UNIVERSITY OF TECHNOLOGY SYDNEY

Real-time 3D Traffic Cones Detection and Pose Estimation using Monocular Camera for Autonomous Motorsport Racing

Graduate Research Report

The Cong Phan

thecong.phan@student.uts.edu.au

STUDENT ID: 25078231

Supervisor: ***Prof. Shoudong Huang***

shoudong.huang@uts.edu.au

Table of Contents

ABSTRACT

This project introduces a real-time 3D traffic cone detection and pose estimation system for autonomous motorsport racing, based on a monocular visual SLAM (Simultaneous Localization and Mapping) framework. The system integrates odometry, perception, and mapping using only a single camera, eliminating the need for additional sensors and reducing complexity. A layered approach combines the Extended Kalman Filter (EKF) for accurate mapping with a perception pipeline that detects and estimates 3D positions of traffic cones, allowing for precise, real-time navigation in high-speed environments.

To validate the effectiveness and reliability of the developed algorithm, extensive benchmarking was conducted using real-world race data collected by the AMZ Racing Team from ETH Zurich. The results demonstrate that the system achieves high-accuracy 3D detection and pose estimation of traffic cones, which are essential for precise localization and mapping necessary for autonomous race car performance. Additionally, the monocular SLAM system exhibits computational efficiency, ensuring real-time processing capabilities that meet the demanding requirements of motorsport racing.

This project not only advances the field of autonomous navigation in high-speed racing applications but also offers a scalable and adaptable framework applicable to other autonomous vehicle scenarios requiring swift and accurate environmental perception. The successful implementation and validation of this monocular visual SLAM system highlight its potential to enhance the safety, reliability, and competitiveness of autonomous race cars, paving the way for future innovations in autonomous motorsport technologies.

1. INTRODUCTION

Autonomous driving is rapidly emerging as a transformative technology, poised to revolutionize the transportation industry by enhancing safety, efficiency, and accessibility. Combining advancements in computer vision, robotics, and machine learning, autonomous vehicles aim to navigate complex environments with minimal human intervention. Despite significant progress, achieving reliable and adaptable autonomous systems remains a formidable challenge, particularly in dynamic and unpredictable scenarios such as navigating through roadside construction or responding to sudden obstacles. Addressing these challenges is essential for the widespread adoption and success of autonomous vehicles.

1.1. Background and Problems of Research

The pursuit of fully autonomous vehicles involves overcoming numerous technical hurdles, particularly in perception, localization, and decision-making under uncertainty. Traditional approaches often rely on multiple sensors, such as LiDAR, radar, and cameras, to gather comprehensive environmental data. However, this multi-sensor setup can be expensive, complex, and computationally demanding, posing limitations for applications that require lightweight and cost-effective solutions. Additionally, ensuring accurate real-time mapping and localization in diverse and ever-changing environments remains a critical area of research. Developing robust algorithms that can efficiently process sensor data and adapt to varying conditions is paramount for advancing autonomous driving technologies.

1.2. UTS Motorsport and Formular Student

The University of Technology Sydney (UTS) Motorsport team has established itself as a leader in engineering competitions, particularly within the Formula Student framework. Formula Student provides an international platform for multidisciplinary student teams to design, build, and race self-developed racecars, fostering innovation and practical engineering skills. While UTS Motorsport has demonstrated excellence in the Formula Electric category, the team is now expanding its expertise by entering the Driverless category. This transition marks a significant step towards integrating advanced autonomous technologies into their competitive endeavors, aligning with global trends towards intelligent and automated transportation systems.

Figure 1. UTSME22: Ramona, UTSME's Seventh electric vehicle, built from the ground up to participate in the Formula SAE Australasia competition [UTSME, 2022].

1.3.Formular Student Driverless

Amid rising interest in autonomous driving and its technical challenges, Formula Student introduced the Driverless competition in Australia in 2019, with other countries soon following. Formula Student (FS) is a prominent international engineering competition in which multidisciplinary student teams design, build, and race self-developed racecars annually. The addition of the Driverless category marks a significant step forward for the competition, as it requires teams to incorporate advanced autonomous systems into high-performance race vehicles.

The primary event within the Driverless category, known as *Track Drive*, requires teams to complete ten laps around an unfamiliar track as swiftly as possible. These tracks are meticulously defined by small cones measuring 228 × 335 mm, with blue cones indicating the left boundary and yellow cones marking the right boundary. Each track extends up to 500 meters in length and maintains a minimum width of 3 meters, with cones in the same boundary spaced up to 5 meters apart. This precise setup creates a challenging environment that tests the

limits of autonomous navigation and vehicle control. An exemplary track layout schematic is provided in Figure 2.

The lap structure is intentionally designed to evaluate a wide array of driving capabilities essential for autonomous systems. Tracks typically feature a combination of the following elements:

- **Long Straights:** These sections test the vehicle's ability to accelerate and maintain high speeds, challenging the efficiency of the autonomous acceleration and throttle control algorithms.
- **Tight Hairpins:** Sharp, 180-degree turns that assess the precision of the vehicle's steering and braking systems, as well as the effectiveness of real-time path planning and obstacle avoidance strategies.
- **Chicanes:** Series of quick, alternating left and right turns that require rapid directional changes, evaluating the system's agility and responsiveness under dynamic conditions.
- **Multiple Turns with Decreasing Radii:** These sections introduce progressively tighter curves, testing the vehicle's ability to adapt its speed and trajectory in real-time to navigate complex maneuvers without losing control.

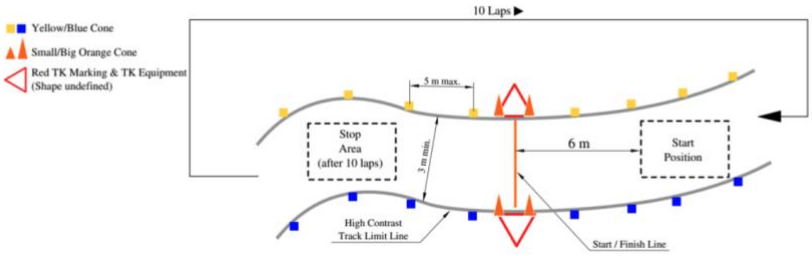

Figure 2. The Track Drive course (FSG, 2018) uses blue and yellow cones to mark the track edges.

An exemplary track layout schematic is provided in Figure 2, illustrating the intricate design and multifaceted challenges that teams must navigate. The combination of these diverse elements ensures that the autonomous systems developed by competing teams are not only capable of handling high-speed segments but also adept at managing complex, low-speed maneuvers with precision and reliability.

Participation in the Formula Student Driverless competition drives innovation by pushing student teams to develop sophisticated autonomous technologies. Teams must achieve

seamless integration of perception, localization, mapping, and control systems to excel in the competition. The Driverless category serves as a catalyst for advancing autonomous driving research, providing a competitive yet collaborative environment where emerging engineers can test and refine their solutions in real-world-like conditions.

By competing in these rigorous events, UTS Motorsport aims to contribute to the evolution of autonomous driving technologies, fostering a culture of excellence and innovation that prepares students to tackle the future challenges of intelligent transportation systems

1.4. Our approach

In response to the demands of the Formula Student Driverless competition, our project adopts a streamlined and efficient approach by utilizing a monocular camera for real-time detection and mapping of traffic cones. The decision to rely on a single camera is driven by the desire to create a lightweight and cost-effective system, reducing the complexity associated with multi-sensor configurations. By focusing on monocular vision, we aim to explore the feasibility of achieving accurate and reliable Simultaneous Localization and Mapping (SLAM) using minimal sensor input. Our methodology incorporates advanced techniques such as the Extended Kalman Filter (EKF) for enhanced tracking, robust feature-matching algorithms for precise visual odometry, and an optimized cone detection pipeline. Additionally, to manage computational efficiency, especially with a high density of landmarks, we implement Submap Joining, dividing the environment into smaller local maps that are later integrated into a cohesive global map. This layered approach ensures real-time performance and scalability, enabling the autonomous racecar to navigate effectively and reliably within the competition environment.

By leveraging these strategies, our approach seeks to deliver a robust and adaptable SLAM solution tailored for the high-speed and dynamic conditions of autonomous racing. This project not only aims to excel in the Formula Student Driverless competition but also contributes valuable insights into the development of efficient and effective autonomous driving systems.

2. LITERATURE REVIEW

In recent years, significant advancements have been made in the field of driverless vehicles, driven by developments in computer vision, robotics, and machine learning. Early studies extensively addressed the challenges of autonomous racing vehicles (Iagnemma & Buehler, 2006). However, the system presented in this paper differentiates itself by focusing on navigating narrow, unknown race tracks marked solely by cones, introducing unique detection and localization challenges not thoroughly explored in prior literature.

2.1. Perception Systems

A critical component of autonomous driving is the perception system, responsible for accurately detecting and interpreting environmental markers such as traffic cones. Traditional systems often employ a combination of LiDAR and vision-based sensors to detect cones, estimate their 3D poses, and determine their colors, enhancing robustness through sensor redundancy (Gosala et al., 2019). Given that the tracks used in this research are exclusively demarcated by colored cones, a perception system capable of precise cone detection and pose estimation under varying conditions is essential.

