CLASH: A Comfort-Aware Local Autonomous System for Height-Optimized Path Planning using Semantic Occupancy Grids

Introduction to Intelligent Vehicles 2025 - Final Report

R13922A15 陳星佑　R13922186 高永杰　R13922193 渠景量

Abstract

In autonomous driving, local path planning is essential for navigating short-range, obstacle-rich environments. While traditional methods often prioritize the shortest distance, they frequently overlook terrain-induced ride discomfort caused by abrupt elevation changes. In this paper, we propose CLASH (Comfort-aware Local Autonomous System using Height-based planning), a novel framework that jointly optimizes path length and ride smoothness by leveraging semantic and geometric understanding of the scene. Given a single frame of 6-view surround images, we employ a 3D occupancy prediction network (FlashOCC [1]) to generate a voxelized semantic grid of the environment. From this grid, we directly extract drivable surface height information by identifying the vertical voxel level corresponding to each drivable cell. The resulting semantic voxel map is projected into a Bird's Eye View (BEV) representation, where we integrate both semantic class labels and height data into a cost map. A comfort-aware A* algorithm then plans a local trajectory that avoids obstacles while minimizing both lateral distance and vertical terrain variation.

This work presents a comprehensive combination of algorithmic research and system-level implementation, validating our framework through detailed case studies and visualized experiments under both average and edge-case scenarios.

Contribution

R13922A15 陳星佑

- Project manager
- Project structure and pipeline designing
- Comfort-aware A* algorithm ideation
- BEV, Cost map implementation
- Report writing

R13922186 高永杰

- FlashOCC studying and inferencing
- 3D semantic occupancy visualizing
- Height info. construction
- Comfort-aware A* algorithm implementation

- Conduct experiments
- Report writing

R13922193 渠景量

- Background study
- FlashOCC inferencing
- BEV, Cost map implementation
- Report writing

Double Assignment

We confirm that this report does not involve any double assignment and does not include any work completed before this semester.

Use of AI Tools

We have used artificial-intelligence tools during the preparation of this report. Specifically, we used AI tools to help translate and refine our writing for better clarity and fluency. In addition, we used AI-assisted tools like GitHub Copilot and ChatGPT to help identify and resolve bugs during programming and debugging. All core ideas, designs, and implementations were developed and carried out by the team members.

1 Introduction

Autonomous vehicle navigation has made significant strides in recent years, with robust advancements in perception and path-planning technologies becoming increasingly crucial. Conventional path-planning algorithms predominantly emphasize obstacle avoidance and shortest-path determination, leveraging 2D bird's-eye-view (BEV) representations. However, these approaches often neglect vital factors influencing passenger comfort, such as road surface irregularities and height variations, which directly impact ride smoothness and vehicle stability. As self-driving technology progressively moves toward deployment in complex urban and suburban environments, addressing these factors becomes increasingly critical to ensure practical viability and widespread adoption.

Existing local path-planning methods frequently operate on simplified assumptions, typically utilizing flat occupancy maps and failing to incorporate nuanced terrain information. Specifically, traditional BEV-based techniques overlook the vertical variations of drivable surfaces, resulting in path selection that, while shortest in distance, may traverse uneven terrain, leading to discomfort or instability. Thus, a gap persists between obstacle-aware shortest-path solutions and a comprehensive planning approach that adequately addresses comfort-related criteria.

Motivated by the necessity to bridge this gap, our work seeks to integrate advanced 3D occupancy prediction frameworks, such as FlashOCC, into local navigation systems for

autonomous vehicles. By leveraging the capability of FlashOCC to generate semantic-rich voxel grids from multi-view camera inputs, we aim to construct comprehensive semantic BEV representations. Furthermore, we propose capturing the vertical distribution of drivable surfaces as supplementary information for path-planning tasks. This methodology not only provides a richer environmental understanding but also serves as a crucial experiment evaluating the practical feasibility and efficacy of employing 3D occupancy prediction techniques for future local navigation applications in autonomous vehicles.

Specifically, this research aims to develop a local path-planning pipeline that explicitly considers both ride comfort and route efficiency. First, we use FlashOCC to predict semantic voxel occupancy from a single frame of multi-camera imagery. Next, we compress these 3D voxel data into semantic BEV maps, incorporating height-level information about drivable surfaces. Finally, we create a cost map that integrates semantic, obstacle, and terrain irregularity information, and implement a comfort-aware A* algorithm to plan paths that simultaneously minimize road unevenness and overall travel distance. Through this holistic approach, our objective is to advance local path-planning techniques, moving closer to realizing practical, comfortable, and efficient autonomous driving.

