/10.3390/s21237843.

Choi, S., Park, J., Lim, E., Yu, W., 2012. Global path planning on uneven elevation maps. In: International Conference on Ubiquitous Robots and Ambient Intelligence. pp. 49–54. http://dx.doi.org/10.1109/URAI.2012.6462928.

Choi, S., Yu, W., 2011. Any-angle path planning on non-uniform costmaps. In: IEEE International Conference on Robotics and Automation. pp. 5615–5621. http://dx.doi.org/10.1109/ICRA.2011.5979769.

Colas, F., Mahesh, S., Pomerleau, F., Liu, M., Siegwart, R., 2013. 3D path planning and execution for search and rescue ground robots. In: IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 722–727. http://dx.doi.org/10.1109/IROS.2013.6696431.

Daniel, K., Nash, A., Koenig, S., Felner, A., 2010. Theta*: Any-angle path planning on grids. J. Artif. Intell. Res. 39, 533–579. http://dx.doi.org/10.1613/jair.2994.

Delmerico, J., Mueggler, E., Nitsch, J., Scaramuzza, D., 2017. Active autonomous aerial exploration for ground robot path planning. IEEE Robot. Autom. Lett. 2 (2), 664–671. http://dx.doi.org/10.1109/LRA.2017.2651163.

Dimri, S.C., 2015. A simple and efficient algorithm for line and polygon clipping in 2-D computer graphics. Int. J. Comput. Appl. 127, 31–34. http://dx.doi.org/10.5120/ijca2015906352.

Dolgov, D., Thrun, S., Montemerlo, M., Diebel, J., 2010. Path planning for autonomous vehicles in unknown semi-structured environments. Int. J. Robot. Res. 29 (5), 485–501. http://dx.doi.org/10.1177/0278364909359210.

Fabian, S., Kohlbrecher, S., Von Stryk, O., 2020. Pose prediction for mobile ground robots in uneven terrain based on difference of heightmaps. In: IEEE International Symposium on Safety, Security, and Rescue Robotics. pp. 49–56. http://dx.doi.org/10.1109/SSRR50563.2020.9292574.

Fainekos, G.E., Girard, A., Kress-Gazit, H., Pappas, G.J., 2009. Temporal logic motion planning for dynamic robots. Automatica 45, 343–352. http://dx.doi.org/10.1016/J.AUTOMATICA.2008.08.008.

Fedorenko, R., Gabdullin, A., Fedorenko, A., 2018. Global UGV path planning on point cloud maps created by UAV. In: 2018 3rd IEEE International Conference on Intelligent Transportation Engineering. ICITE, pp. 253–258. http://dx.doi.org/10.1109/ICITE.2018.8492584.

Fink, W., Baker, V., Brooks, A.-W., Flammia, M., Dohm, J., Tarbell, M., 2019. Globally optimal rover traverse planning in 3D using Dijkstra's algorithm for multi-objective deployment scenarios. Planet. Space Sci. http://dx.doi.org/10.1016/j.pss.2019.104707.

Gabrlik, P., Lazna, T., Jilek, T., Sladek, P., Zalud, L., 2021. An automated heterogeneous robotic system for radiation surveys: Design and field testing. J. Field Robotics http://dx.doi.org/10.1002/rob.22010.

Garrido, S., Álvarez, D., Martín, F., Moreno, L., 2019. An anisotropic fast marching method applied to path planning for mars rovers. IEEE Aerosp. Electr. Syst. Mag. 34 (7), 6–17. http://dx.doi.org/10.1109/MAES.2019.2924124.

Garrido, S., Malfaz, M., Blanco, D., 2013. Application of the fast marching method for outdoor motion planning in robotics. Robot. Auton. Syst. 61 (2), 106–114. http://dx.doi.org/10.1016/j.robot.2012.10.012.

Hart, P., Nilsson, N., Raphael, B., 1968. A formal basis for the heuristic determination of minimum cost paths. IEEE Trans. Syst. Sci. Cybern. 4 (2), 100–107. http://dx.doi.org/10.1109/TSSC.1968.300136.

Hedrick, G., Ohi, N., Gu, Y., 2020. Terrain-aware path planning and map update for mars sample return mission. IEEE Robot. Autom. Lett. 5 (4), 5181–5188. http://dx.doi.org/10.1109/LRA.2020.3005123.

Hines, T., Stepanas, K., Talbot, F., Sa, I., Lewis, J., Hernandez, E., Kottege, N., Hudson, N., 2021. Virtual surfaces and attitude aware planning and behaviours for negative obstacle navigation. IEEE Robot. Autom. Lett. 6 (2), 4048–4055. http://dx.doi.org/10.1109/LRA.2021.3065302.