Prior literature in high-performance autonomy has largely concentrated on control systems (Kabzan et al., 2019), motion planning (Caporale et al., 2019), full-vehicle integration (Howard et al., 2017), and multi-sensor fusion (Gosala et al., 2019). While these studies have significantly advanced the field, they often overlook the complexities involved in developing a comprehensive visual perception system. This research addresses these gaps by extending current knowledge in visual perception, specifically targeting the challenges of real-time, monocular vision-based cone detection and mapping.

Other prior works related to computer vision tasks have explored more specific solutions, such as monocular depth estimation (Roy & Todorovic, 2016; Zhan et al., 2018), stereo depth and pose estimation (Barrois et al., 2009), fused monocular/stereo pose estimation (Dhall et al., 2019), 2D object detection (Redmon et al., 2016), obstacle detection (Lee et al., 2017), and instance segmentation (Chen et al., 2018). These papers provide valuable insights for system design but often ignore critical system-level challenges that arise in real-world applications, such as training dataset domain mismatch, lens effects, and timing issues impacting the latency-accuracy trade-off.

To navigate a track marked by cones, accurate detection of cones in images is paramount. Traditional computer vision techniques, such as the Viola-Jones algorithm (pedestrian detection using Histograms of Oriented Gradients (HOG) combined with Support Vector Machines (SVM) (Dalal & Triggs, 2005), laid the groundwork for object recognition tasks. However, these methods are limited in efficiency and scalability. Recent advancements have seen the adoption of Convolutional Neural Networks (CNNs) and models like YOLO (Redmon & Farhadi, 2016), which offer efficient, single-pass image processing capabilities for object detection. In our work, we employ a modified version of YOLOv8 for real-time traffic cone detection. To facilitate robotics applications, accurate 3D positioning of detected cones is essential. We utilize a CNN-based keypoint detection method (Dhall et al., 2019) combined with geometric information to estimate cone poses through 2D-3D correspondences.

2.2. Motion Estimation

Accurate motion estimation is vital for autonomous navigation. Several filtering techniques, including the Extended Kalman Filter (EKF), Unscented Kalman Filter, and Particle Filter (Thrun et al., 2005), have been explored in the literature. The EKF was chosen for this project due to its computational efficiency and effectiveness in handling mildly nonlinear systems influenced by Gaussian noise (Xue & Schwartz, 2013). The EKF-based motion estimation system integrates measurements while rejecting outliers through the chi-squared test and a variance-based drift detector.

2.3. Visual Odometry

Visual odometry using a single camera presents challenges, notably the scale ambiguity in estimated poses. In unknown environments, the camera pose is incrementally determined by computing feature correspondences between consecutive images and then calculating the relative pose from these correspondences. However, resolving the scale ambiguity is a significant challenge. Traditional methods address this by assuming known camera height and utilizing ground surface information (Choi et al., 2011). Recent approaches have incorporated deep learning to enhance odometry estimation (Costante et al., 2016). Our method leverages the detected cone positions to estimate depth between two poses, thereby improving accuracy without additional sensors.

2.4. Simultaneous Localization and Mapping (SLAM)

Mapping and localization are fundamental for autonomous navigation, particularly in unfamiliar environments. Numerous online SLAM methods, including graph-based approaches (Engel et al., 2014; Mur-Artal & Tardos, 2017; Labbé & Michaud, 2018), have been proposed. However, racing tracks characterized by sparse, color-coded cones present unique challenges, such as indistinguishable landmarks and a lack of unique identifiers. This problem lends itself well to a combination of EKF and graph-based SLAM. Submap joining, although complex and computationally intensive, is suitable in this scenario due to the sparsity and distinguishability of the environment (He et al., 2011; Huang et al., 2009).

3. METHODOLOGY

3.1. System Overview

The SLAM solution for Formula Student racing, illustrated in Figure 3, starts with input from a monocular camera, which provides data for both Vision Cone Detection and Visual Odometry. The Vision Cone Detection module identifies cones marking the track, while Visual Odometry estimates the vehicle's movement, adjusting for scale using detected cones. Together, these modules form the perception system, detailed further in the Perception section.

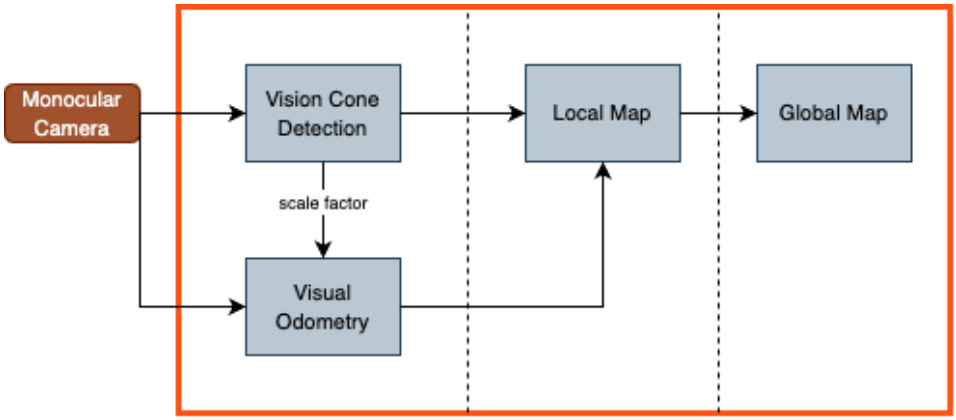

Figure 3. Architecture of the SLAM pipeline

The system's SLAM algorithm operates in two phases: Local Map Building and Global Mapping. The Local Map captures immediate surroundings to support quick decision-making, feeding information back to improve odometry accuracy. This local data is then integrated into a comprehensive Global Map of the track. Subsequent sections will detail each component, illustrating how they work together to optimize perception and mapping under the dynamic conditions of autonomous racing.

3.2. Perception

The perception pipeline's primary goal is to provide real-time, accurate estimates of cone positions, colors, and associated uncertainties. These cones serve as essential landmarks, enabling the SLAM module to construct a reliable map that guides the autonomous car along the track. To achieve this, a camera-based multi-object detection algorithm is employed, supplemented by additional methods to accurately determine the 3D positions of cones solely from monocular images. This pipeline ensures that the vehicle can navigate effectively using a

minimal sensor setup, relying on precise cone detection and positioning to inform SLAM and support autonomous navigation in the high-speed racing environment.

3.2.1. Cone Detection Algorithms

The pipeline illustrated in Figure 4 is developed to detect traffic cones on tracks and determine their 3D positions using a monocular camera. It consists of three primary phases: multi-object detection via a neural network (NN), keypoint regression, and pose estimation. The process begins by identifying cones in the camera frame and extracting keypoints within their bounding boxes based on the basic geometric features of cones. Following this, the 3D position of each cone is estimated using the Perspective-n-Point (PnP) algorithm.

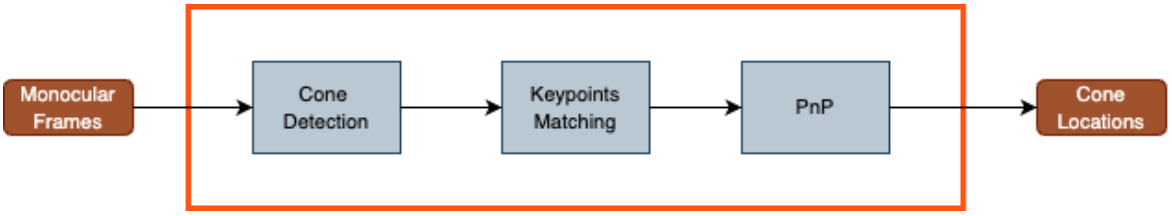

Figure 4. Pipeline of cone detection and position estimation

To accurately estimate the 3D locations of multiple cones from a single image, it is essential to detect cones and classify them by color—blue, yellow, or orange. YOLOv8, a real-time, high-performance object detection network, was chosen for this purpose (Varghese & Sambath M, 2024). This network was specifically trained to distinguish between the three types of cones, leveraging pre-existing weights as priors to enhance training efficiency. Using YOLOv8 allows the model to provide robust, accurate bounding box outputs around detected cones and confidence scores for each detection. Additionally, YOLOv8's fine-tuning capability with varied image conditions, such as low brightness and rainy weather, using the FSOCO dataset (Vödisch et al., 2022), makes it particularly well-suited for this application. This adaptability ensures robust detection performance across diverse environmental conditions, enabling accurate cone identification and classification even under challenging lighting or weather scenarios, which is depicted in Figure 5.

