sensors

MDPI

Article

Enhanced Visual SLAM for Collision-Free Driving with Lightweight Autonomous Cars

Zhihao Lin [1,†], Zhen Tian [1,†], Qi Zhang [2], Hanyang Zhuang [3] and Jianglin Lan [1,*]

[1] James Watt School of Engineering, University of Glasgow, Glasgow G12 8QQ, UK; 2800400l@student.gla.ac.uk (Z.L.); 2620920z@student.gla.ac.uk (Z.T.)
[2] Faculty of Science, University of Amsterdam, Science Park 904, 1098 XH Amsterdam, The Netherlands; q.zhang2@uva.nl
[3] University of Michigan-Shanghai Jiao Tong University Joint Institute, Shanghai Jiao Tong University, Shanghai 200240, China; zhuanghany11@sjtu.edu.cn
* Correspondence: jianglin.lan@glasgow.ac.uk
† These authors contributed equally to this work.

Abstract: The paper presents a vision-based obstacle avoidance strategy for lightweight self-driving cars that can be run on a CPU-only device using a single RGB-D camera. The method consists of two steps: visual perception and path planning. The visual perception part uses ORBSLAM3 enhanced with optical flow to estimate the car's poses and extract rich texture information from the scene. In the path planning phase, the proposed method employs a method combining a control Lyapunov function and control barrier function in the form of a quadratic program (CLF-CBF-QP) together with an obstacle shape reconstruction process (SRP) to plan safe and stable trajectories. To validate the performance and robustness of the proposed method, simulation experiments were conducted with a car in various complex indoor environments using the Gazebo simulation environment. The proposed method can effectively avoid obstacles in the scenes. The proposed algorithm outperforms benchmark algorithms in achieving more stable and shorter trajectories across multiple simulated scenes.

Keywords: autonomous car; obstacle avoidance; SLAM; vision-based navigation

Citation: Lin, Z.; Tian, Z.; Zhang, Q.; Zhuang, H.; Lan, J. Enhanced Visual SLAM for Collision-Free Driving with Lightweight Autonomous Cars. *Sensors* **2024**, *24*, 6258. https://doi.org/10.3390/s24196258

Academic Editor: Yasufumi Enami

Received: 16 August 2024
Revised: 23 September 2024
Accepted: 25 September 2024
Published: 27 September 2024

1. Introduction

In recent years, there has been an increasing demand for autonomous vehicles in complex indoor environments. Compared with drones, autonomous cars can perform a variety of ground transportation tasks while maintaining good stability. To avoid causing damage to the items being transported, it is crucial for unmanned vehicles to remain stable under all circumstances. Therefore, these vehicles should be equipped with obstacle avoidance algorithms to perform effectively in hazardous situations such as the sudden appearance of moving obstacles, tight corners, and narrow corridors. However, the existing algorithms [1–3] face various practical issues. Unmanned vehicles are often limited by heavy sensors (Radar, LiDAR, etc.), which can lead to short battery life, high costs, and large sizes, limiting the use cases and performance of these algorithms. Thus, choosing the most suitable sensors for unmanned vehicles is a critical task. Unmanned vehicles are often limited by heavy sensors (Radar, LiDAR, etc.), which can lead to short battery life, high costs, and large sizes, limiting the use cases and performance of these algorithms. Thus, choosing the most suitable sensors for unmanned vehicles is a critical task.

Common sensors used for these tasks include ultrasonic sensors, LiDAR, and cameras. Ultrasonic sensors excel at measuring short-range obstacles, but their accuracy decreases with distance. LiDAR sensors offer rich information, high precision, long range, and a wide field of view, but are expensive and heavy, reducing unmanned vehicles' flexibility. In contrast, cameras provide extensive scene information with low power consumption,

compact size, and affordability. Therefore, developing high-precision algorithms based on camera sensors is of significant practical importance.

Recent advancements in autonomous navigation have made significant progress, but challenges remain, especially for lightweight vehicles in complex environments. State-of-the-art methods can be broadly categorized into two approaches:

(1) Deep learning-based methods: Works such as [4,5] utilize neural networks for information extraction from cameras. While highly accurate, they require substantial computational resources, including GPUs, limiting their applicability on lightweight platforms.
(2) Traditional feature-based methods: vision-based SLAM using feature extraction can operate on CPU-only platforms but often lack the necessary accuracy for reliable navigation in complex scenarios.

Many works utilize deep learning to extract information from cameras [4,5]. These methods require high-performance computing boards with GPUs to run the algorithms, which are not suitable for lightweight small vehicle platforms. Vision-based Simultaneous Localization and Mapping (SLAM) using feature extraction methods can run on CPU-only platforms, obtaining the map information of the scene to make more precise decisions and estimating the necessary pose information for the vehicle. However, additional information is still needed to enhance the SLAM system's accuracy.

When the environment is entirely constructed by perception technologies in real-time, trajectory planning for the moving vehicle is necessary. Trajectory planning is a challenging module, as there are requirements from both the moving vehicle and the environment. From the vehicle's perspective, the trajectory must be stable to ensure good stability and comfort. i.e., avoiding damping and sharp changes. From the environment's perspective, there is a series of obstacles that increase the risk of collision. Therefore, a method that addresses both stability and safety is needed. Control Lyapunov function (CLF) with stability constraints can be combined with the control optimization process to enhance the stability of the control system. Control barrier function (CBF) is used in RGB-D space to improve driving safety [6]. CLF-CBF-QP is used to ensure the stability and safety of mobile car trajectory planning [7].

Recent works predominantly utilize depth maps and poses estimated via odometry as inputs for corresponding path-planning tasks [5,8–11]. These approaches often rely on IMUs, GPS, or LiDAR for pose calculation. Meanwhile, pose and depth estimation cannot be effectively globally optimized together, which fails to handle the noise introduced by depth estimation and odometry. This significantly impairs the performance of the planners. Some research employs deep learning to reduce uncertainty in depth estimation and uses reinforcement learning to deal with the noise in inputs [5,8], but it places higher demands on the GPUs and CPUs onboard the vehicles. This restricts the application of such works in scenarios with limited computational resources.

To address these limitations and advance the state-of-the-art in autonomous navigation for lightweight vehicles, the proposed method improves enhanced visual SLAM with advanced trajectory planning. The proposed approach offers the following key advancements:

This paper proposes a new autonomous obstacle avoidance algorithm for lightweight self-driving cars. The state-of-the-art visual SLAM algorithm ORBSLAM3 is used to perceive the scene, and its performance is enhanced by eliminating outliers with optical flow epipolar constraints. ORB-SLAM3 is capable of joint optimization of pose and 3D mapping using only the CPU, and it can merge submaps to achieve real-time reconstruction of large-scale scenes whilst storing their 3D maps. Based on the rich pose and 3D information of the scene obtained, a new autonomous obstacle avoidance algorithm for lightweight self-driving cars is proposed. The proposed method employs a trajectory generation method combined with an obstacle shape reconstruction process (SRP) in irregular-shaped environments (CLF-CBF-QP-SRP) for global planning. Global planning trajectories with safety and stability can provide a general reference trajectory for self-driving cars, together with local planning using TEB (Timed Elastic Band) for local planning to ensure avoiding the local collisions. The proposed

method can effectively avoid obstacles in the scene whilst minimizing unnecessary movement. The contributions of this work are summarized as follows:

- Development of a lightweight, single-camera-based visual SLAM system enhanced with optical flow for outlier culling, capable of perceiving rich environmental information and avoiding obstacles efficiently on CPU-only platforms.
- Introduction of a novel path planning algorithm for irregular environments, combining CLF-CBF-QP-SRP for global planning with TEB for local planning, achieving robust collision-free navigation to various target points.
- Enhancement of trajectory generation robustness through a unified approach that ensures dynamic stability and safety throughout the entire movement, from the initial point to the target point.
- Comprehensive evaluation and comparison with state-of-the-art methods, demonstrating superior performance in generating safe, stable, and efficient trajectories for lightweight autonomous vehicles in complex indoor environments.

Through extensive simulations and comparisons with existing methods, exhaustive experiments demonstrate that the proposed approach outperforms current state-of-the-art techniques in terms of computational efficiency, trajectory stability, and obstacle avoidance capability for lightweight autonomous vehicles.

2. Related Works

2.1. Geometric Methods Enhanced SLAM Approach

Visual SLAM has been improved through a series of geometric techniques. Sun et al. [12] used additional optical flow information and a foreground model based on depth maps to eliminate outliers in the scene. Cheng et al. [13] employed the fundamental matrix to enhance the additional information provided by the LK sparse optical flow to further increase the algorithm's ability to outliers.

Recent work [14] has utilized the relationships between points to filter stable data associations in the scene, and ref. [15] used multi-frame rather than single-frame historical observations to further identify outliers in the scene. However, these techniques do not address outliers in certain specific scenes and are usually limited to certain types of cameras. In addition, some works [16–18] have introduced a significant amount of navigation-irrelevant information and often involve lengthy processing times.