Hu, J., Hu, Y., Lu, C., Gong, J., Chen, H., 2022. Integrated path planning for unmanned differential steering vehicles in off-road environment with 3D terrains and obstacles. IEEE Trans. Intell. Transp. Syst. 23 (6), 5562–5572. http://dx.doi.org/10.1109/TITS.2021.3054921.

Hu, H., Zhang, K., Tan, A.H., Ruan, M., Agia, C.G., Nejat, G., 2021. A sim-to-real pipeline for deep reinforcement learning for autonomous robot navigation in cluttered rough terrain. IEEE Robot. Autom. Lett. 6 (4), 6569–6576. http://dx.doi.org/10.1109/LRA.2021.3093551.

Hua, C., Niu, R., Yu, B., Zheng, X., Bai, R., Zhang, S., 2022. A global path planning method for unmanned ground vehicles in off-road environments based on mobility prediction. Machines 10 (5), http://dx.doi.org/10.3390/machines10050375.

Huang, Y., Zhou, Y., Xiong, Z., 2021. Path planning for nuclear emergency robot in radiation environment with uneven terrain. Lecture Notes in Comput. Sci. 13016 LNAI, 680–690. http://dx.doi.org/10.1007/978-3-030-89092-6_62.

Isher, M.K., Sapkota, R., Maharjan, S., 2022. Multi-objective optimization of feedback parameters of wheel terrain interaction of an autonomous vehicle. Int. J. Mech. Eng. Robot. Res. 11 (1), 43–50. http://dx.doi.org/10.18178/ijmerr.11.1.43-50.

Ishigami, G., Nagatani, K., Yoshida, K., 2011. Path planning and evaluation for planetary rovers based on dynamic mobility index. In: IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 601–606. http://dx.doi.org/10.1109/IROS.2011.6094768.

Ji, Y., Tanaka, Y., Tamura, Y., Kimura, M., Umemura, A., Kaneshima, Y., Murakami, H., Yamashita, A., Asama, H., 2019. Adaptive motion planning based on vehicle characteristics and regulations for off-road UGVs. IEEE Trans. Ind. Inform. 15 (1), 599–611. http://dx.doi.org/10.1109/TII.2018.2870662.

Jing, Y., Luo, C., Liu, G., 2022. Multiobjective path optimization for autonomous land levelling operations based on an improved MOEA/D-ACO. Comput. Electron. Agric. 197, http://dx.doi.org/10.1016/j.compag.2022.106995.

Kubota, T., Kuroda, Y., Kunii, Y., Yoshimitsu, T., 2001. Path planning for newly developed microrover. In: IEEE International Conference on Robotics and Automation, Vol. 4. pp. 3710–3715 vol.4. http://dx.doi.org/10.1109/ROBOT.2001.933195.

Kumar, J., Dutta, A., 2022. Energy optimal motion planning of a 14-DOF biped robot on 3D terrain using a new speed function incorporating biped dynamics and terrain geometry. Robotica 40 (2), 250–278. http://dx.doi.org/10.1017/S0263574721000515.

Lauterbach, H.A., Koch, C.B., Hess, R., Eck, D., Schilling, K., Nüchter, A., 2019. The Eins3D project — Instantaneous UAV-based 3D mapping for search and rescue applications. In: IEEE International Symposium on Safety, Security, and Rescue Robotics. pp. 1–6. http://dx.doi.org/10.1109/SSRR.2019.8848972.

Miranda, V.R.F., Mozelli, L.A., Neto, A.A., Freitas, G.M., 2019. On the robust longitudinal trajectory tracking for load transportation vehicles on uneven terrains. In: International Conference on Advanced Robotics. pp. 320–325. http://dx.doi.org/10.1109/ICAR46387.2019.8981641.

Morales, J., Martinez, J.L., Mandow, A., Seron, J., Garcia-Cerezo, A.J., 2013. Static tip-over stability analysis for a robotic vehicle with a single-axle trailer on slopes based on altered supporting polygons. IEEE/ASME Trans. Mechatronics 18 (2), 697–705. http://dx.doi.org/10.1109/TMECH.2011.2181955.

Muñoz, P., R-Moreno, M.D., Castaño, B., 2017. 3Dana: A path planning algorithm for surface robotics. Eng. Appl. Artif. Intell. 60, 175–192. http://dx.doi.org/10.1016/j.engappai.2017.02.010.

Murphy, R.R., Tadokoro, S., Kleiner, A., 2016. Disaster Robotics. In: Siciliano, B., Khatib, O. (Eds.), Springer Handbook of Robotics, second ed. Springer, pp. 1577–1604. http://dx.doi.org/10.1007/978-3-319-32552-1_60.