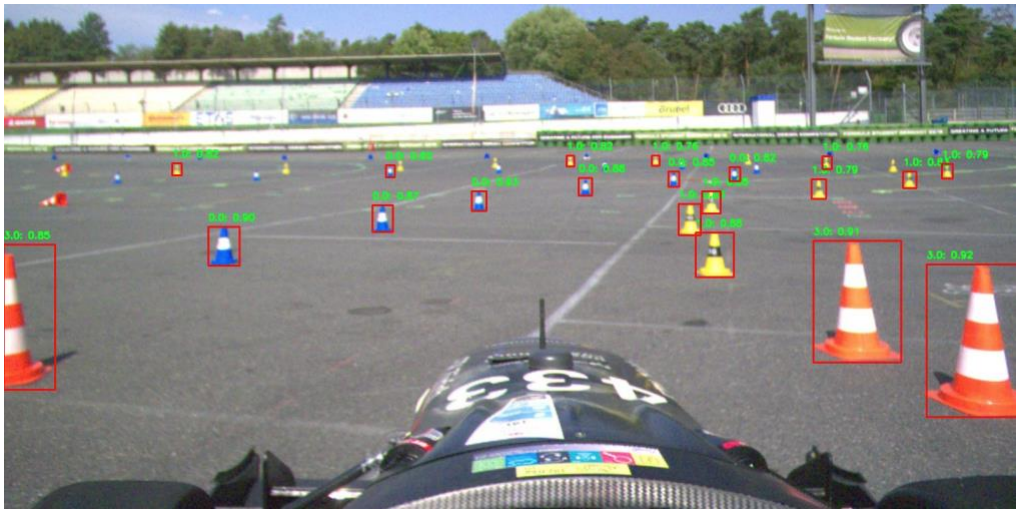

Figure 5. YOLO cones detection and classification

3.2.2. Keypoint Matching and Regression

To retrieve the 3D positions of objects from a single image, additional geometric information is required, as depth information alone cannot be derived from a single frame. This problem, which is inherently ambiguous due to factors like scale and limited visual data, can be addressed by incorporating prior knowledge of the objects' geometry along with the 2D information available in the image. Dhall et al. (2019) proposed a "keypoint regression" approach that utilizes prior knowledge of an object's shape and dimensions to estimate specific feature points on the image, which can be mapped to their corresponding 3D coordinates within a reference frame $\mathcal{F}_\omega$ (Dhall et al., 2019), as illustrated in Figure 6a. This method uses up to seven keypoints to match with points on the real cone, but handling this many keypoints introduces uncertainty in detection.

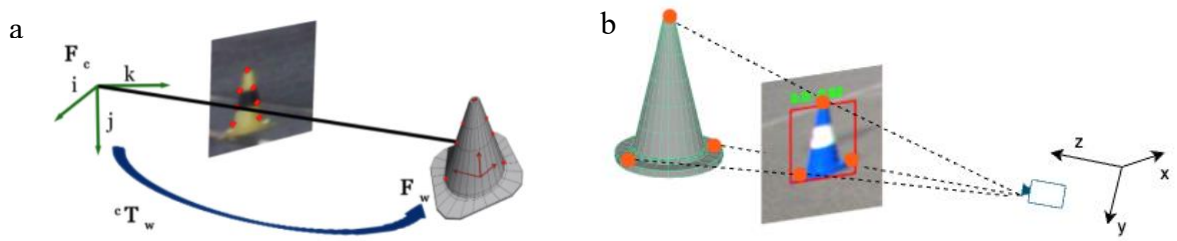

Figure 6. A schematic showing the matching of 2D-3D points and the calculation of the transformation between the camera and world frames. (a) Method following Dhall et al. (2019), (b) My method

To address the limitations of Dhall's approach, a novel method was developed, as depicted in Figure 6b. Instead of training an additional neural network model to detect numerous keypoints around the cone shape, this method reduces complexity by identifying only three primary keypoints, representing the vertices of a triangular shape. The YOLOv8 model was trained to fit a bounding box precisely around the cone. The two bottom vertices of this bounding box correspond to two base keypoints, while the peak is approximated as the midpoint of the bounding box's top edge (Figure 6b). This simplified approach reduces computational costs without requiring a trade-off in accuracy for the cones' location estimation.

3.2.3. Cone Location Estimation

The camera coordinate system is denoted as $\mathcal{F}_c$, and the world coordinate system as $\mathcal{F}_\omega$. For simplified computation of the 3D positions of keypoints, $\mathcal{F}_\omega$ is established at the base of the cone. Detected keypoints within the image are treated as 2D points, which are then corresponded to specific points on the 3D model of a cone. Utilizing these 2D-3D correspondences alongside the camera's intrinsic parameters, the Perspective-n-Point (PnP) algorithm is employed to estimate the pose of each detected cone. This process involves determining the transformation cT_ω between the camera and world coordinate systems. Since $\mathcal{F}_\omega$ is positioned at the base of the cone within an arbitrary location in $\mathbb{R}^3$, this transformation accurately represents the pose that needs to be estimated, as illustrated in Figure 7.

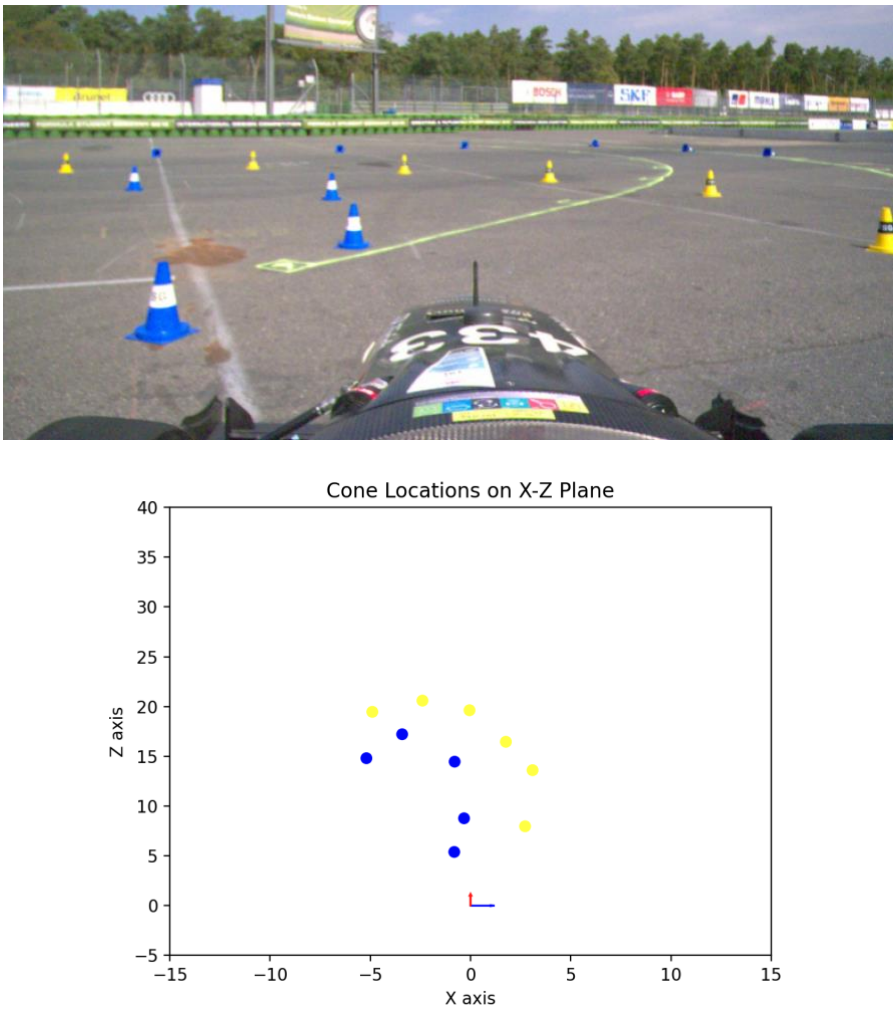

Figure 7. Estimated Cone Location

To achieve precise estimation of the cone's position, a non-linear variant of the PnP algorithm is utilized. Specifically, SQPnP, a fast and globally convergent non-polynomial PnP solver, is employed as it requires only three matching points between the object and the image. SQPnP reformulates the PnP problem as a quadratically constrained quadratic program and resolves it by conducting local searches around specific feasible points, efficiently locating the global minima within a few iterations (Terzakis & Lourakis, 2020). This approach enhances computational efficiency while maintaining accurate pose estimation. By applying SQPnP to the output from the keypoint regressor and the pre-defined 3D correspondences, the pipeline effectively estimates the pose transformations of multiple cones from a single image.

3.3. Visual Odometry

The visual odometry pipeline for the monocular camera setup is illustrated in Figure 8. This odometry system was designed specifically for this project, leveraging foundational computer vision algorithms rather than high-level library tools, and was refined in collaboration with Prof. Shoudong Huang. Developing monocular odometry posed significant challenges, requiring extensive parameter tuning and research to achieve reliable results based on the dataset used in this project.

Unlike conventional approaches such as ORB-SLAM, which use triangulation to create map points as anchors for estimating camera motion (Tardos, 2016), this method focuses on reconstructing a map of traffic cones. Here, the cones serve as primary landmarks, allowing for refined estimation of the vehicle's relative motion between consecutive camera frames. Further details on this approach and the underlying algorithms are provided in the following sections.