2 Related Work

2.1 3D Occupancy Prediction and BEV Representation

Recent advances in 3D scene understanding have focused on semantic occupancy prediction from monocular or multi-view images. FlashOCC [1] proposes a fast and memory-efficient framework that predicts sparse 3D voxel grids with semantic labels directly from multi-camera inputs, making it suitable for downstream planning tasks in autonomous systems. In contrast, MonoScene [3] performs dense volumetric semantic scene completion from a single RGB image, effectively reconstructing occluded geometry, but at the cost of increased computational demand and reduced scalability.

Beyond voxel-based approaches, several methods directly generate BEV features from 2D images. Lift-Splat-Shoot (LSS) [4] introduced a camera-based pipeline that lifts image features into 3D frustums and splats them onto a bird’s-eye view plane, forming the foundation for many subsequent BEV-based perception models. M²BEV [5] further demonstrates the versatility of BEV representation by jointly performing object detection and HD map segmentation within a unified BEV space. Compared to these view-transformer-based approaches, our method adopts a voxel-first pipeline that preserves complete 3D structural information, which is essential for constructing elevation-aware cost maps in path planning.

2.2 Cost Map Design for Path Planning

Classical path planning methods, such as A* and Dijkstra, operate on 2D occupancy grids where cost maps are handcrafted using distance metrics and obstacle inflation. While effective for structured environments, these approaches often neglect terrain variability and fail to adapt to dynamic or uneven surfaces.

To address this, the work titled *Real-time Optimal Navigation Planning Using Learned Motion Costs* [6] proposed a real-time navigation framework that uses learned motion costs from simulated trajectories, allowing planners to reason about traversability on challenging terrain. Although their focus was on legged ground robots, the underlying principle—learning cost from geometric structure—informs our inclusion of elevation-aware penalties.

We further ground our method in the nuScenes dataset [7], a large-scale autonomous driving benchmark that provides multi-camera imagery, 3D annotations, and drivable surface labels. This enables us to generate BEV-based cost maps enriched with semantic and elevation features for realistic urban navigation.

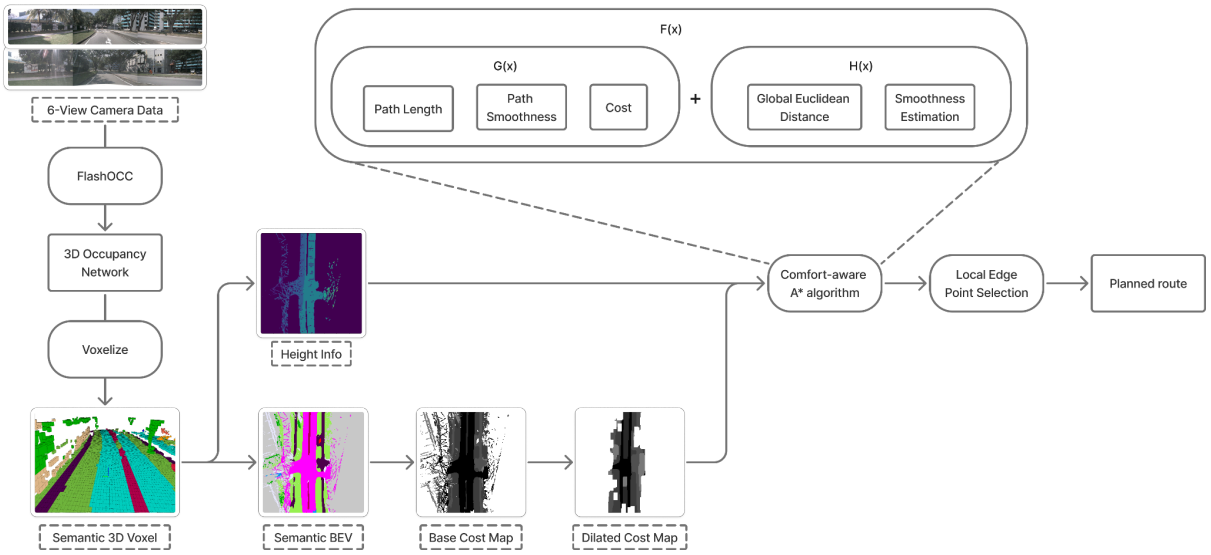
2.3 Ride Comfort in Autonomous Driving

While classical planning approaches prioritize shortest distance and obstacle avoidance, ride comfort is an increasingly important consideration for autonomous passenger vehicles. Factors such as terrain roughness, vertical jerk, and abrupt elevation changes can significantly degrade passenger experience, especially in urban and semi-structured environments.