Ono, M., Heverly, M., Rothrock, B., Almeida, E., Calef, F., Soliman, T., Williams, N., Gengl, H., Ishimatsu, T., Nicholas, A., Stilley, E., Otsu, K., Lange, R., Milkovich, S.M., 2018. Mars 2020 site-specific mission performance analysis: Part 2. Surface traversability. In: AIAA SPACE and Astronautics Forum and Exposition. http://dx.doi.org/10.2514/6.2018-5419.

Overbye, T., Saripalli, S., 2021. Path optimization for ground vehicles in off-road terrain. In: IEEE International Conference on Robotics and Automation. pp. 7708–7714. http://dx.doi.org/10.1109/ICRA48506.2021.9561291.

Papadakis, P., 2013. Terrain traversability analysis methods for unmanned ground vehicles: A survey. Eng. Appl. Artif. Intell. 26 (4), 1373–1385. http://dx.doi.org/10.1016/j.engappai.2013.01.006.

Safar, M.J.A., Watanabe, K., Maeyama, S., Nagai, I., 2012. Tip-over stability prediction for a holonomic omnidirectional transport mobile robot. In: The 6th International Conference on Soft Computing and Intelligent Systems, and the 13th International Symposium on Advanced Intelligence Systems. IEEE, pp. 763–768. http://dx.doi.org/10.1109/SCIS-ISIS.2012.6505217.

Sánchez-Ibáñez, J.R., Pérez-del Pulgar, C.J., Azkarate, M., Gerdes, L., García-Cerezo, A., 2019. Dynamic path planning for reconfigurable rovers using a multi-layered grid. Eng. Appl. Artif. Intell. 86, 32–42. http://dx.doi.org/10.1016/j.engappai.2019.08.011.

Sánchez-Ibañez, J.R., Pérez-del Pulgar, C.J., García-Cerezo, A., 2021. Path planning for autonomous mobile robots: A review. Sensors 21 (23), http://dx.doi.org/10.3390/s21237898.

Santos, L., Santos, F., Mendes, J., Costa, P., Lima, J., Reis, R., Shinde, P., 2020. Path planning aware of robot's center of mass for steep slope vineyards. Robotica 38 (4), 684–698. http://dx.doi.org/10.1017/S0263574719000961.

Shum, A., Morris, K., Khajepour, A., 2015. Direction-dependent optimal path planning for autonomous vehicles. Robot. Auton. Syst. 70, 202–214. http://dx.doi.org/10.1016/j.robot.2015.02.003.

Singh, A.K., Krishna, K.M., Saripalli, S., 2016. Planning non-holonomic stable trajectories on uneven terrain through non-linear time scaling. Auton. Robots 40 (8), 1419–1440. http://dx.doi.org/10.1007/s10514-015-9505-5.

Toscano-Moreno, M., Bravo-Arrabal, J., Sánchez-Montero, M., Serón Barba, J., Vázquez-Martín, R., Fernandez-Lozano, J.J., Mandow, A., Garcia-Cerezo, A., 2022. Integrating ROS and android for rescuers in a cloud robotics architecture: Application to a casualty evacuation exercise. In: IEEE International Symposium on Safety, Security, and Rescue Robotics. pp. 1–7. http://dx.doi.org/10.1109/SSRR56537.2022.10018629.

UMA, 2022. LAENTIEC: laboratory and experimentation area in new technologies for emergency intervention. URL: https://www.uma.es/laentiec. (Accessed 3 September 2022).

Vidoni, R., Bietresato, M., Gasparetto, A., Mazzetto, F., 2015. Evaluation and stability comparison of different vehicle configurations for robotic agricultural operations on side-slopes. Biosyst. Eng. 129, 197–211. http://dx.doi.org/10.1016/j.biosystemseng.2014.10.003.

Xiaodong, Y., Tianqing, C., Kaixuan, C., Liyang, Z., Jie, Z., 2022. Off road path planning based on hybrid artificial potential field and ant colony algorithm. In: International Conference on Innovations and Development of Information Technologies and Robotics. pp. 27–31. http://dx.doi.org/10.1109/IDITR54676.2022.9796489.

Yang, B., Wellhausen, L., Miki, T., Liu, M., Hutter, M., 2021. Real-time optimal navigation planning using learned motion costs. In: IEEE International Conference on Robotics and Automation. pp. 11739–11745. http://dx.doi.org/10.1109/ICRA48506.2021.9561861.

Yu, X., Huang, Q., Wang, P., Guo, J., 2020. Comprehensive global path planning for lunar rovers. In: International Conference on Unmanned Systems. pp. 505–510. http://dx.doi.org/10.1109/ICUS50048.2020.9274967.