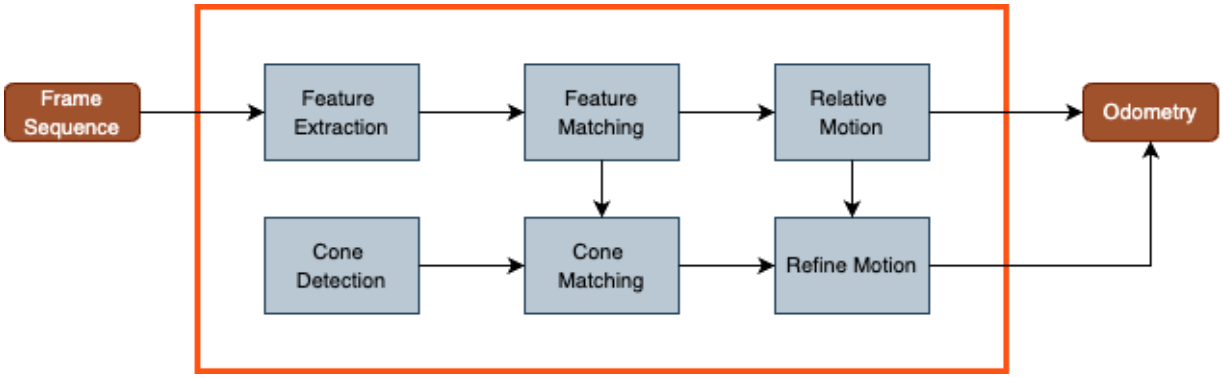

Figure 8. Visual Odometry pipeline.

3.3.1. Feature Extraction

The ORB (Oriented FAST and Rotated BRIEF) detector and descriptor are used to identify keypoints and extract descriptors. ORB, which combines the FAST (Features from Accelerated Segment Test) detector with the BRIEF (Binary Robust Independent Elementary Features) descriptor, offers both rapid computation (approximately 10 times faster than SURF) and strong performance in detecting keypoints and capturing distinctive local features.

To ensure the accuracy of feature extraction, the input image from the camera is pre-processed to remove parts of the car's hood, as shown in Figure 9a, to avoid duplicate keypoints and improve matching quality for odometry. Figure 9b illustrates the result after this pre-processing step, where the hood is excluded from the image.

a b

Figure 9. Image Pre-proccessing. Car Filter-out

To achieve an even distribution of keypoints across each frame, a grid sampling strategy is applied, extracting up to 1500 keypoints uniformly. This approach ensures that keypoints are spread evenly, enhancing robustness in the odometry process.

3.3.2. Feature Matching

The **FLANN** (Fast Library for Approximate Nearest Neighbors) matcher is utilized in this project for its high speed and accuracy in feature matching. However, FLANN alone may produce matches that are unevenly distributed across the image, potentially overlooking important regions. To address this, Grid-Based Motion Statistics (GMS) matching, as proposed by Bian et al. (2017), is incorporated to enhance spatial uniformity and introduce scale invariance, ensuring that matches remain effective even when the scale of features varies between frames.

To further refine the matching process, a filtering method based on adapted from the *SLAM Book*, is applied across all matched point pairs, not limited to FLANN. Here, d_{min} represents the minimum distance among all pairs. For a match to be retained, the distance $dist\left(p_{A_i}, p_{B_j}\right)$ between a point p_{A_i} in frame A and a point p_{B_j} in frame B must satisfy $dist\left(p_{A_i}, p_{B_j}\right) < \max(30, 2 * d_{min})$ where 30 and 2 are empirically chosen hyperparameters. This filtering step ensures that only the most reliable matches are retained.

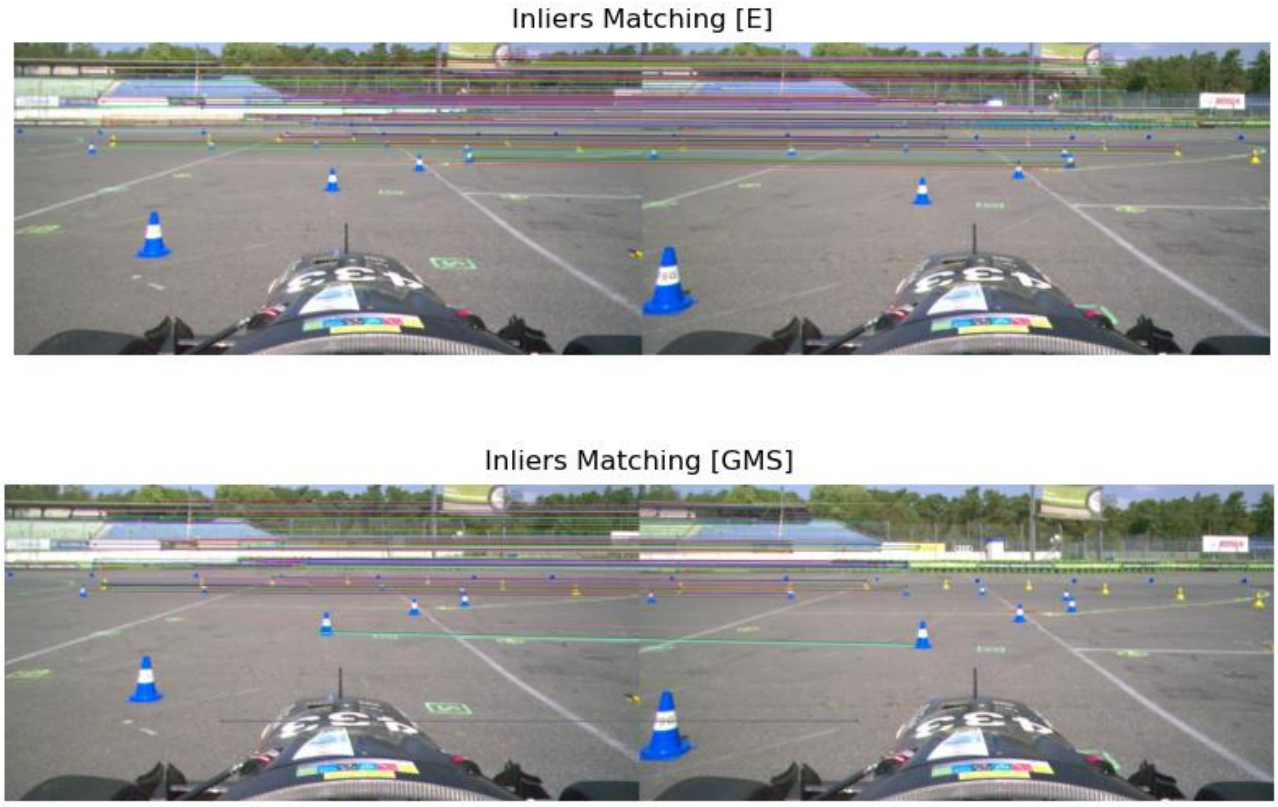

Figure 10. Inlier Matching between two frames

By combining FLANN for fast matching, GMS for uniform distribution and scale invariance, and d_{min}-based filtering for selecting the best matches, the feature matching process becomes robust and accurate, supporting reliable visual odometry.

3.3.3. Relative Motion

The rotation and translation between each frame are decomposed from the Essential Matrix combined with RANSAC. The CV2 library contains all built-in functions necessary for these calculations.

Another method involves calculating the Homography Matrix. The essential score and Homography score are then compared to determine which matrix is more accurate. This approach is referenced from Chapter 7 of Slambook.

We also calculate the symmetric transfer error of both the Homography matrix (H) and the Essential matrix (E):

$$e_E = \frac{1}{N}\sum_{n=1}^{N}\frac{(x_i')^T E x_i}{\|Ex_i\|^2 + \|E^T x_i'\|^2}$$

$$e_H = \frac{1}{N}\sum_{n=1}^{N}\frac{\|Hx_i - x_i'\|^2}{\|x_i\|^2 + \|x_i'\|^2}$$

Where:

- x_i and x_i' are corresponding points in the two frames
- N is the total number of correspondences

The symmetric transfer error serves as a metric to evaluate the accuracy of the estimated transformation matrices. It measures the consistency between the observed correspondences and the projections defined by the matrices E and H.

After calculating both E and H, we compare these errors to determine which matrix more accurately represents the relative motion:

- If $e_H < e_E$, the Homography Matrix is preferred, suggesting that the scene may contain dominant planar elements or that the camera motion is primarily rotational.
- *If* $e_H > e_E$, the Essential Matrix provides a better fit, indicating that the scene's structure and motion are better captured by the combination of rotation and translation without assuming planarity.

This selection process ensures that the most appropriate model is used for subsequent tasks such as camera pose estimation and 3D reconstruction.

3.3.4. Cone Bounding Box Matching:

Cone objects are detected using the YOLO (You Only Look Once) framework, as detailed in Section 3.2. Due to the visual similarity of these cones, accurately matching bounding boxes between consecutive frames presents a significant challenge. Consistent identification of individual cones is essential for estimating their 3D locations within the camera coordinate system, which is critical for motion optimization in monocular visual odometry systems that operate without an absolute scale.

To address the challenge of matching similar cones across frames, we developed a two-step Bounding Box Matching algorithm comprising Algorithm 1 and Algorithm 2.