To address this, prior work has investigated traversability assessment using 3D perception data. Suger et al. [8] proposed a semi-supervised framework that estimates terrain traversability from 3D LiDAR data, enabling mobile robots to avoid rough or unstable surfaces. Although their method was developed for outdoor robot navigation, it highlights the importance of surface-based cues in improving motion quality. Inspired by this, our approach integrates elevation and slope penalties directly into the cost map, allowing the planner to select smoother, more comfortable paths while maintaining collision avoidance and goal-oriented behavior.

3 Methodology

Figure 1: System Overview



3.1 Overview

In this section, we present the overall pipeline of our proposed CLASH (Comfort-aware Local Autonomous System using Height-based planning) framework. The system aims to generate a comfort-optimized local path by jointly considering semantic scene understanding and terrain geometry.

As illustrated in **Figure 1**, our method consists of five main stages. First, we leverage FlashOCC to process multi-view camera images and estimate a dense 3D occupancy field, which is then voxelized into a semantic voxel grid. Second, we project the semantic voxel grid into a bird's-eye-view (BEV) map to obtain a 2D representation of the scene layout. Third, we extract the drivable surface height from the voxel data to model the underlying terrain elevation. Fourth, we generate a base cost map from the BEV map and further apply morphological dilation to form a smooth and conservative dilated cost map. Finally, we apply a comfort-aware A* algorithm that integrates both geometric cost and ride-smoothness penalties to search for the optimal path. Among multiple candidate routes leading to the edge of the local planning region, we select the path with the lowest f(x) score, which balances path length, terrain smoothness, and global directionality.

Each of these components is elaborated in the following subsections.

3.2 FlashOCC

To construct the 3D occupancy map, we adopt FlashOCC as the core perception backbone in our framework. According to the original paper, FlashOCC outperforms existing state-of-the-art methods in accuracy, inference speed, memory efficiency, and ease of

deployment. Given the computational constraints of onboard systems in autonomous vehicles, the lightweight and efficient design of FlashOCC makes it a strong candidate for future real-time deployment.

We selected scene-0039 from the nuScenes dataset as our inference dataset. This scene lasts approximately 20 seconds, consisting of 40 frames captured at intervals of roughly 0.5 seconds per frame. For each frame, six camera images covering surround-view perspectives (front-left, front, front-right, rear-left, rear, and rear-right) serve as input to FlashOCC.

After inference, FlashOCC outputs an .npz file containing a 3D semantic occupancy field. Each voxel in this grid carries a semantic label, fully describing drivable areas, vehicles, pedestrians, static obstacles, and other environment features.

From this semantic occupancy map, we extract two key types of information separately: (1) a semantic projection into a 2D BEV map, as detailed in Section 3.3, to provide a planar semantic layout of the scene; and (2) the voxel height indices associated with drivable surfaces as terrain height information, described in Section 3.4, to estimate road unevenness and consequently inform ride comfort considerations.

3.3 Semantic BEV

With the 3D occupancy map provided by FlashOCC, our next step is to generate a semantic bird's-eye-view (BEV) representation, which is essential for the subsequent construction of the cost map used in path planning.

Since the occupancy data is inherently three-dimensional, it must be compressed into a two-dimensional plane to obtain a meaningful BEV map. Specifically, the original 3D occupancy for each frame is structured as a voxel grid of dimension $200 \times 200 \times 16$, where 16 denotes the vertical resolution along the height axis (z-axis). Consequently, each pixel in the resulting BEV map corresponds to a vertical stack of 16 voxels. To resolve this dimensional reduction while retaining critical semantic information, we establish a predefined semantic priority order shown in Table 1. Each pixel in the BEV is then assigned the semantic category corresponding to the highest-priority voxel present within the vertical stack, thereby converting the 3D occupancy data into a compact and semantically informative 2D BEV representation.

Given the inherent limitations in prediction accuracy of FlashOCC, certain pixels in the semantic BEV may appear as "free," indicating either truly vacant spaces or regions lacking sufficient sensor coverage. To address these semantic gaps, we implement a localized semantic completion method within a 66×66 pixel square surrounding the ego-vehicle. This targeted region, rather than the entire BEV map, is sufficient for our local navigation task and reduces computational overhead.