ORB-SLAM3 represents a significant advancement in visual SLAM systems, offering robust performance with minimal computational requirements. Unlike deep learning-based approaches, ORB-SLAM3 employs fast ORB feature extraction and binary descriptor matching, enabling real-time performance even on CPU-only platforms. The system also supports multi-map and multi-session capabilities, which allow for efficient map reuse and reliable loop closure. Additionally, ORB-SLAM3 incorporates visual-inertial odometry to enhance robustness in challenging environments, while efficient bundle adjustment and pose graph optimization ensure accurate trajectory estimation. These characteristics make ORB-SLAM3 particularly suitable for lightweight autonomous vehicles, where computational resources are limited but real-time performance remains essential.

2.2. Trajectory Generation Methods

The process of path planning is important for moving robot cars to reach the target point safely and with good stability. A series of methods are proposed for the trajectory generation [19], such as rapid random tree (RRT) and the Voronoi diagram-based method. RRT can be used to solve motion planning problems for robots to move from one state to another whilst avoiding obstacles by generating a space-filling tree to effectively find an optimal path. However, RRT has a major limitation that the solution may not be optimal, as the convergence rate is uncertain. Therefore, some adjusted versions of RRT are proposed, such as [20]. A Voronoi diagram-based method for trajectory planning is proposed in [21], with simplicity, versatility, and efficiency. Ref. [22] uses the Voronoi diagram-based method combined with a roadmap to find the shortest path. Ref. [23] uses the Voronoi diagram to

exclude collisions in the free space. Ref. [24] uses the Voronoi diagram to generate a safe path among the road map. However, the performance of the Voronoi diagram heavily relies on cell distribution. In areas with sparse cells, the effectiveness of the Voronoi diagram diminishes.

Other techniques, like the use of artificial potential field (APF) in [25], for trajectory planning with multiple obstacles have also been implemented. By using the attractive and repulsive force fields on the target point and obstacles, APF guides the vehicle to the target point. A framework that combines the APF with reinforcement learning is proposed in [26], achieving collision avoidance with dense obstacles. However, the main problem of APF is the unstable trajectory, as the state of the car is affected by the joint force of the attractive and repulsive force fields. In contrast to other route planning algorithms, CBF is effective for collision avoidance and enhancing safety, while CLF contributes to the stability of nonlinear systems [27]. Thus, their combination promises both collision-free and stable navigation. Moreover, to address a variety of scenarios, local planning is essential to correct any discrepancies introduced by global planning. Hence, TEB [28], recognized for its proficiency in local route planning, is well-suited for this role.

3. System Overview

As shown in Figure 1, the proposed system uses visual sensors to map the environment in advance to obtain spatial boundary and obstacle information. During the process of path planning and navigation, the proposed method utilizes images captured by the car's camera to generate feature points through descriptor matching. The epipolar constraints of the LK optical flow [29] are used to filter and remove outliers from the scene. Finally, the proposed method uses the relocation algorithm to obtain the current location and attitude information of the car. When the car is moving, the proposed method uses the ORB-SLAM3 algorithm to update the car's pose information in real time. After the navigation coordinate points are completely set, the system will use the pre-known static cost map for inflation. For global path planning, the proposed method uses CBF to update the relative distance between obstacles and the car in real time to achieve autonomous obstacle avoidance. For complex terrain and new obstacles, the autonomous car uses the Timed Elastic Band (TEB) [28] local planning algorithm to plan the car's path locally and use the constraints between the car and surrounding obstacles to speed up iterations to find the optimal path.

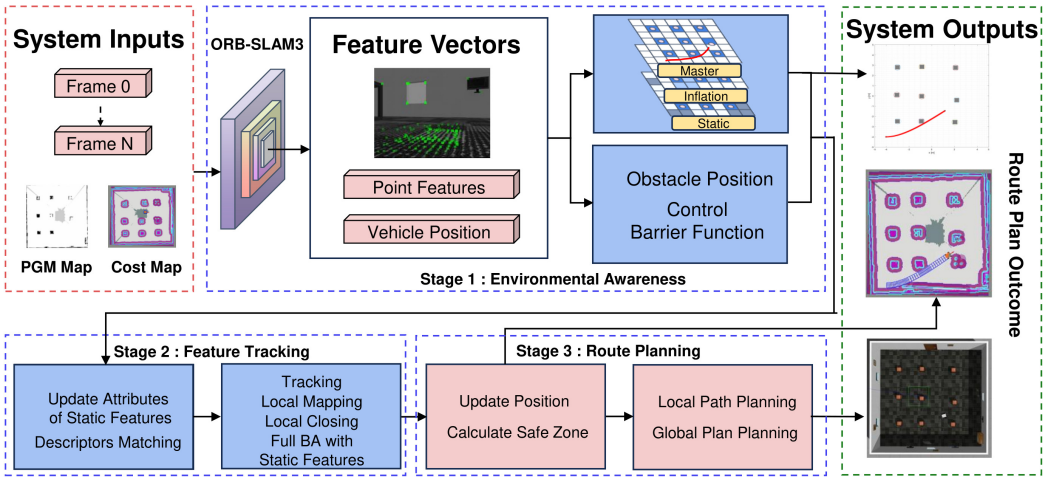


Figure 1. The proposed system workflow. This system comprises two main components: environment perception and path planning. Initially, a PGM map and cost map are constructed. The vehicle, equipped

with a visual sensor, extracts point features from the environment and uses relocation to ascertain its position and identify obstacles. A static map is inflated for navigational safety. The vehicle's pose is dynamically updated by tracking map points, and a global path is mapped using CBF. For local path planning, the TEB algorithm is employed. The system updates the vehicle's pose in real-time, calculates safe passage areas with CBF, and facilitates optimal, obstacle-free path selection to the destination.

3.1. Perception Based on Vision

Compared with LiDAR, visual sensors have the advantages of low cost and small size and are widely used in various autonomous driving platforms. This article uses a visual SLAM system based on a RGB-D camera as shown in Figure 2, which ensures positioning accuracy and saves costs, facilitating subsequent deployment on low-cost unmanned vehicles and realizing the project as soon as possible.

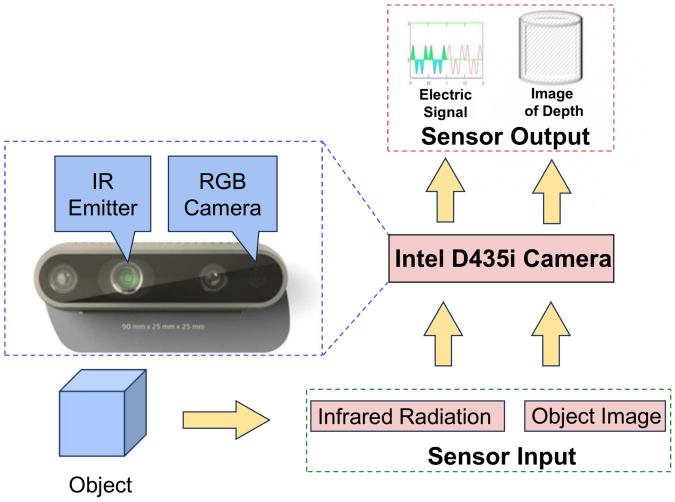


Figure 2. The Intel D435i RGB-D camera utilizes the structured light triangulation method for depth sensing.

Point Features Matching and Attributes Updating

A descriptor is a compact representation of a feature point's local appearance, typically a binary or floating-point vector. In this context, desc(.) represents the function that computes this descriptor for a given point. The proposed method employs a comprehensive point feature-matching methodology for stereo camera configurations. The initial step involves identifying point features from the last frame and current from the RGB-D camera that share the same "grid ID" of these points. Each point feature is assigned a unique "grid ID" based on its spatial locality, facilitating the association of features between frames. To ensure temporal consistency, the algorithm matches features by comparing descriptors and selecting the closest match based on the minimal Euclidean distance:

$$\text{Distance} = \min_{j} \|\text{desc}_{\text{prev}}(i) - \text{desc}_{\text{curr}}(j)\| \tag{1}$$

where $\text{desc}_{\text{prev}}(i)$ and $\text{desc}_{\text{curr}}(j)$ are descriptors of points from previous and current frames, respectively. Further validation is performed by examining the cosine similarity of the angle between the direction vectors of matched points, ensuring the matched points align accurately with the expected motion model.

$$\text{Cosine Similarity} = \cos^{-1}\left(\frac{\vec{v}_{\text{prev}} \cdot \vec{v}_{\text{curr}}}{\|\vec{v}_{\text{prev}}\|\|\vec{v}_{\text{curr}}\|}\right) \tag{2}$$

where $\vec{v}_{\text{prev}}$ and $\vec{v}_{\text{curr}}$ are the direction vectors of points in consecutive frames. The point feature matching process among consecutive frames is visualized in Figure 3. In the figure, $T_{k-1,k} \in SE(3)$ represents the relative pose transformation between frames or the last and current image frames. Point features in 3D space, defined by a point B_j and another point F_j, result in two points: the point a_i and another point f_i when projected onto the image coordinate system I_{k-1} at time $t-1$. At the time t, the same point projected onto the image coordinate system I_k results in new points: point a_i' and another point f_i'.