Algorithm 1 is designed to verify whether keypoints are located within any bounding box. This initial step ensures that only relevant feature points are considered for subsequent matching, effectively filtering out noise and irrelevant data. By accurately determining the presence of keypoints within bounding boxes, Algorithm 1 lays the groundwork for reliable feature correspondence.

Algorithm 1 findBoxContaining

Require: *boxes*: List of boxes $((x1, y1), (x2, y2))$; x, y: Point coordinates
Ensure: *index*: Index of containing box or **null**
  for $index \leftarrow 0$ **to** $length(boxes) - 1$ **do**
    $((x1, y1), (x2, y2)) \leftarrow boxes[index]$
    if $x1 \leq x \leq x2$ **and** $y1 \leq y \leq y2$ **then** *index*
    end if
  end fornull

Once keypoints within bounding boxes are identified, Algorithm 2 facilitates the matching of bounding boxes between consecutive frames. Leveraging feature correspondences obtained from inlier matches after applying RANSAC, Algorithm 2 systematically associates bounding boxes from the previous frame to those in the current frame by evaluating the number of feature matches within each bounding box pair. This step involves counting the number of matches between each pair of bounding boxes and assigning confidence scores based on these counts.

By comparing these match counts, the algorithm assigns confidence scores to potential matches, prioritizing pairs with higher numbers of internal matches. This confidence-based approach ensures that the most reliable associations are selected, enhancing the accuracy of the matching process. A key feature of the algorithm is its ability to avoid duplicate bounding box matches. It enforces a one-to-one correspondence between bounding boxes in the previous and current frames by ensuring that each bounding box is uniquely paired with its best match based on the highest confidence score. This mutual exclusivity prevents scenarios where a single bounding box in one frame is incorrectly matched to multiple bounding boxes in another frame, thereby maintaining the integrity of cone identities across frames. By eliminating duplicate matches, the algorithm ensures that each cone is consistently tracked, which is vital for accurate motion optimization and subsequent 3D reconstruction.

The implementation of the Bounding Box Matching algorithm, incorporating both Algorithm 1 and Algorithm 2, is integrated into the project's codebase. This integration processes the feature matches to generate reliable bounding box pairings systematically. This two-tiered approach not only enhances the robustness of cone identification but also contributes to the overall stability of the visual odometry system. By ensuring that each cone is consistently and accurately matched across frames, the algorithm facilitates precise 3D localization and supports the generation of an up-to-scale 3D map of the environment. This method effectively mitigates the challenges posed by similar-looking cones, enabling reliable performance in dynamic and visually complex settings.

Algorithm 2 findMatches

Require: $Boxes\_A$, $Boxes\_B$: Lists of boxes; $Matches$: List of (a_x, a_y, b_x, b_y)
Ensure: $selected\_pairs$: List of matched (i, j) indices

```
N ← length(Boxes_A); M ← length(Boxes_B)
Initialize counts[N][M] ← 0
for all match in Matches do
    (a_x, a_y, b_x, b_y) ← match
    A_i ← findBoxContaining(Boxes_A, a_x, a_y)
    B_j ← findBoxContaining(Boxes_B, b_x, b_y)
    if A_i ≠ null and B_j ≠ null then
        counts[A_i][B_j] ← counts[A_i][B_j] + 1
    end if
end for
Initialize Max_A[N] ← 0; Top_B_for_A[N] ← empty lists
for i ← 0 to N − 1 do
    for j ← 0 to M − 1 do
        if counts[i][j] > Max_A[i] then
            Max_A[i] ← counts[i][j]
            Top_B_for_A[i] ← [j]
        else if counts[i][j] = Max_A[i] and counts[i][j] > 0 then
            Append j to Top_B_for_A[i]
        end if
    end for
end for
Initialize Max_B[M] ← 0; Top_A_for_B[M] ← empty lists
for j ← 0 to M − 1 do
    for i ← 0 to N − 1 do
        if counts[i][j] > Max_B[j] then
            Max_B[j] ← counts[i][j]
            Top_A_for_B[j] ← [i]
        else if counts[i][j] = Max_B[j] and counts[i][j] > 0 then
            Append i to Top_A_for_B[j]
        end if
    end for
end for
selected_pairs ← empty list
for i ← 0 to N − 1 do
    if Max_A[i] == 0 or length(Top_B_for_A[i]) ≠ 1 then
        continue
    end if
    j ← Top_B_for_A[i][0]
    if counts[i][j] == Max_B[j] and length(Top_A_for_B[j]) == 1 then
        Append (i, j) to selected_pairs
    end if
end for
    selected_pairs
```

3.3.5. Refine Motion

While the cv2.recoverPose function provides initial estimates for the rotation (RRR) and translation (ttt) matrices between two consecutive poses, these estimates often carry a degree of uncertainty that renders them insufficient for reliable Simultaneous Localization and

Mapping (SLAM). The inherent limitations of this initial pose estimation necessitate a refinement process to enhance the accuracy and stability of the motion parameters. This subsection outlines the methodology employed to refine the R and t matrices using optimization techniques, leveraging the established correspondences between cone detections in the 3D world.

Initial Pose Estimation and Its Limitation

The cv2.recoverPose function utilizes the Essential Matrix derived from feature correspondences to estimate the relative rotation and translation between two camera poses. While this method effectively captures the geometric relationship between frames, it is susceptible to inaccuracies due to noise, outliers, and the absence of scale information inherent in monocular systems. Consequently, the R and t matrices obtained directly from recoverPose may exhibit significant uncertainty, undermining their utility in precise SLAM applications.

Leveraging Cone Correspondences for Optimization

To mitigate the uncertainties associated with the initial pose estimates, we exploit the correspondences between detected cones across consecutive frames. Each cone detected in the environment provides a robust feature point with a known 3D location in the camera coordinate system. By establishing consistent matches between cones in successive frames, we obtain a set of reliable 3D correspondences that serve as the foundation for refining the motion parameters.

Optimization Framework

The refinement process is formulated as an optimization problem aimed at minimizing the discrepancy between the predicted and actual positions of the cones in the 3D world. The optimization adjusts the rotation and translation matrices to achieve a better alignment of the cone positions, thereby enhancing the accuracy of the pose estimation.

a. *Least Squares Minimization for Translation Scale*

   The initial translation vector ttt obtained from cv2.recoverPose lacks absolute scale due to the monocular nature of the system. To recover the true scale, we introduce a scaling factor α\alphaα and formulate the following least squares minimization problem:

$$\alpha = \min_{\alpha} \sum_{m} \left|\left| R_{k-1,k} \cdot X_{c(k-1)} + \alpha \cdot t_{k-1,k} - X_{c(k)} \right|\right|^2$$

Here:

- $R_{k-1,k}$ and $t_{k-1,k}$ are the rotation and translation matrices between poses $k-1$ and k.
- $X_{c_{(k-1)}}$ and $X_{c_{(k)}}$ are the 3D coordinates of a cone in the frame of poses $k-1$ and k, respectively.
- m is the number of paring cones.
- The objective is to find the scaling factor α that minimizes the sum of squared errors between the transformed previous cone positions and the current cone positions.

b. Refining Rotation and Translation Parameters in Complex Scenarios

In more complex motion scenarios, such as sharp turns, the uncertainty extends beyond translation scale to the rotation and translation parameters themselves. Given that the vehicle operates predominantly on a planar surface, we constrain the optimization to consider only the yaw angle θ and the horizontal components of the translation vector t_x and t_y. This simplification leverages the known planar movement, reducing the dimensionality of the optimization problem and enhancing computational efficiency.

The rotation matrix is parameterized solely by the yaw angle θ, resulting in a 2x2 rotation matrix R_{2x2}:

$$R_{2X2} = \begin{bmatrix} cos(\boldsymbol{\theta}) & -sin(\boldsymbol{\theta}) \\ sin(\boldsymbol{\theta}) & cos(\boldsymbol{\theta}) \end{bmatrix}$$

The translation vector is confined to the plane:

$$t = \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

The optimization problem is thus extended to refine θ, t_x and t_y simultaneously:

$$\theta, t_x, t_y = \arg\min_{\theta,t_x,t_y} \sum_{m} \left\| \begin{bmatrix} weight_x \\ weight_y \end{bmatrix} * ((R_{2X2} \cdot X_{c_{(k-1)}} + t) - X_{c(k)}) \right\|^2$$

To further refine θ and t_x values. we apply a weighted approach to the error in cone location along the x-axis. Since the car primarily moves forward, small errors in rotation or lateral movement can lead to significant discrepancies. Therefore, we assign a higher weight to the x-axis error $(weight_x)$ than to the y-axis error $(weight_y)$, with values set at 2 and 1, respectively. This formulation emphasizes corrections to the yaw angle and horizontal translation in the x-direction, effectively minimizing the total error across all cone correspondences.