The semantic filling strategy proceeds as follows:

1. If a pixel is initially labeled as "free," we first assign it the most frequent semantic category among its immediate vertical and horizontal neighbors.
2. In cases where multiple categories are equally represented, we then examine the semantic labels of the diagonal neighboring pixels.
3. If ambiguity remains, the final semantic assignment is determined according to the predefined semantic priority order.

Through this localized completion process, we achieve a fully semantic-rich BEV map around the vehicle, suitable for precise and efficient local path planning.

Table 1: The semantic priority order used in our BEV filling process is defined such that higher numerical values indicate higher priority. The semantic class is defined by nuScenes. This priority determines which class is selected when multiple candidates are present in a voxel stack or when filling ambiguous pixels. The complete priority list is as follows:

Semantic Class	Priority	Semantic Class	Priority
car	100	sidewalk	55
truck	95	terrain	50
bus	90	manmade	40
motorcycle	85	vegetation	35
pedestrian	80	other_flat	30
bicycle	75	barrier	25
traffic_cone	70	trailer	20
construction_vehicle	65	others	0
driveable_surface	60	free	0

3.4 Height Information

Given that our project focuses on comfort-oriented navigation, accurately capturing road unevenness is critical to minimize vehicle vibration and enhance passenger comfort. Leveraging the previously generated 3D occupancy map, we explicitly model road elevation information to identify and avoid surface irregularities such as bumps and potholes.

To encode terrain elevation, we first identify all voxels labeled as *driveable_surface* within the current frame. Among these, the lowest voxel height level is designated as the baseline (set to height zero). The remaining drivable surface voxels are then assigned heights relative to this baseline, indicating their relative elevation. Conversely, all voxels that do not belong to the *driveable_surface* class are assigned a height value of -1. By this approach,

non-drivable voxels are naturally excluded from consideration in our subsequent comfort-aware A* path planning algorithm.

This explicit encoding of terrain height information enables the Comfort-aware A* algorithm to effectively prioritize routes that avoid abrupt elevation changes, thus ensuring smoother and more comfortable vehicle trajectories.

3.5 Cost Map

With the generated semantic BEV map, we proceed to construct the cost map, which serves as a foundational input for our path planning algorithm. Since each pixel in the BEV is assigned a semantic label, we can associate each class with a predefined cost value according to the *id2cost* mapping (see Table 2).

In our configuration, darker pixels represent lower cost and are thus more traversable—for example, *driveable_surface* is assigned the lowest cost and rendered as black. In contrast, dynamic obstacles such as vehicles and pedestrians are assigned the highest cost (white), discouraging the planner from selecting paths that intersect with these regions. Using this mapping, we convert the semantic BEV into a grayscale cost map where pixel intensity directly encodes traversal cost.

To further ensure safe navigation, we apply a morphological dilation operation to the cost map based on the physical dimensions of the vehicle, which we assume to be 15 pixels in length and 8 pixels in width. As long as all obstacles are dilated by at least 15 pixels, we can safely treat the vehicle as a single pixel during path planning without risk of collision.

The dilation is applied uniformly across all obstacle regions. Because higher-cost areas appear brighter in the cost map, their values will dominate over lower-cost (darker) areas during the dilation process. As a result, the obstacle regions effectively expand outward, overwriting nearby drivable surfaces. The final dilated cost map ensures that all obstacles are padded sufficiently and can be used directly by our Comfort-aware A* algorithm for efficient and safe route generation.

Table 2. id2cost: cost assigned to each semantic class *(Lower values indicate more drivable regions)*

Semantic Class	Priority	Semantic Class	Priority
car	255	sidewalk	80
truck	255	terrain	60
bus	255	manmade	120
motorcycle	255	vegetation	50
pedestrian	255	other_flat	40

bicycle	255	barrier	255
traffic_cone	230	trailer	255
construction_vehicle	255	others	200
driveable_surface	1	free	1

3.6 Comfort-Aware A* Algorithm

To enhance navigation in unstructured environments, we propose a Comfort-Aware A* algorithm that integrates terrain smoothness and obstacle risk into the path planning process. This algorithm extends the classical A* approach by incorporating real-time cost and surface height information, allowing for more comfortable and safer trajectory planning in challenging terrain.

Input & Output The input to our algorithm consists of a dilated 2D cost map and a 2D height information. In the cost map, pixel intensity encodes the difficulty of traversal, where higher values indicate greater risk, and values above a threshold ($\geq$240) represent non-drivable obstacles. The height map provides elevation information, with invalid regions marked by -1. The algorithm also receives a local start point, representing the vehicle's current position, and dynamically selects a local goal by searching for the closest traversable point in the direction of the global goal. The output is a local path optimized for safety and ride comfort.