Bidirectional cosine similarity is used to match point pairs across frames based on grid IDs, discarding pairs below a similarity threshold. For RGB-D camera frames, feature points' grayscale centroids and direction vectors will be computed, which are crucial for pose optimization during bundle adjustment.

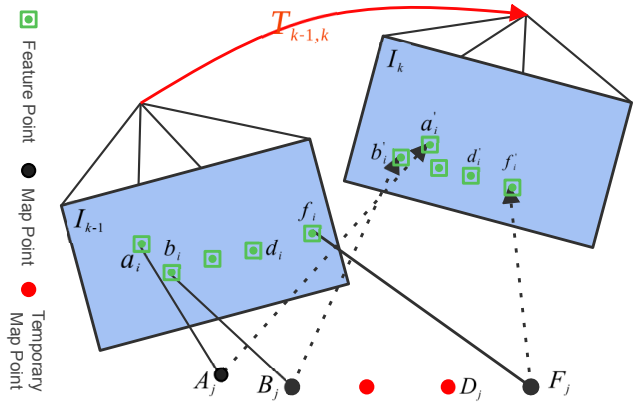


Figure 3. Illustration of the point feature matching process. Point features and their associated descriptors (compact representations of local appearance) are matched between consecutive frames using grid IDs, Euclidean distance, and cosine similarity to ensure alignment and temporal consistency. Solid lines represent the connections between the map points projected onto the 2D plane in the previous frame and the feature points, while dashed lines represent the connections between the map points projected onto the 2D plane in the current frame and the feature points.

3.2. Outlier Features Removing

In the proposed SLAM system, the proposed method mitigates the impact of inconsistent feature points on pose estimation by implementing the Lucas–Kanade (LK) method to enforce epipolar constraints, guided by the fundamental matrix F. This ensures geometrically coherent feature point pairs by setting a threshold distance from the epipolar line, allowing us to filter out mismatches. This fusion of techniques bolsters system stability.

To refine feature selection, the proposed algorithm eschews traditional Harris corner [30] matches in favor of those with lower disparity in central pixel blocks. The proposed method then applies a stringent distance metric to eliminate outliers, which is central to the precision of the proposed algorithm.

For rigor, the RANSAC algorithm [31] is employed to extract the fundamental matrix F that maximizes inlier correspondences. With this matrix, features are extracted from one frame to their epipolar counterparts in the subsequent frame.

Consider matched points p_1 and p_2 in consecutive frames with homogeneous coordinates $\mathbf{P}_1 = [u_1, v_1, 1]^\top$ and $\mathbf{P}_2 = [u_2, v_2, 1]^\top$, respectively. The epipolar line $\mathbf{L}_1$ for $\mathbf{P}_1$ is obtained as $\mathbf{L}_1 = F\mathbf{P}_1$. The distance to the epipolar line is defined as

$$D = \frac{|\mathbf{P}_2^\top F \mathbf{P}_1|}{\sqrt{F_{row1}^2 + F_{row2}^2}} \tag{3}$$

where F, F_{row1}, and F_{row2} represent the fundamental matrix, the first and second rows of the fundamental matrix F, respectively. Points with D exceeding a predefined threshold are deemed outliers and discarded.

3.3. Point Features Optimizing Algorithm

In 3D visual SLAM systems, map point features play a critical role in environment modeling and localization. However, factors like sensor noise and dynamic environments can induce errors in the orientation and position of these features. To address this, the proposed method adopts an optimization-based reference keyframe pose correction method to enhance the accuracy and consistency of map point features.

Sim3 (Similarity Transform in 3D) is a seven-dimensional transformation that includes rotation, translation, and scale. It is used here to represent the relationship between different coordinate frames. The process of pose correction is detailed as follows: Let R_{wr} and t_{wr} represent the rotation matrix and the translation vector from the reference keyframe to the world coordinate system, respectively. The world coordinates of the map point features to be corrected are assumed to be P_{3Dw}, with $P_{3Dw_{sp}}$ denoting the coordinate of the point feature. The Sim3 transformation matrices S_{rw} and $corSwr$ are used to transform the pose of map point features from the reference keyframe of the current frame to the corrected reference keyframe as follows:

$$CorP_{3D_{wsp}} = corSwr \times (S_{rw} \times P_{3D_{wsp}}) \tag{4}$$

where $\times$ denotes matrix multiplication, and $CorP_{3Dw_{sp}}$ represents the coordinates of the corrected map point feature in the corrected reference keyframe coordinate system. Enhancing the precision of these transformations directly impacts the accuracy of the camera pose and the overall SLAM performance. By refining the keyframe poses through these corrections, the proposed method effectively reduces the impact of positioning errors caused by sensor noise and dynamic environmental factors. This sets a robust foundation for subsequent optimization processes such as motion-only bundle adjustment (BA).

Motion-only BA optimizes the camera orientation $R \in SO(3)$ and position $t \in \mathbb{R}^3$, by minimizing the reprojection error between matched 3D points X_i' in world coordinates and their corresponding image keypoints x_i', which may be either monocular $x_m^i \in \mathbb{R}^2$ or stereo $x_s^i \in \mathbb{R}^3$, with $i \in \Lambda$ the set of all matches, as follows:

$$\{R, t\} = \arg\min_{R,t} \sum_{i \in \Lambda} \rho \left\| x_i' - \pi_d \left(R X_i' + t \right) \right\|_{\Sigma}^2 \tag{5}$$

where ρ is the robust Huber cost function, and Σ represents the covariance matrix related to the scale of the keypoint. The optimization problem in Equation (5) is typically solved using iterative nonlinear least squares methods, such as the Levenberg–Marquardt algorithm. This method is particularly effective for bundle adjustment problems due to its ability to handle the sparsity of the Jacobian matrix efficiently. And π_d is the projection function for RGB-D cameras defined as

$$\pi_d \left(\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \right) = \begin{bmatrix} f_x \frac{X}{Z} + c_x \\ f_y \frac{Y}{Z} + c_y \\ Z \end{bmatrix} \tag{6}$$

where $[X, Y, Z]^\top$ represent the coordinates of a point in the world coordinate system. $\frac{X}{Z}$ and $\frac{Y}{Z}$ are the normalized image plane coordinates. f_x and f_y are the focal lengths of the camera along the X and Y axes, respectively. c_x and c_y are the coordinates of the principal point, typically at the center of the image. Z is the depth value directly measured by the RGB-D sensor, providing real-time depth at each image pixel.

Local BA optimizes a subset of covisible keyframes K_L and all points observed in those frames P_L. Non-optimized keyframes K_F, while fixed during optimization, contribute observations of P_L, adding constraints to enhance map stability without altering their

poses. Define λ_k as the set of matches between points in P_L and keypoints in keyframe k. The optimization is formulated as follows:

$$\{\mathbf{X}_i, R_l, t_l \mid i \in P_L, l \in K_L\} = \arg\min_{\mathbf{X}_i, R_l, t_l} \sum_{k \in K_L \cup K_F} \sum_{j \in \lambda_k} \rho(E_{kj}) \tag{7}$$

where $\mathbf{X}_i$ represents the 3D position of the i-th point in P_L. R_l and t_l are the rotation matrix and translation vector for the l-th keyframe in K_L. Reprojection error E_{kj} is quantified as

$$E_{kj} = \left\| \mathbf{x}'_j - \pi_d(R_k \mathbf{X}_i + t_k) \right\|^2_{\Sigma} \tag{8}$$

where $\mathbf{x}'_j$ represents the 2D projection coordinates of the j-th feature point on the image plane. The optimization problem involving Equation (8) is part of the larger Local BA process, which is typically solved using sparse bundle adjustment techniques. These methods, often implemented using libraries such as g2o or Ceres Solver, exploit the problem's sparsity structure to efficiently optimize over multiple keyframe poses and 3D point positions simultaneously. In contrast to Local BA, Full BA adjusts all keyframes and map points, except the origin keyframe, which remains fixed to resolve scale ambiguity. This extensive optimization ensures the highest accuracy by refining camera poses and landmark positions across the entire map based on all available visual information.