Optimization Procedure

The optimization is performed using a least squares minimization approach, which iteratively adjusts the parameters α, θ, t_x, and t_y to reduce the overall error in the alignment of cone positions. The procedure involves the following steps:

1. Initialization:
    - **Scaling Factor (α):** Initialized at 1, set the maximum and minimum value at 0.1 and 2, respectively.
    - **Yaw Angle (θ):** Extracted from the initial rotation matrix R, focusing solely on the yaw component to align with planar movement.
    - **Translation Components (t_x, t_y):** Derived from the initial translation vector t.
2. Stage 1 – Optimizing α:
    - To calculate α, the algorithm requires at least a single pair of cone correspondences to minimize the error between the scaled translation and the actual displacement observed in the cone positions.
    - This step provides an initial scale factor that brings the translation vector into a more accurate range relative to the environment.
3. Stage 2 – Refining θ, t_x, and t_y:
    - With α optimized, the algorithm proceeds to refine the rotation and translation parameters
    - Using at least two pairs of cone correspondences, the algorithm employs gradient descent to iteratively adjust θ, t_x, and t_y to minimize the sum of squared errors.
    - The gradient descent algorithm updates the parameters in the direction that most reduces the error, continuing until convergence is achieved or the changes fall below a predefined threshold.

3.4. Mapping

Once the cone locations and camera motion data have been accurately collected and refined, the final step in the visual odometry pipeline is to reconstruct the surrounding environment. This reconstruction process culminates in the creation of a comprehensive map, which is essential for navigation and localization tasks. To achieve this, we employ the Extended

Kalman Filter Simultaneous Localization and Mapping (EKF-SLAM) algorithm, a widely recognized and effective method for landmark-based environmental mapping.

3.4.1. Motion Model and Prediction Step

Given that the vehicle operates exclusively on a single planar surface, we simplify the motion model to consider one-dimensional (1D) motion.

The state vector X at time k is defined as:

$$X_k = \begin{bmatrix} x_k \\ y_k \\ \theta_k \\ m_1 \\ m_2 \\ \vdots \\ m_n \end{bmatrix}$$

Where:

- x_k, y_k represent the vehicle's position coordinates.
- θ_k demotes the vehicle's orientation (yaw angle).
- m_i for $i = 1, 2, \dots, n$ are the positions of the observed landmarks (cones).

The motion model updates the vehicle's state based on the previous state and the control inputs derived from the refined motion estimates. Specifically, the position and orientation are updated as follows:

$$\bar{X}_k = F(X_{k-1}, \theta, t_x, t_y)$$

$$\begin{bmatrix} x_k \\ y_k \end{bmatrix} = R(\theta_{k-1}). \begin{bmatrix} x_{k-1} \\ y_{k-1} \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

$$\theta_k = \theta_{k-1} + \Delta\theta$$

$$\bar{P}_k = \nabla F . P_{k-1} . \nabla F^T + Q$$

Here:

- $\bar{X}_k$ is the predicted state vector, Q is the motion noise
- ∇F is the Jacobian of function F evaluate at X_{k-1}
- $R(\theta_k - 1)$ is the rotation matrix corresponding to the previous orientation $(\theta_k - 1)$.
- t_x, t_y are the translation components derived from the optimized motion parameters.

- $\Delta\theta$ is the change in orientation obtained from the motion refinement step.

This model encapsulates the vehicle's planar movement by updating its position and orientation based on the estimated rotation and translation between consecutive poses.

3.4.2. Observation Model and Update Step

The observation vector Z consists of the positions of all detected landmarks:

$$Z_k = H(\bar{X}_k)$$

Update the state vector using observation:

$$X_k = \bar{X}_k + K(Z_k - H(\bar{X}_k))$$

$$P_k = \bar{P}_k + K.S.K^T$$

where the innovation covariance S (here $Z_k - H(\bar{X}_k)$ is called innovation) and the Kalman gain K are given by:

$$S = \nabla H.\bar{P}_k.\nabla H^T + W$$

$$K = \bar{P}_k.\nabla H^T.S^{-1}$$

where ∇H is the Jacobian function of H evaluate at $\bar{X}_k$

3.4.3. Add new landmark to current model

When the new landmark is observed for the first time, we initialize the position of new landmark and add to the state vector:

$$X_k = \begin{bmatrix} \bar{X}_k \\ H(\bar{X}_k) \end{bmatrix}$$

$$P_k = \begin{bmatrix} \bar{P}_k & \bar{P}_k.H_x^T \\ H_x\bar{P}_k & H_x.\bar{P}_k.H_x^T + H_z W H_z^T \end{bmatrix}$$

where H_x and H_z are the Jacobian function evaluated at $\bar{X}_k$ and new observed Z_i

3.4.4. Data Association

In our application, data association involves matching detected cones in the current frame to existing landmarks in the map using Nearest Neighbor (NN) matching and Chi-Square statistical tests, leveraging the cone variance information derived from the covariance matrix.

3.4.5. Submap Joining

To enhance computational efficiency, Submap Joining (Huang, 2009) is employed. Rather than building a single global map, multiple local maps are created using EKF-SLAM, each containing around 30–40 landmarks. These local maps are then joined to form a coherent global map. This method significantly reduces computation time by limiting the scope of each EKF update, allowing for real-time performance even in environments with a high density of landmarks.

4. RESULTS AND DISCUSSION

This chapter presents a comprehensive analysis of the accuracy and performance of each module within the proposed pipeline, including Perception, Visual Odometry, and SLAM, as well as the data association method under investigation. We evaluate the initial implementation using the Extended Kalman Filter (EKF) and compare it against the Submap Joining approach to assess their respective effectiveness. Additionally, our data association strategy combines Nearest Neighbor matching with Computer Vision-based Bounding Box Matching to enhance the reliability of landmark correspondences. Through these comparisons, we aim to demonstrate the strengths and limitations of each method, highlighting their impact on the overall system accuracy and performance.

4.1. Perception Validation and Results

The 3D position estimation error is calculated based on the values obtained before and after optimization. As expected, the error tends to increase with the distance to the cone; however, it remains under 1.0 m at a distance of 20 m, as illustrated in Figure 11.

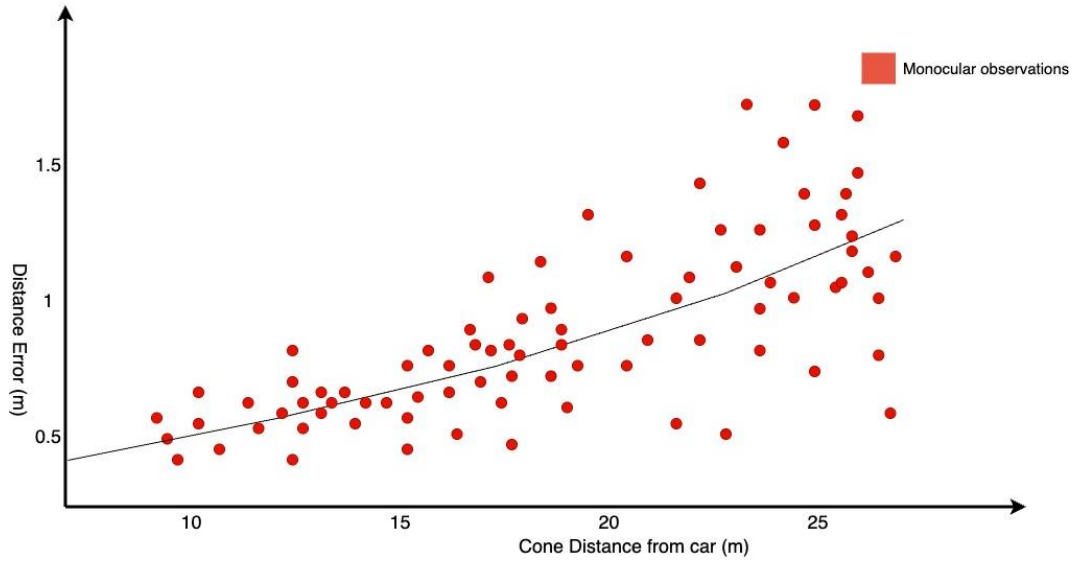

Figure 11. The error of cone estimated position

This level of accuracy is crucial, as the precise localization of cones directly impacts the construction of an accurate map. Consequently, only cones within a range of less than 20 m are used to calculate odometry and feed into the Extended Kalman Filter (EKF). This selection

criterion ensures reliable inputs for the EKF, enhancing the stability and accuracy of the overall mapping and localization process.

4.2. Visual Odometry Validation and Results

4.2.1. Cone Bounding Box Matching

Detecting cone correspondences between frames is a challenging task in visual odometry due to the visual similarity of cones and the limited variation in background across consecutive frames. As discussed above, the visual similarities make it difficult to establish reliable correspondences, especially since the keypoints around the cones often lack high matching confidence. This limitation results in fewer matches between cone keypoints when using traditional FLANN (Fast Library for Approximate Nearest Neighbors) matching, as shown in Figure 12.