Motivation & Design The classical A* algorithm uses Euclidean distance as the sole cost metric, which is insufficient for driving over complex terrain. Our design modifies the A* cost structure by redefining the cumulative cost G(n) and heuristic cost H(n) to reflect terrain and vehicle dynamics. Specifically, the G(n) cost from the start node to a candidate node n is expressed as:

$$G(n) = \alpha \cdot \text{distance} + \beta \cdot \text{bumpiness} + \gamma \cdot \text{total\_cost}$$

Here, distance denotes the accumulated Euclidean distance from the start, bumpiness is the sum of absolute height differences encountered along the path, and total_cost is the sum of pixel values from the cost map, reflecting obstacle proximity or surface risk. The weights α, β, and γ are hyperparameters that tune the emphasis on each component.

To estimate the cost from a candidate node n to the global goal, we define the heuristic function H(n) as:

$$H(n) = \text{Euclidean}(n, \text{goal}) + w_1 \cdot \text{local\_bumpiness}(n)$$

The first term measures direct Euclidean distance to the global goal, while the second term introduces a penalty based on local terrain irregularity. This local bumpiness is computed as the minimum elevation difference between the node and its immediate neighbors, serving as a proxy for the surface roughness the vehicle may encounter beyond the visible local map.

Search Process The search proceeds by maintaining a priority queue ordered by the total estimated cost f(n) = G(n) + H(n). At each iteration, the node with the lowest f(n) is expanded. For every such node, we evaluate five neighboring positions—down, down-left, down-right, left, and right. Neighboring cells are only considered if they lie within the map bounds, have valid elevation data, and do not exceed the cost threshold. For each viable neighbor, we compute its updated cumulative distance, bumpiness, and cost values, then evaluate the new G(n) and H(n). If the resulting G(n) is lower than any previously recorded value for that node, it is updated and reinserted into the queue.

Summary In summary, the Comfort-Aware A* algorithm improves upon the classical method by embedding considerations of terrain comfort and traversal safety into the cost evaluation. This modification makes the approach particularly well-suited for local planning tasks in autonomous driving systems, enabling real-time generation of paths that balance efficiency, comfort, and risk avoidance.

4 Experiment

4.1 Experiment Overview & Setup

In this section, we present the experimental setup used to evaluate the performance of our comfort-aware A* algorithm in various driving scenarios. Our experiments aim to assess the impact of different weighting combinations on the path planning process, particularly focusing on the trade-off between path length, ride comfort (bumpiness), and cost efficiency. To achieve this, we examine three distinct case studies, including both average and extreme cases, with each case subject to the same set of weight combinations for the G(n) function in our A* algorithm.

For each of the three case studies, we test six different weight combinations for the G(n) function used in our comfort-aware A* algorithm. These weights represent the relative importance of three factors: path length, path smoothness (bumpiness), and overall cost. The weight combinations tested are as follows:

- Setup 1 (1, 0, 0, 1): Emphasizing path length.
- Setup 2 (0, 1, 0, 1): Emphasizing ride comfort (bumpiness).
- Setup 3 (0, 0, 1, 1): Emphasizing cost efficiency.
- Setup 4 (1, 1, 0, 1): Balancing path length and ride comfort.

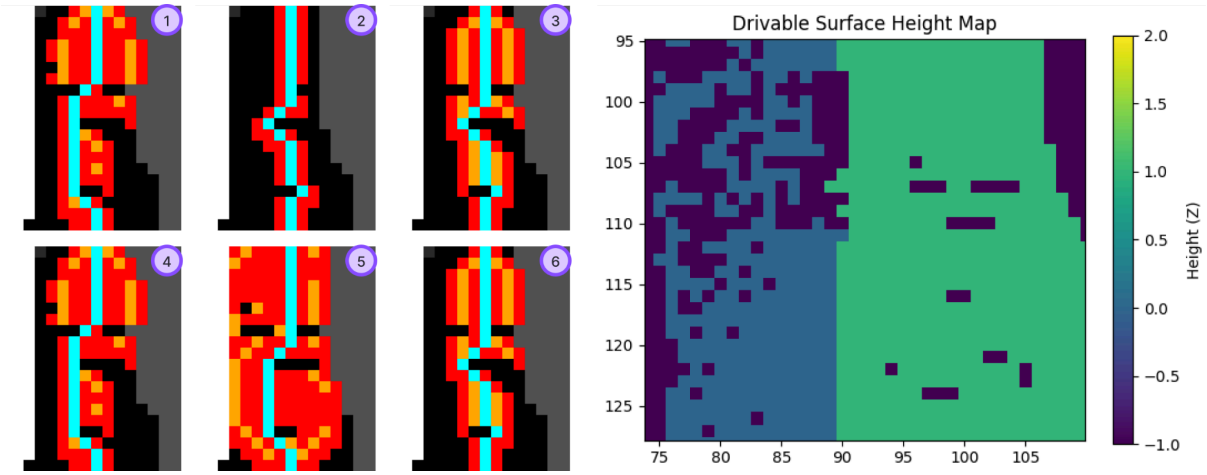
- Setup 5 (1, 0, 1, 1): Balancing path length and cost efficiency.
- Setup 6 (0, 1, 1, 1): Balancing ride comfort and cost efficiency.