3.4. Global Path Planning by Using CLF-CBF-QP-SRP

In this section, two key concepts are introduced for designing a safe and stable control system: control Lyapunov function (CLF) and control barrier function (CBF). A CLF is a positive definite function that decreases along the trajectories of the system and can be used to ensure asymptotic stability of a desired equilibrium point. A CBF is a function that satisfies some conditions on its Lie derivatives and can be used to enforce state constraints in the operating space. By combining CLF and CBF, a control law can be designed that guarantees both safety and stability of the car.

3.4.1. Vehicle Model

To simplify the computation during the trajectory generation, a kinematic model [26] is used. The kinematic model is governed by

$$\begin{aligned} \dot{x} &= v\cos(\theta) \\ \dot{y} &= v\sin(\theta) \\ \dot{\theta} &= \omega \end{aligned} \tag{9}$$

where v denotes the velocity of the vehicle, x and y denote the longitudinal and lateral coordinates of the vehicle's midpoint, ω is the angular velocity of the vehicle, and θ is the course angle of the vehicle.

3.4.2. The Formulation of CLF

To generate a stable trajectory for a moving car, the vehicle model is written as the following nonlinear control system:

$$\dot{s} = f(s) + g(s)u \tag{10}$$

where $s \in \mathbb{R}^n$ is the state of the moving car, and $u \in \mathbb{R}^m$ is the control input. Functions f and g are smooth vector fields. The control input is subject to the following constraints:

$$u \in \mathcal{U} \subset \mathbb{R}^m := \{u | u_{\min} \leq u \leq u_{\max}\} \tag{11}$$

where $\mathcal{U}$ is the admissible input set of the control system. $u_{\min}$ and $u_{\max}$ are the minimum value and maximum value of inputs. Definition 1 combines an affine constraint with u to achieve an optimization-based controller.

Definition 1. *Assume V is a Lyapunov function when the following condition is satisfied [27]:*

$$\inf_{u \in \mathcal{U}} \left[L_f V(s) + L_g V(s) u \right] \leq K(V(s)) \tag{12}$$

where $L_f V(s)$ and $L_g V(s)$ are the Lie-derivatives of $V(x)$ and K() is a class $\mathcal{K}$ function. The class $\mathcal{K}$ function is a function $k : (0, p] \to (0, \infty]$ with the property of strictly increasing and the initial value $k(0) = 0$. Then s can be stabilized by the following equation:

$$K_{CLF}(s) := \left\{ u \in \mathcal{U}, L_f V(s) + L_g V(s) u \leq K(V(s)) \right\}. \tag{13}$$

3.4.3. The Stability Control of Moving Car Using CLF

As the basic formulation CLF is introduced in (13), the problem is to combine CLF with the dynamics of the moving car. Assume the current position of the moving car is $p_c = (x_c, y_c, \theta_c)$ and the target position is $p_t = (x_t, y_t, \theta_t)$. The error between the current position and the final position, $e = [x_c - x_t, y_c - y_t, \theta_c - \theta_t]$, can be used for building the CLF as follows:

$$V(s) = ePe^T \tag{14}$$

where P is a 3×3 symmetric matrix with five parameters used to ensure that the $V(s)$ is positive definite. The format of P in this paper is formulated as

$$P = \begin{bmatrix} p_1 & 0 & p_2 \\ 0 & p_3 & p_4 \\ p_2 & p_4 & p_5 \end{bmatrix}. \tag{15}$$

Therefore, (10) can be transferred to the format suitable for the target of the moving car.

3.4.4. The Formulation of CBF

CBF is used for the collision-free control, together with CLF for stability. A set $\mathcal{D}$ is defined as composing a continuously differentiable function h: $\mathcal{Z} \subset \mathbb{R}^n \to \mathbb{R}$, yielding

$$\begin{aligned} \mathcal{D} &= \{ \mathbf{s} \in \mathcal{Z} \subset \mathbb{R}^n : h(\mathbf{s}) \geq 0 \}, \\ \partial\mathcal{D} &= \{ \mathbf{s} \in \mathcal{Z} \subset \mathbb{R}^n : h(\mathbf{s}) = 0 \}, \\ \mathrm{Int}(\mathcal{D}) &= \{ \mathbf{s} \in \mathcal{Z} \subset \mathbb{R}^n : h(\mathbf{s}) > 0 \}, \end{aligned} \tag{16}$$

where $\mathcal{D}$ is the set that achieves collision-free.

Definition 2. *The nonlinear control system is safe if $\mathcal{D}$ is forward invariant. Assume T is the time interval. The set $\mathcal{D}$ is forward invariant when each $\mathbf{s}_0 \in \mathcal{D}, \mathbf{x}(nT) \in \mathcal{D}$ for $\mathbf{s}(0) = \mathbf{x}_0, \forall n \geq 0$ [32].*

Definition 3. *The function h is a CBF defined on the set $\mathcal{Z}$ if there exists an extended class $\mathcal{K}_\infty$ function α such that the nonlinear control system satisfies [27]:*

$$\sup_{u \in \mathcal{U}} [L_f h(s) + L_g h(s) u] \geq -\alpha(h(x)) \tag{17}$$

where $L_f h(s)$ and $L_g h(s)$ are Lie-derivatives of $h(s)$.

The set of controls that allow $\mathcal{D}$ to be collision-free for all $s \in \mathcal{Z}$ is formulated as

$$K_{\mathrm{cbf}}(s) := \{ u \in \mathcal{U}, L_f h(s) + L_g h(s) u \geq -\alpha(h(s)) \}. \tag{18}$$

3.4.5. Safe Control of Moving Car Using CBF

In this paper, all obstacles are considered to be static with square shapes, in contrast to the regular circular shapes used for CBF calculations. While the initial approach simplified obstacles as squares, a more nuanced method is employed to accurately represent and avoid obstacles of various shapes. The obstacle shape reconstruction process begins with point cloud generation, where the RGB-D camera provides a set $P = \{p_1, p_2, \ldots, p_n\}$ of points representing obstacle surfaces in the environment. These points are then clustered to identify distinct obstacles, with each cluster C_i representing a potential obstacle. For each cluster C_i, a convex hull H_i is computed to approximate the obstacle's shape, defined by a set of vertices $V_i = \{v_1, v_2, \ldots, v_m\}$. Based on these convex hulls, barrier functions are constructed for each obstacle. Specifically, for each convex hull H_i, a barrier function $h_i(s)$ is formulated as

$$h_i(s) = \min_{j=1\ldots m} \left((s - v_j)^T n_j - d_{\text{safe}} \right), \tag{19}$$

where s is the vehicle state, v_j are the vertices of the convex hull, n_j are the outward-facing normal vectors of the hull faces, and d_{safe} is a safety distance. This formulation ensures that $h_i(s) > 0$ when the vehicle is outside the obstacle (plus safety distance) and $h_i(s) \leq 0$ when it's inside or on the boundary. The overall barrier function for all obstacles is then defined as

$$h(s) = \min_i h_i(s). \tag{20}$$

This approach allows for more accurate representation and avoidance of obstacles with complex shapes. To compute CBFs, the SRP (Specific Required Parameter) of each obstacle must be determined. As illustrated in Figure 4, a circle is used to enclose the ith obstacle, whose radius is calculated as

$$r_i = \sqrt{\left(\frac{l_i}{2}\right)^2 + \left(\frac{w_i}{2}\right)^2} \tag{21}$$

where l_i and w_i are the length and width of the ith obstacle. To reach the target point in an open space, acquired by the perception stage, there is a moving car together with a set of $\mathbb{N}$ obstacles. Each obstacle is represented by $\mathrm{O}_i \in \mathbb{O} = \{\mathrm{O}_0, \mathrm{O}_1, \ldots, \mathrm{O}_{N-1}\}$. Assume the position of the obstacle O_i is denoted by $\mathbf{z}_{\mathrm{O}_i} = (x_{\mathrm{O}_i}, y_{\mathrm{O}_i})$, thus (17) is converted to

$$\sup_{u \in \mathcal{U}} \left[L_f h_i(s) + L_g h_i(s) u + \frac{\partial h_i(s)}{\partial t} \right] \geq -\alpha(h_i(s)). \tag{22}$$

Thus, for $\mathbf{u} \in \mathcal{U}$, the set of controls that ensures the robot car to be safe is expressed as

$$K^i_{\text{cbf}}(s) := \left\{ L_f h_i(s) + L_g h_i(s) u \geq -\alpha(h_i(s)) \right\}. \tag{23}$$

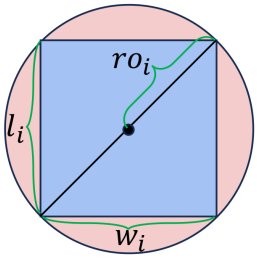


Figure 4. Illustration of SRP for an obstacle.