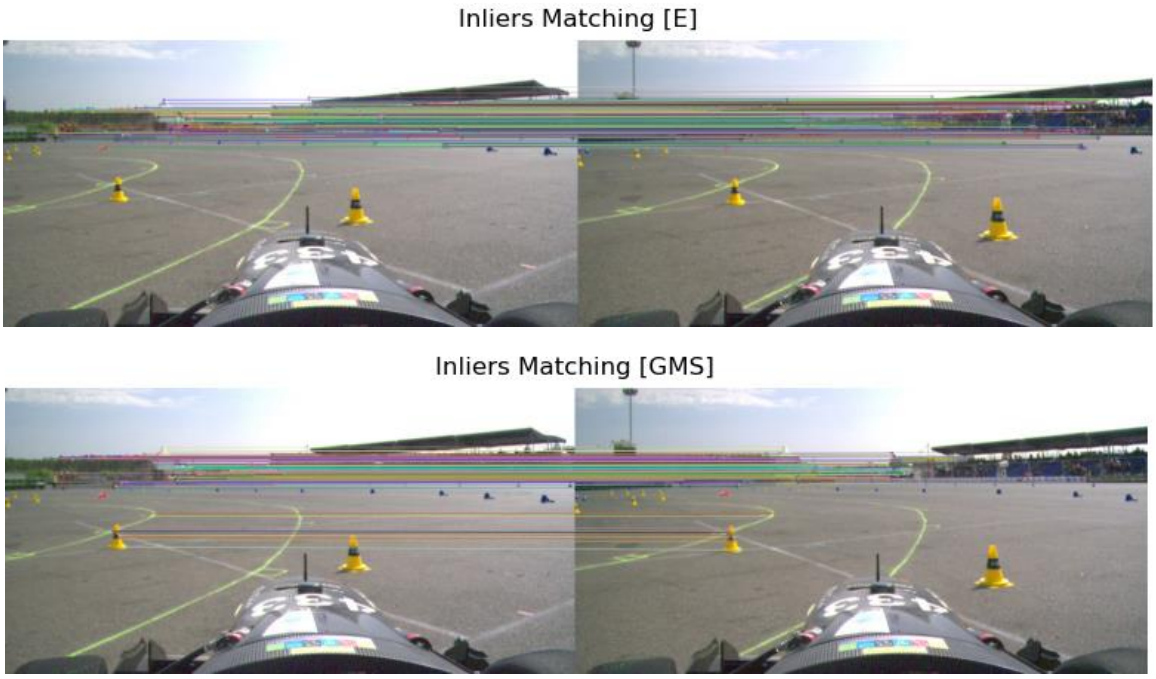

Figure 12. (a) FLANN+RANSAC Matching Technique, (b) GMS Matching Technique

However, applying Grid-based Motion Statistics (GMS) matching, which is scale-invariant, yields more consistent matches among cone keypoints, as seen in Figure 12b. The improved number of matches provided by GMS helps to maintain reliable correspondences, which is

essential for the cone bounding box algorithm and for achieving accurate absolute scale calculations in visual odometry.

To optimize matching performance, combining FLANN and GMS techniques increases the number of matches between cone bounding boxes while ensuring they remain well-distributed across the frame. This hybrid approach provides a balance between the broad matching capacity of FLANN and the stability of GMS, enhancing the robustness of cone tracking across frames. Figure 13 demonstrates how these two matching techniques improve the number of cone pairs, resulting in more accurate and reliable cone correspondences.

Figure 13. Cone Matching and Tracking with index

Since cones serve as landmarks for calculating relative motion between consecutive camera poses, maintaining accurate correspondences is critical for deriving precise 3D motion. The improved matching techniques enhance the tracking of these cone landmarks, contributing to more reliable visual odometry and a more accurate estimation of motion in the 3D environment.

4.2.2. Odometry Optimization

Figure 14a presents the car's trajectory scaled up without the application of any optimization or SLAM methods, with all translation values normalized to 1. When this raw odometry data is processed using the Extended Kalman Filter (EKF), as depicted in Figure 14b, we observe that the robot's pose returns to its original starting point; however, its orientation is incorrect. This discrepancy highlights the limitations of the unoptimized odometry data in accurately capturing the vehicle's true motion.

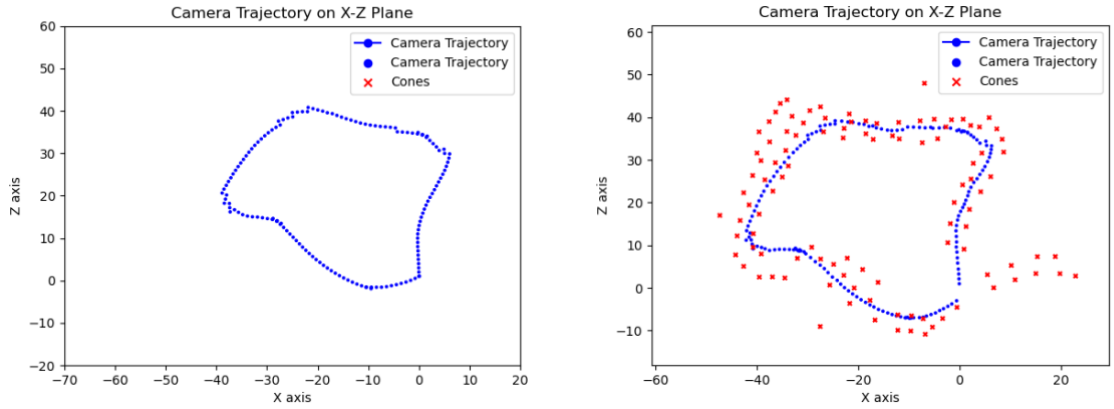

Figure 14. Odometry without Scale factor

After implementing the optimization techniques outlined earlier, there is a significant improvement in the map's accuracy, as shown in Figure 15. The optimized trajectory now appears more realistic: the car decelerates when taking turns and accelerates along straight paths, which aligns with expected physical behavior. Additionally, the car's orientation during loop closure events is more accurate, indicating that the optimization effectively refines the pose estimates.

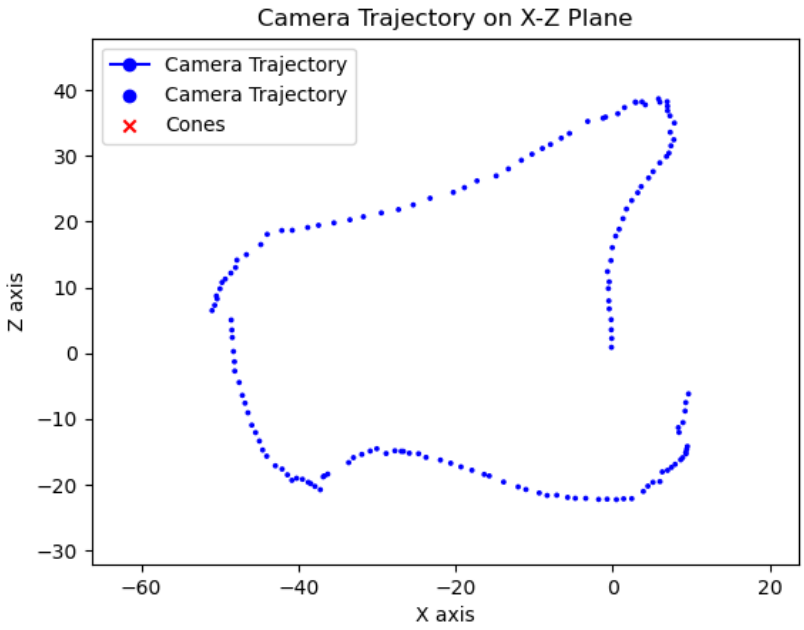

Figure 15. Odometry with Scale Factor and Optimization

Despite these improvements, the loop closure is not yet detected in the map due to the ineffective integration of landmarks for localization. The lack of properly implemented

landmarks means that while the trajectory is more accurate, the system cannot fully recognize when it revisits previously mapped areas. This issue underscores the need for further enhancement in the landmark integration process to achieve reliable loop closure detection and improve the overall robustness of the mapping system.

4.3. SLAM

Figure 1**6** illustrates the significant enhancements achieved through the application of EKF-SLAM immediately following each camera pose estimation. The car's trajectory successfully forms a closed loop when it returns to the starting point, demonstrating the system's ability to maintain accurate localization over extended paths. Additionally, the positions of the cones within the map are notably more precise, exhibiting reduced noise and uncertainty compared to trajectories without SLAM optimization. These improvements highlight EKF-SLAM's effectiveness in refining pose estimates and producing a reliable and accurate environmental map**.**