These combinations will allow us to observe the effect of different prioritization strategies on the generated paths, especially in terms of smoothness, efficiency, and navigation feasibility.

There are several figures (Figure 2, 3, and 4) that visualize the experimental results presented in the following sections. In each figure, the left part displays six subfigures corresponding to different configurations of the comfort-aware A* cost function. The background of each subfigure represents the cost map, where darker regions indicate lower cost and brighter regions indicate higher cost. Light blue points represent the final selected path, orange points indicate traversed nodes, and red points denote all nodes considered during the search. Each configuration is denoted as (α, β, γ, w), where α, β, and γ correspond to the weights of distance, bumpiness, and total cost in G(n), respectively, and w denotes the weight of local bumpiness in H(n). The right part of each figure shows the drivable surface height map, provided as a visual reference for interpreting terrain variation across the planning area.

4.2 Case Study - Average Case (Figure 2)

Figure 2: The route planning process and drivable surface height map of Average Case



- Setup 1 – (Top left):
  The algorithm strongly favors the shortest route. The path is relatively direct but passes through bumpier areas, showing limited concern for comfort.
- Setup 2 – (Top middle):
  With a focus on bumpiness, the path shifts laterally to avoid rough terrain. Although longer, it results in a smoother ride.
- Setup 3 – (Top right):
  This setup prioritizes cost minimization, leading to a route that balances terrain and

structure but does not explicitly minimize path length or bumpiness. The path tends to follow semantically preferred regions but may not be smooth.

- Setup 4 – (Bottom left):
  Balancing length and comfort, the path avoids sharp turns and rough zones while remaining relatively efficient, showing compromise between travel efficiency and ride quality.

- Setup 5 – (Bottom middle):
  This configuration reduces path length and cost, often resulting in the most efficient route that still respects semantic drivable zones, though at the expense of some comfort.

- Setup 6 – (Bottom right):
  Prioritizing both cost and ride comfort, the path avoids irregular surfaces while making only necessary detours, often producing the smoothest and most semantically aligned route.

The results show how varying the weights influences path selection. Configurations favoring smoothness lead to more comfortable but longer paths, while those prioritizing path length result in shorter, less comfortable routes. These experiments demonstrate CLASH's adaptability to different needs, balancing efficiency and comfort.

4.3 Case Study - Extreme Case 1: Flaws in 3DOcc Network (Figure 3)

Figure 3: The route planning process and drivable surface height map of Extreme Case 1