Assuming that the robot car is defined with a safe radius of r_s, the safe distance between the car and obstacle O_i is defined as $r_i = r_s + r_{O_i}$. Then the CBF is designed as

$$h_i(s) = (x_c - x_{O_i})^2 + (y_c - y_{O_i})^2 - r_i^2. \tag{24}$$

3.5. Safe and Stable Control for Self-Driving Cars

Since both the constraints of accurate and safe control have the affine form, real-time solutions can be acquired. Therefore, a QP-based controller that combines CBFs for safety and CLF with (13) and (23) for stability is as follows:

CLF-CBF-QP:

$$\min_{(u,\delta)\in\mathbb{R}^{m+1}} \frac{1}{2}u^T H u + p\delta^2 + (u-u_l)^T Q(u-u_l) \tag{25a}$$

$$\textbf{s.t.} \quad L_f V(s) + L_g V(s)u + K(V(s)) \leq \delta \tag{25b}$$

$$L_f h_i(s) + L_g h_i(s)u + \alpha(h_i(s)) \geq 0,\ i = 0, 1, \ldots, N-1 \tag{25c}$$

$$u \in \mathcal{U} \tag{25d}$$

where the objective function (25a) is divided into three parts: the first part minimizes the magnitude of **u**, the second part adds an extra quadratic cost, and the third part ensures the smoothness of u. H and Q are positive definite matrices, $p > 0$ is the weight coefficient of the relaxation variable δ and u_l is the control value of the last moment.

4. Experimental Evaluation

The simulations were conducted to verify the safety, stability, and efficiency of the proposed route planning algorithm. The experiments were conducted on a Linux machine with the Ubuntu 18.04.6 LTS OS, a 12th generation 16-thread Intel®Core™ i5-12600KF CPU, an NVIDIA GeForce RTX 3070Ti GPU, and 16 GB of RAM. The QP problem is solved using the quadprog solver in MATLAB R2022B.

In order to verify the effectiveness of the proposed system, a closed square experimental scene is built. In order to ensure that the visual algorithm can extract a certain number of feature points in the surrounding environment, TVs, murals, and other items are added to the inside of the room walls. Among them, 9 square hollow tables are used as obstacles. The feature points in the middle of such hollow objects are often not on the obstacles but on the wall behind them. Such a scenario poses certain challenges to the perception algorithm to test. The car is placed on the map as the carrier of the system, rather than the car's preset starting point. This can effectively test the effectiveness of the proposed system's relocation system. The car in the experiment is rectangular in shape and has four wheels, equipped with an RGB-D vision sensor on the top of the front end. The initial verification of the obstacle avoidance algorithm (CLF-CBF-QP) will be verified in Matlab, and the experiments of the whole visual navigation system are carried out in simulation environments Gazebo and Rviz. Figure 5 illustrates one experiment on how to navigate the car from the start point to the destination.

4.1. Simulation Environments Setup

The car is placed at the starting point with coordinates $(-4, -4)$ and the target points with coordinates (0, 1.5). First, the car uses the visual sensor to detect the feature points in the image, as shown in Figure 5a. After comparing it with the pre-saved atlas, it uses the relocation algorithm to obtain the current location of the car and updates the car's posture and obstacle distance. Then, the velocity and direction of the car are updated by the route planning algorithm. In order to verify the effectiveness of the proposed algorithm, CBF was first performed on MATLAB for the motion trajectory planning test. The simulation trajectory on MATLAB is shown in the red trajectory in Figure 5b. The proposed system ensures that it does not encounter obstacles and selects the shortest distance between two points. In order to further verify the effectiveness of the proposed algorithm, the ROS

system is used to conduct further simulation tests under Ubuntu. The planned path simulation of the car is displayed as a blue box in Rviz, where the point in the middle of the blue box represents the trajectory of the vehicle. The blue box represents the space the car takes up in the environment. The colored map represents costmap, which is used to represent obstacles and passable areas in the environment. The costmap is represented as a two-dimensional grid, with each grid cell (or pixel) colored according to the "cost" or "safety" it represents. The meaning of the colors is as follows: Blue usually represents low-cost areas, i.e., areas that are relatively safe and free for the robot car. Red usually indicates very high costs and is often directly associated with obstacles. This is an area that robot cars should avoid. Grey represents unknown areas, i.e., those areas that the robot car has not yet explored or whose properties cannot be determined. White represents known free areas, i.e., areas without obstructions.

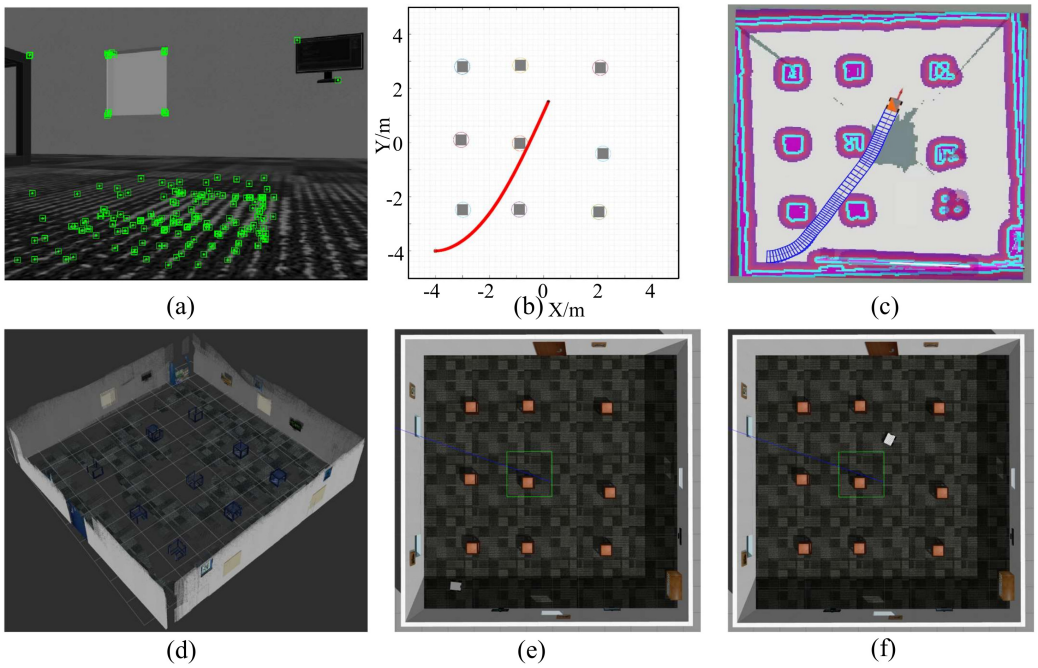


Figure 5. Illustrative experiment showing the robot car's navigation from the start to the destination. (**a**) Detected feature points by the car's vision sensor in a simulated environment, (**b**) ideal trajectory planned using the proposed method in MATLAB, (**c**) actual path followed by the car in Rviz, (**d**) 3D point cloud map of the environment generated by the proposed method, (**e**) starting position of the car in Gazebo, and (**f**) final position of the car in Gazebo.

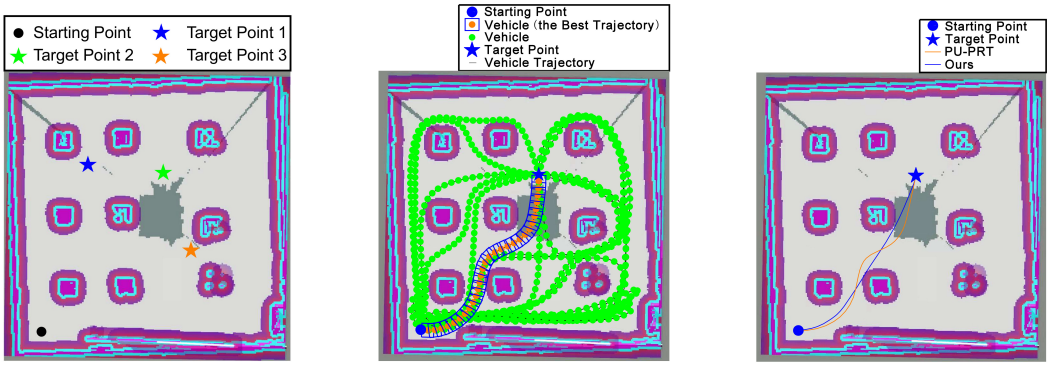
4.2. Stability and Safety of Trajectory Generation

The stability and safety of trajectory generation for the moving car using CLF-CBF-QP-SRP are validated in this section through different testing target points. As shown in Figure 6, three different testing scenarios are illustrated. Target points 1, 2, and 3 are located at the lower right, upper right, and upper left of the center point, respectively. These three target points are defined to test the capability of achieving stability and safety in trajectory generation with various obstacles involved. These target points also test the adaptivity of the proposed CLF-CBF-QP-SRP program. In each case, different states of the moving car will be discussed.