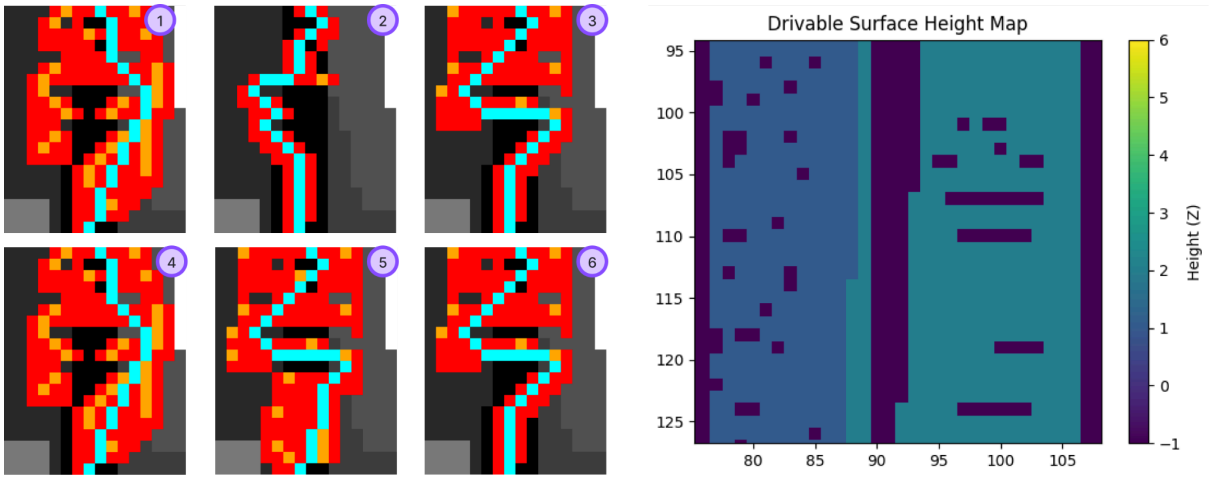


Figure 3 presents an extreme case designed to reveal the influence of imperfect predictions from the 3D occupancy network (FlashOCC) on path planning performance. In this scenario, the global goal lies in a straight direction with minimal semantic obstacles. However, due to flaws or inconsistencies in the predicted 3D voxel representation—such as missing drivable voxels, over-segmented road patches, or misclassified terrain—the planner faces challenges in identifying an optimal path.

- Setup 1 – (Top left):
  The path strongly prefers minimal distance. However, due to gaps or fragmentation in the 3D voxel map, the shortest path is distorted and deviates noticeably from the global direction.

- Setup 2 – (Top middle):
  With a focus on terrain smoothness, the planner avoids bumpy or inconsistent areas. The resulting path is smoother but exhibits large lateral detours caused by local voxel noise.

- Setup 3 – (Top right):
  When minimizing cost, the path becomes sensitive to implicit penalties in flawed regions. This often causes sudden turns or detours, even without true obstacles.

- Setup 4 – (Bottom left):
  Combining length and bumpiness, this configuration partially compensates for voxel artifacts, but still reflects non-ideal detours and minor path jitter.

- Setup 5 – (Bottom middle):
  Focusing on distance and cost, this path attempts to stay efficient but is disrupted by incomplete drivable surface predictions, resulting in suboptimal trajectory quality.

- Setup 6 – (Bottom right):
  Emphasizing comfort and cost, this route avoids rough and uncertain zones, but due to the flawed occupancy map, it takes an over-conservative path, deviating significantly from the intended direction.

This case highlights a critical vulnerability in planning over predicted occupancy maps. Even with no visible obstacles, inaccuracies in 3DOcc outputs—such as fragmented or missing drivable regions—can mislead the planner and result in unnatural detours. The effectiveness of our comfort-aware A* algorithm is closely tied to the quality of semantic voxel inputs, underscoring the importance of robust voxel estimation in upstream perception networks for reliable local planning.

4.4 Case Study – Extreme Case 2: Perfect Prediction Case (Figure 4)

Figure 4: The route planning process and drivable surface height map of Extreme Case 2

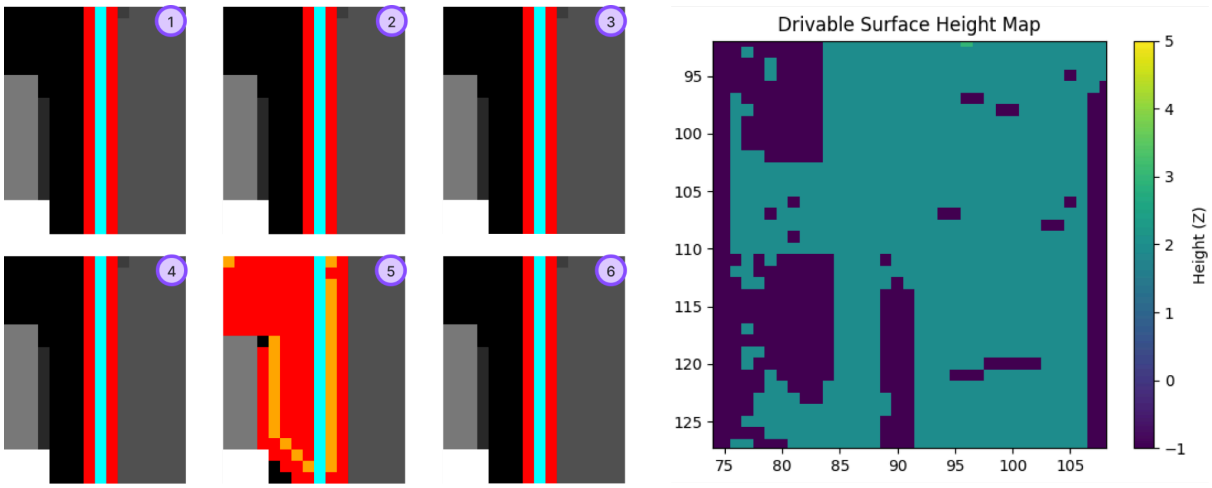


Figure 4 illustrates an ideal scenario in which both the drivable surface and the predicted voxel occupancy map are perfectly clean, continuous, and obstacle-free. The terrain is entirely flat, and the road is straight with no elevation changes, holes, or segmentation errors. In such an environment, the optimal path is trivially a straight line, and the system's ability to maintain consistent behavior across cost configurations can be evaluated.