4.2.1. Comparison with State-of-the-Art Algorithm

To demonstrate the efficient trajectory generated by the CLF-CBF-QP-SRP, target point 2 is set as a test point to compare with a state-of-the-art algorithm called PU-RRT [33]. The PU-RRT is capable of generating safe and risk-bounding trajectories and outperforms other adapted RRT algorithms. Therefore, this paper selects PU-RRT for comparison to

verify the relatively more efficient trajectory using the CLF-CBF-QP-SRP while maintaining safety. As illustrated in Figure 6b, a smooth and efficient trajectory without any collisions among a series of candidate trajectories is chosen by the PU-RRT. However, as illustrated in Figure 6c, the trajectory generated by the proposed CLF-CBF-QP-SRP is shorter than that of PU-RRT. This is because the proposed method aims to find the most efficient trajectory while avoiding collisions. Therefore, the proposed method generates a more efficient trajectory than the PU-RRT, while both can efficiently avoid collisions with objects.



(**a**) (**b**) (**c**)

Figure 6. The three different destinations chosen in the experiment, and comparison of the proposed method and PU-PRT algorithm performances for target point 2. (**a**) Experimental settings. (**b**) PU-PRT for target point 2. (**c**) Comparison of the proposed method and PU-PRT.

4.2.2. Safe and Efficient Trajectory Generation for Three Target Points

Figure 7 shows the trajectory generation using the LBF-CBF-QP-SRP program from the starting point to three target points, respectively. Figure 7a illustrates the real environment for trajectory generation. Figure 7e illustrates the detected environment by visual perception. Figure 7b illustrates the LBF-CBF-QP-SRP-based global trajectory for target point 1. Figure 7c illustrates the LBF-CBF-QP-SRP-based global trajectory for target point 2 in MATLAB. Figure 7d illustrates the LBF-CBF-QP-SRP-based global trajectory for target point 3 in MATLAB. Figure 7f illustrates the final trajectory with local safe planning for target point 1 in Rviz. Figure 7g illustrates the final trajectory with local safe planning for target point 2 in Rviz. Figure 7h illustrates the final trajectory with local safe planning for target point 3 in Rviz. To provide a comprehensive comparison, Figure 7b–d represent the optimized estimated poses generated by the proposed algorithm, while Figure 7f–h depict the true poses of the car in the simulated environment. This juxtaposition allows for a direct comparison between the estimated and actual trajectories, demonstrating the accuracy and effectiveness of the proposed approach. From these trajectories, it can be observed that the moving car can avoid collisions and drive close to the boundary of reshaped obstacles, improving the efficiency of reaching the target point. Furthermore, all the trajectories are smooth and stable, as there is no sharp changing. The close correspondence between the estimated and true poses further validates the reliability of the proposed LBF-CBF-QP-SRP program in real-world scenarios. Therefore, the proposed LBF-CBF-QP-SRP program can ensure the safety, smoothness, and stability of driving throughout the whole process.

4.2.3. Safe and Stable Control of Target Point 3

This section illustrates the states, control variables, and total CBF of all obstacles, from the starting point to the target point, among the time axis. As illustrated in Figure 8a, the total CBF is always larger than 0 throughout the whole time, which suggests it is safe driving. Figure 8b shows the angular velocity of the moving car. It is obvious that the changing of the angular velocity is smooth and with a small scale, which suggests stability during the driving. Figure 8c elaborates on the longitudinal position x, lateral position y, and course angle θ of a moving car. The longitudinal distance has changed with

smoothness gradually from −4 to the point near 0, suggesting the smoothness and stability of longitudinal driving. The lateral distance has gradually increased with smoothness from −4 to the point near 2, suggesting the smoothness and stability of lateral driving. The variation of θ is also smooth and reaches a stable level at the end of time, revealing the stability of turning among the whole process.

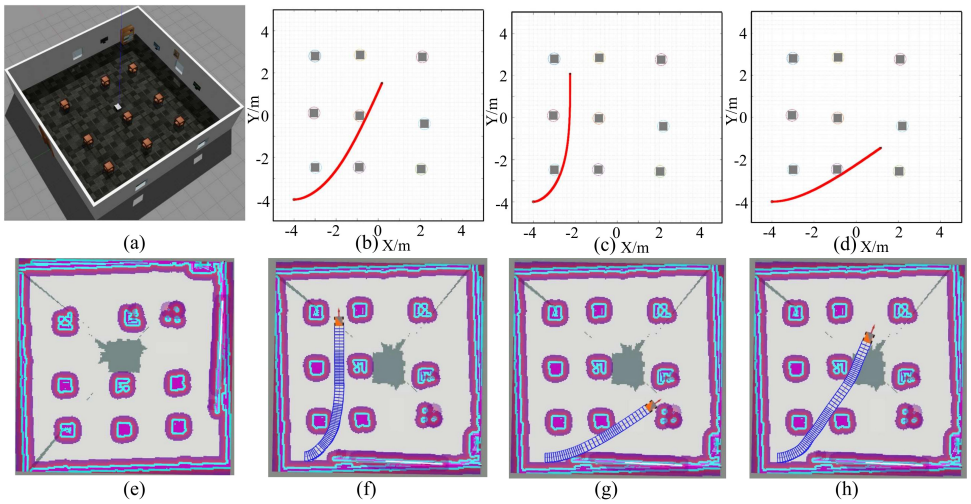


Figure 7. Comparison of the estimated and actual trajectories from the start point to three target points. (**a**) Experimental environment in Gazebo, (**b**) ideal trajectory from the start to target 1 planned in MATLAB, (**c**) ideal trajectory from the start to target 2 planned in MATLAB, (**d**) ideal trajectory from the start to target 3 planned in MATLAB, (**e**) 2D grid map constructed in Rviz, (**f**) actual trajectory from the start to target 1 in Rviz, (**g**) actual trajectory from the start to target 2 in Rviz, (**h**) actual trajectory from the start to target 3 in Rviz.

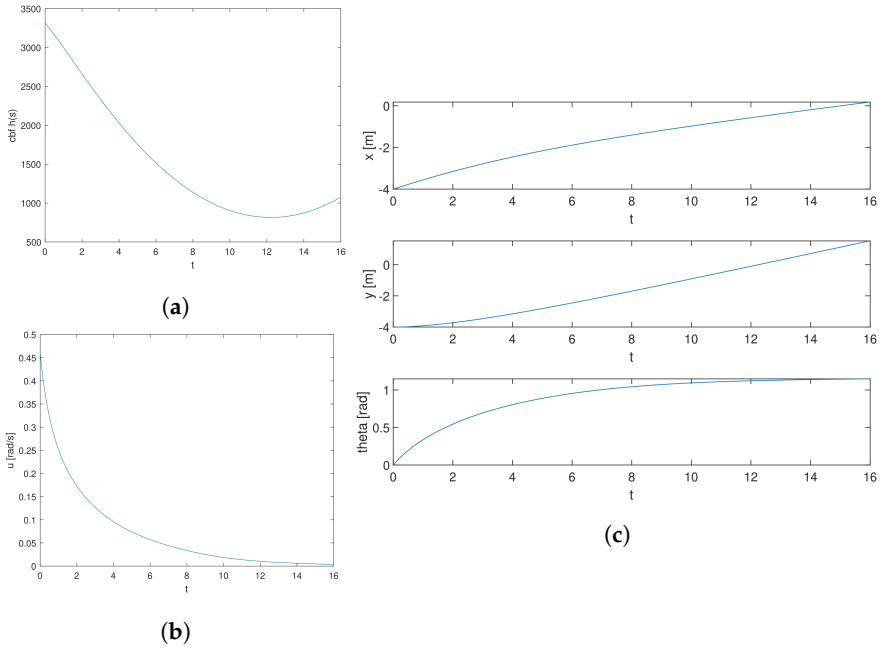


Figure 8. Variation of variables for target point 3. (**a**) Variation of CBF. (**b**) Variation of ω. (**c**) Variation of Longitudinal Distance, Lateral Distance, and θ.

4.2.4. Comparison with Advanced Trajectory Generation Algorithm

To evaluate the performance of the LBF-CBF-QP-SRP program and enhance the validity of the simulation results, it is compared with two benchmark trajectory generation methods: Voronoi diagram [34] and APF [35]. As discussed in Section 2.2, these methods have

the advantages of safety and efficiency. Therefore, they are used to be basepoints for comparison. Figure 9 shows the trajectories generated by the three methods for three different target points, starting from the same initial point $(-4, -4)$. For target point 1, the LBF-CBF-QP-SRP program produces a smooth and collision-free trajectory. However, the APF method exhibits severe oscillation when it approaches the obstacle in the middle of the third row, which indicates a lack of stability. The Voronoi diagram method collides with the leftmost obstacle in the second row. Moreover, its trajectory is neither smooth nor efficient. For target point 2, the LBF-CBF-QP-SRP program generates a smooth and efficient trajectory, as it drives close to the boundary of the map-centered obstacle. However, the APF method has two noticeable oscillations, with the leftmost obstacle in the third row and the map-centered obstacle, respectively, resulting in unstable driving. The Voronoi diagram method follows a longer trajectory than the other two methods, which implies low efficiency. For target point 3, the LBF-CBF-QP-SRP program ensures a smooth, efficient, and good trajectory. The APF method encounters oscillation with the leftmost obstacle in the second row. The Voronoi diagram method makes frequent turns in some parts of the driving, which reduces the efficiency. For the computational time, each case is tested ten times, and the average time is calculated. The average computational times of LBF-CBF-QP-SRP, APF, and the Voronoi diagram are 2.0176 s, 2.2723 s, and 0.0732 s, respectively. Therefore, the computational time for planning the global trajectory of the proposed method is relatively low.

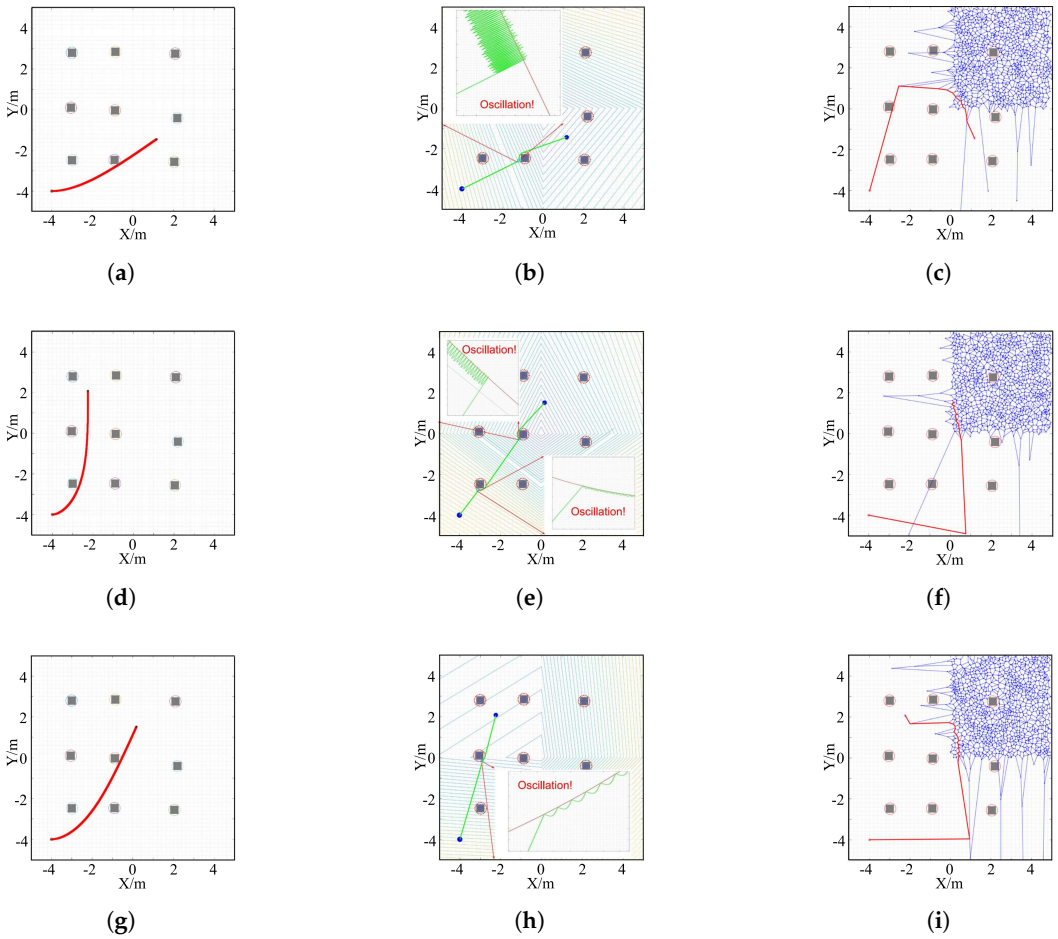


Figure 9. Comparison of CLF-CBF-QP-SRP, APF, and Voronoi diagram for three target points. (**a**) CLF-CBF-QP-SRP for target point 1. (**b**) APF for target point 1. (**c**) Voronoi diagram for target point 1. (**d**) CLF-CBF-QP-SRP for target point 2. (**e**) APF for target point 2. (**f**) Voronoi diagram for target point 2. (**g**) CLF-CBF-QP-SRP for target point 3. (**h**) APF for target point 3. (**i**) Voronoi diagram for target point 3.

4.2.5. Comparison with Deep Learning-Based Methods

To further validate the efficiency and effectiveness of the proposed enhanced visual SLAM system, a comparison with state-of-the-art deep learning-based methods was conducted. Specifically, the proposed approach is compared with two recent deep learning-based methods for simultaneous localization, mapping, and navigation, DS-SLAM [36] and Blitz-SLAM [37], under the same experimental settings. Table 1 presents a quantitative comparison of computational resource consumption and performance metrics:

Table 1. Comparison of Resource Consumption and Performance

Metric	Proposed Method	DS-SLAM [36]	Blitz-SLAM [37]
CPU Usage (%)	45.3	78.6	82.1
GPU Usage (%)	N/A	87.2	91.5
Memory Usage (GB)	2.9	4.8	5.3
Processing Time (ms/frame)	48	62	21
Localization Error (m)	0.082	0.076	0.079

N/A indicates that GPU was not used in the proposed method.

As evident from Table 1, the proposed method significantly outperforms the deep learning-based approaches in terms of computational resource consumption. The proposed method operates without GPU acceleration, utilizing only 45.3% of CPU resources compared with the 78.6% and 82.1% CPU usage of DS-SLAM and Blitz-SLAM, respectively. Memory usage is also substantially lower, requiring only 2.9 GB compared with 4.8 GB and 5.3 GB for the deep learning methods. In terms of processing time, the proposed method achieves 48 ms per frame, more than twice as fast as the deep learning approaches. This efficiency is crucial for real-time applications in lightweight autonomous vehicles.

5. Conclusions

An enhanced visual SLAM-based collision-free driving framework for lightweight autonomous vehicles is proposed in this paper. The proposed method improves the advanced ORB-SLAM3 algorithm, augmented with optical flow techniques to efficiently cull outliers, thereby significantly enhancing the perception capabilities of a single RGB-D camera in complex indoor environments. The novel path planning algorithm integrates control Lyapunov function (CLF) and control barrier function (CBF) within a quadratic programming (QP) framework, which is further refined through an obstacle shape reconstruction process (SRP). The simulation experiments conducted in the Gazebo environment demonstrated that the proposed method effectively generates safe, stable, and efficient trajectories, outperforming existing approaches in computational efficiency and trajectory optimization. The adoption of a camera-based system not only reduces reliance on heavier, more expensive sensor setups but also offers a cost-effective solution with broad applicational potential in autonomous driving technologies. Future efforts will focus on enhancing the adaptability of this system to dynamic environments and integrating advanced machine learning techniques to improve decision-making processes in varying scenarios.

Author Contributions: Conceptualization, Z.L. and Z.T.; methodology, Z.L.; software, Z.T.; validation, Z.L., Z.T. and Q.Z.; formal analysis, Z.L.; investigation, Z.T.; resources, Z.L.; data curation, Z.T.; writing—original draft preparation, Z.L.; writing—review and editing, J.L. and H.Z.; visualization, Z.T.; supervision, J.L.; project administration, Z.L.; funding acquisition, J.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the China Scholarship Council Ph.D. Scholarship for 2023-2027 (No.202206170011), in part by the Leverhulme Trust Early Career Fellowship (ECF-2021-517), and in part by the UK Royal Society International Exchanges Cost Share Programme (IEC\NSFC\223228.)