- Setup 1 to 6 (Top left to Bottom right):
  All configurations yield nearly identical straight-line paths, with minimal variation in explored nodes. This uniformity arises not from the cost function design, but from the inherently perfect environment—no terrain irregularities, cost map inconsistencies, or occupancy prediction noise are present to influence the search. As a result, the planner converges on the same optimal trajectory regardless of weight configuration.

This case highlights that when the terrain is flat, complete, and free of prediction flaws, the planner's behavior becomes stable and configuration-invariant. It demonstrates that our comfort-aware A* algorithm will not introduce unnecessary deviation when the environment is geometrically and semantically ideal. This serves as a sanity check and confirms the expected baseline behavior under perfect conditions.

5 Conclusion

In this work, we introduced a novel local path-planning pipeline for autonomous vehicles that incorporates semantic 3D occupancy predictions, terrain elevation information, and a comfort-aware A* algorithm to simultaneously address both ride comfort and path efficiency. By leveraging FlashOCC, a state-of-the-art semantic 3D occupancy network, we successfully transformed multi-view camera images into semantic-rich bird’s-eye-view (BEV) maps.

These BEV representations were augmented by explicitly capturing drivable surface elevation, enabling more nuanced and comfort-aware path planning decisions.

The main contributions of our work are threefold. First, we successfully integrated FlashOCC-derived semantic occupancy results into the local path-planning framework, demonstrating the practical feasibility of employing advanced 3D occupancy networks for vehicle navigation tasks. Second, we improved passenger comfort by explicitly incorporating road surface height information, thereby providing a smoother and more stable driving experience. Finally, we proposed an enhanced version of the A* algorithm that combines traditional shortest-path metrics with terrain smoothness criteria, enabling routes that balance minimal distance and minimal road irregularity.

Despite these advances, several opportunities remain for future improvement. Currently, our framework operates on single-frame inputs; therefore, introducing temporal consistency across sequential frames could significantly enhance the stability and predictability of generated paths. Furthermore, while terrain bumpiness was inferred from voxel-height data, future work should incorporate empirical measurements using inertial measurement units (IMU) or real-world ride comfort data to validate and refine our approach. Finally, integrating our local planner with a global navigation system could enable comprehensive multi-stage decision-making, thus extending our method's applicability and effectiveness in realistic autonomous driving scenarios.

References

[1] Yu, Z., Shu, C., Deng, J., Lu, K., Liu, Z., Yu, J., ... & Chen, Y. (2023). Flashocc: Fast and memory-efficient occupancy prediction via channel-to-height plugin. *arXiv preprint arXiv:2311.12058*.

[2] https://github.com/Yzichen/FlashOCC

[3] Cao, A. Q., & De Charette, R. (2022). Monoscene: Monocular 3d semantic scene completion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 3991-4001).

[4] Philion, J., & Fidler, S. (2020). Lift, splat, shoot: Encoding images from arbitrary camera rigs by implicitly unprojecting to 3d. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIV 16* (pp. 194-210). Springer International Publishing.

[5] Xie, E., Yu, Z., Zhou, D., Philion, J., Anandkumar, A., Fidler, S., ... & Alvarez, J. M. (2022). M $^ 2$ BEV: multi-camera joint 3D detection and segmentation with unified birds-eye view representation. *arXiv preprint arXiv:2204.05088*.

[6] Yang, B., Wellhausen, L., Miki, T., Liu, M., & Hutter, M. (2021, May). Real-time optimal navigation planning using learned motion costs. In *2021 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 9283-9289). IEEE.

[7] Caesar, H., Bankiti, V., Lang, A. H., Vora, S., Liong, V. E., Xu, Q., ... & Beijbom, O. (2020). nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 11621-11631).

[8] Suger, B., Steder, B., & Burgard, W. (2015, May). Traversability analysis for mobile robots in outdoor environments: A semi-supervised learning approach based on 3D-lidar data. In *2015 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 3941-3946). IEEE.