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Li, Z.; Li, J.; Wang, W. Path Planning and Obstacle Avoidance Control for Autonomous Multi-Axis Distributed Vehicle Based on Dynamic Constraints. *IEEE Trans. Veh. Technol.* **2023**, *72*, 4342–4356. [CrossRef]
2. Kim, C.; Yoon, Y.; Kim, S.; Yoo, M.J.; Yi, K. Trajectory Planning and Control of Autonomous Vehicles for Static Vehicle Avoidance in Dynamic Traffic Environments. *IEEE Access* **2023**, *11*, 5772–5788. [CrossRef]
3. Wang, Y.; Lin, J.; Zhang, L.; Wang, T.; Xu, H.; Qi, Y.; Zhang, G.; Liu, Y. Stable Obstacle Avoidance Strategy for Crawler-Type Intelligent Transportation Vehicle in Non-Structural Environment Based on Attention-Learning. *IEEE Trans. Intell. Transp. Syst.* **2023**, *24*, 7813–7830. [CrossRef]
4. Park, B.; Oh, H. Vision-based obstacle avoidance for UAVs via imitation learning with sequential neural networks. *Int. J. Aeronaut. Space Sci.* **2020**, *21*, 768–779. [CrossRef]
5. Kim, M.; Kim, J.; Jung, M.; Oh, H. Towards monocular vision-based autonomous flight through deep reinforcement learning. *Expert Syst. Appl.* **2022**, *198*, 116742. [CrossRef]
6. Abdi, H.; Raja, G.; Ghabcheloo, R. Safe Control using Vision-based Control Barrier Function (V-CBF). In Proceedings of the 2023 IEEE International Conference on Robotics and Automation (ICRA), London, UK, 29 May–2 June 2023; IEEE: Piscataway, NJ, USA, 2023; pp. 782–788.
7. Desai, M.; Ghaffari, A. Clf-cbf based quadratic programs for safe motion control of nonholonomic mobile robots in presence of moving obstacles. In Proceedings of the 2022 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM), Sapporo, Japan, 11–15 July 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 16–21.
8. Loquercio, A.; Kaufmann, E.; Ranftl, R.; Müller, M.; Koltun, V.; Scaramuzza, D. Learning high-speed flight in the wild. *Sci. Robot.* **2021**, *6*, eabg5810. [CrossRef]
9. Yang, F.; Cao, C.; Zhu, H.; Oh, J.; Zhang, J. FAR Planner: Fast, Attemptable Route Planner using Dynamic Visibility Update. In Proceedings of the 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Kyoto, Japan, 23–27 October 2022; pp. 9–16. [CrossRef]
10. Lin, Z.; Zhang, Q.; Tian, Z.; Yu, P.; Lan, J. DPL-SLAM: Enhancing Dynamic Point-Line SLAM Through Dense Semantic Methods. *IEEE Sens. J.* **2024**, *24*, 14596–14607. [CrossRef]
11. Zhou, X.; Wang, Z.; Ye, H.; Xu, C.; Gao, F. EGO-Planner: An ESDF-Free Gradient-Based Local Planner for Quadrotors. *IEEE Robot. Autom. Lett.* **2021**, *6*, 478–485. [CrossRef]
12. Sun, Y.; Liu, M.; Meng, M.Q.H. Motion removal for reliable RGB-D SLAM in dynamic environments. *Robot. Auton. Syst.* **2018**, *108*, 115–128. [CrossRef]
13. Cheng, J.; Sun, Y.; Meng, M.Q.H. Improving monocular visual SLAM in dynamic environments: An optical-flow-based approach. *Adv. Robot.* **2019**, *33*, 576–589. [CrossRef]
14. Dai, W.; Zhang, Y.; Li, P.; Fang, Z.; Scherer, S. RGB-D SLAM in Dynamic Environments Using Point Correlations. *IEEE Trans. Pattern Anal. Mach. Intell.* **2022**, *44*, 373–389. [CrossRef] [PubMed]
15. Du, Z.J.; Huang, S.S.; Mu, T.J.; Zhao, Q.; Martin, R.R.; Xu, K. Accurate Dynamic SLAM Using CRF-Based Long-Term Consistency. *IEEE Trans. Vis. Comput. Graph.* **2022**, *28*, 1745–1757. [CrossRef] [PubMed]
16. Wang, Y.; Xu, K.; Tian, Y.; Ding, X. DRG-SLAM: A Semantic RGB-D SLAM using Geometric Features for Indoor Dynamic Scene. In Proceedings of the 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Kyoto, Japan, 23–27 October 2022; pp. 1352–1359. [CrossRef]
17. Yuan, C.; Xu, Y.; Zhou, Q. PLDS-SLAM: Point and Line Features SLAM in Dynamic Environment. *Remote Sens.* **2023**, *15*, 1893. [CrossRef]
18. Zhang, Q.; Li, C. Semantic SLAM for mobile robots in dynamic environments based on visual camera sensors. *Meas. Sci. Technol.* **2023**, *34*, 085202. [CrossRef]
19. Mir, I.; Gul, F.; Mir, S.; Khan, M.A.; Saeed, N.; Abualigah, L.; Abuhaija, B.; Gandomi, A.H. A survey of trajectory planning techniques for autonomous systems. *Electronics* **2022**, *11*, 2801. [CrossRef]
20. Wang, K.; Huang, Q.; Wu, S. Application of long short-term memory neural network in geoelectric field data processing. *Chin. J. Geophys.* **2020**, *63*, 3015–3024. (In Chinese) [CrossRef]
21. Bhattacharya, P.; Gavrilova, M.L. Voronoi diagram in optimal path planning. In Proceedings of the 4th International Symposium on Voronoi Diagrams in Science and Engineering (ISVD 2007), Pontypridd, Wales, 9–11 July 2007; pp. 38–47. [CrossRef]
22. Bhattacharya, P.; Gavrilova, M.L. Roadmap-based path planning-using the voronoi diagram for a clearance-based shortest path. *IEEE Robot. Autom. Mag.* **2008**, *15*, 58–66. [CrossRef]
23. Chi, W.; Ding, Z.; Wang, J.; Chen, G.; Sun, L. A Generalized Voronoi Diagram-Based Efficient Heuristic Path Planning Method for RRTs in Mobile Robots. *IEEE Trans. Ind. Electron.* **2022**, *69*, 4926–4937. [CrossRef]
24. Ayawli, B.B.K.; Appiah, A.Y.; Nti, I.K.; Kyeremeh, F.; Ayawli, E.I. Path planning for mobile robots using Morphological Dilation Voronoi Diagram Roadmap algorithm. *Sci. Afr.* **2021**, *12*, e00745. [CrossRef]

25. Triharminto, H.H.; Wahyunggoro, O.; Adji, T.; Cahyadi, A.; Ardiyanto, I. A novel of repulsive function on artificial potential field for robot path planning. *Int. J. Electr. Comput. Eng.* **2016**, *6*, 3262.
26. Wang, M.; Zhang, L.; Zhang, Z.; Wang, Z. A Hybrid Trajectory Planning Strategy for Intelligent Vehicles in On-Road Dynamic Scenarios. *IEEE Trans. Veh. Technol.* **2023**, *72*, 2832–2847. [CrossRef]
27. Ames, A.D.; Coogan, S.; Egerstedt, M.; Notomista, G.; Sreenath, K.; Tabuada, P. Control barrier functions: Theory and applications. In Proceedings of the 2019 18th European control conference (ECC), Naples, Italy, 25–28 June 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 3420–3431.
28. Rösmann, C.; Hoffmann, F.; Bertram, T. Integrated online trajectory planning and optimization in distinctive topologies. *Robot. Auton. Syst.* **2017**, *88*, 142–153. [CrossRef]
29. Lucas, B.D.; Kanade, T. An Iterative Image Registration Technique with an Application to Stereo Vision. In Proceedings of the Imaging Understanding Workshop, Washington, DC, USA, 23 April 1981; pp. 121–130.
30. Harris, C.; Stephens, M. A Combined Corner and Edge Detector. In Proceedings of the Alvey Vision Conference, Manchester, UK, 31 August–2 September 1988; Volume 15.
31. Fischler, M.A.; Bolles, R.C. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Comm. ACM* **1981**, *24*, 381–395. [CrossRef]
32. Huang, J.; Liu, Z.; Zeng, J.; Chi, X.; Su, H. Obstacle avoidance for unicycle-modelled mobile robots with time-varying control barrier functions. In Proceedings of the IECON 2023-49th Annual Conference of the IEEE Industrial Electronics Society, Singapore, 16–19 October 2023; IEEE: Piscataway, NJ, USA, 2023; pp. 1–6.
33. Li, D.; Liu, B.; Huang, Z.; Hao, Q.; Zhao, D.; Tian, B. Safe motion planning for autonomous vehicles by quantifying uncertainties of deep learning-enabled environment perception. *IEEE Trans. Intell. Veh.* **2024**, *9*, 2318–2332. [CrossRef]
34. Jogeshwar, B.K.; Lochan, K. Algorithms for Path Planning on Mobile Robots. *IFAC-PapersOnLine* **2022**, *55*, 94–100. [CrossRef]
35. Mohamed, A.A.; Ziedan, N.I.; Gaafar, T.S. Artificial Potential Field Approaches for Indoor Mobile Robot Path Planning: A Review. *Egypt. Int. J. Eng. Sci. Technol.* **2023**, *44*, 89–98. [CrossRef]
36. Yu, C.; Liu, Z.; Liu, X.J.; Xie, F.; Yang, Y.; Wei, Q.; Fei, Q. DS-SLAM: A Semantic Visual SLAM towards Dynamic Environments. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; pp. 1168–1174. [CrossRef]
37. Fan, Y.; Zhang, Q.; Tang, Y.; Liu, S.; Han, H. Blitz-SLAM: A semantic SLAM in dynamic environments. *Pattern Recognit.* **2022**, *121*, 108225. [CrossRef]