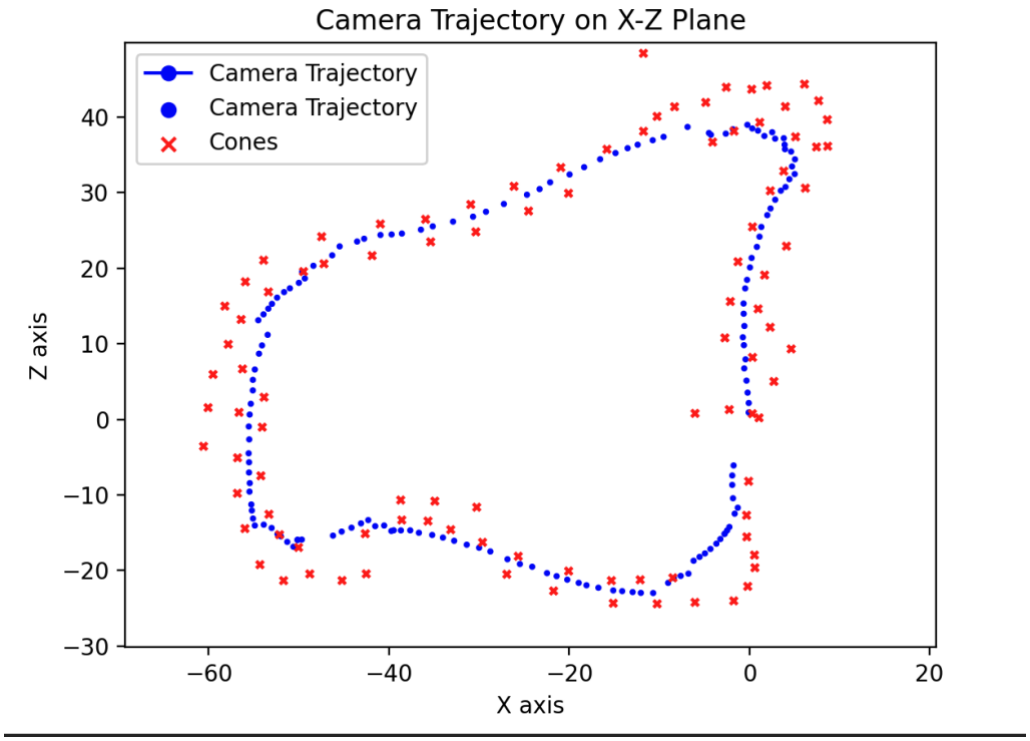

Figure 16. EKF Map and Trajectory

However, managing a large number of cones—approximately 300 in this map—introduces substantial computational challenges. As the number of observed cones incorporated into the

state vector increases, the processing time of the EKF-SLAM algorithm correspondingly grows, resulting in decreased system performance and higher latency. To address this issue and enhance computational efficiency, we implemented a Submap Joining approach based on the methodology proposed by Huang (2009). This strategy involves dividing the global map into multiple smaller local submaps, each containing around 30 poses and 50 landmarks.

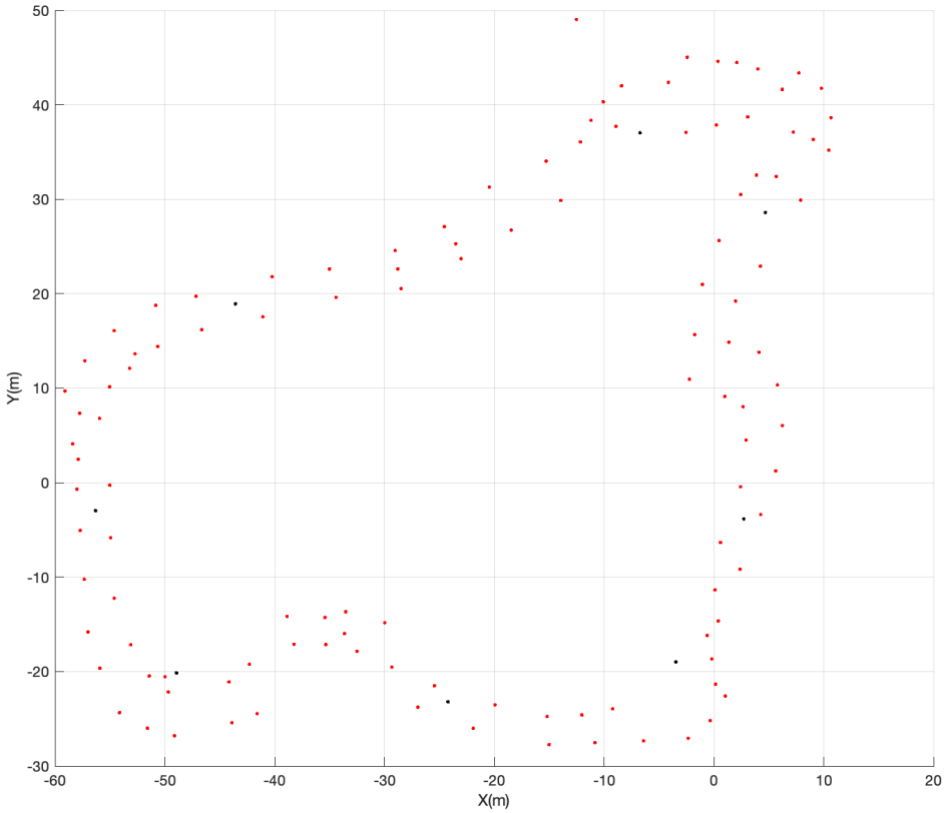

Figure 17. EKF Local Map + Submap Joining

By segmenting the map into these manageable local submaps, we significantly reduce the computational load on the EKF-SLAM algorithm. Once a complete lap is completed, these local maps are recombined into a unified global map using the Submap Joining technique. Figure 17 demonstrates that the accuracy of the global map remains comparable to that of the full EKF-SLAM approach, despite utilizing only half the computational resources. Specifically, the traditional EKF-SLAM processed all data across 156 frames in 385.74 seconds, whereas the combined EKF and Submap Joining method reduced the processing time to 165.23 seconds. This optimization allows each frame to be processed in approximately 1.05 seconds, approaching real-time performance.

Figure 18. Frames during sharp turn

Although the algorithm demonstrates strong overall performance, there are notable errors in certain complex scenarios, as depicted in Figures 1**8**a and 1**8**b. During sharp turns, the camera is unable to capture both sides of the track, resulting in the observation of only yellow cones while blue cones become obscured. This lack of data during such maneuvers leads to significant odometry errors. As highlighted in Figure 1**9**a, errors are particularly evident during two sharp turns, where some blue cones are missing when the car turns left and vice versa. These missing landmarks compromise the accuracy of pose estimation and map consistency**.**

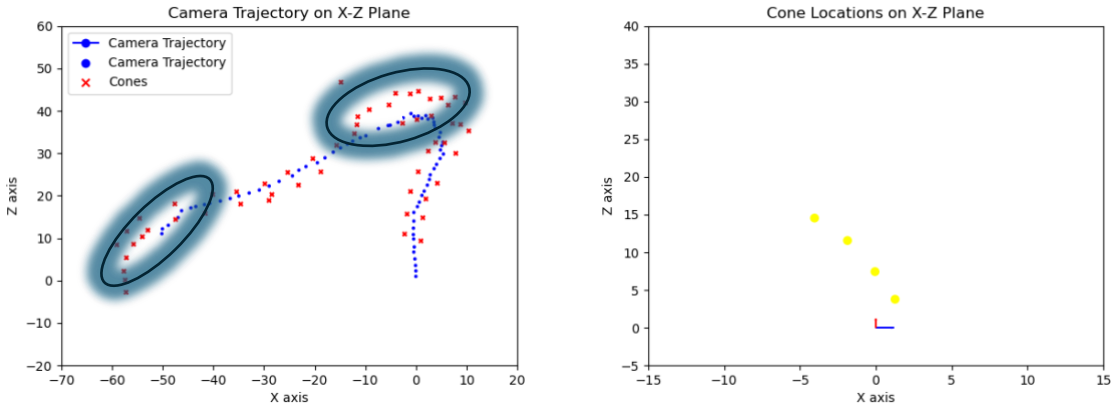

Figure 19. Uncertainty during sharp turn

Despite these challenges, the implemented optimization techniques effectively enhance mapping accuracy and computational efficiency. However, the current system has not yet fully addressed the issue of missing landmark data during complex maneuvers. Future work will focus on developing more robust data association and landmark detection methods to mitigate these errors, ensuring consistent localization and mapping even in challenging scenarios.

5. CONCLUSION

At the outset of this thesis, our primary objective was to develop and implement a robust SLAM algorithm for the UTS Motorsport team, aimed at delivering consistent and accurate results for the Formula Student Driverless competition. Throughout the project, we navigated various constraints, including computational limitations, sensor uncertainties, and the complexities inherent in dynamic racing environments. Despite these challenges, the outcomes achieved are satisfactory and align with the objectives set at the beginning of this thesis.

The developed SLAM system effectively integrates perception, visual odometry, and mapping modules, demonstrating reliable performance in detecting and tracking traffic cones essential for autonomous navigation on competition tracks. The combination of the Extended Kalman Filter (EKF) and Submap Joining techniques has proven to enhance both the accuracy and computational efficiency of the system, enabling real-time performance crucial for autonomous racing scenarios.

Moreover, the project has provided valuable insights and practical experience in applying SLAM methodologies within the unique context of Formula Student competitions. The lessons learned extend beyond technical proficiency, fostering a deeper understanding of the interplay between perception, localization, and decision-making in high-stakes, dynamic environments. These insights are expected to drive further advancements and improvements in future autonomous racecar projects, contributing to the ongoing evolution of intelligent transportation systems.

5.1. Future Work

While the current SLAM system meets the foundational objectives, several areas present opportunities for enhancement to further improve localization accuracy and system robustness.

5.1.1. Sensor Fushion

To advance the perception capabilities of the autonomous vehicle, integrating additional sensors such as Inertial Measurement Units (IMUs) and LiDAR is a critical next step. Sensor fusion will combine data from these sources with the existing monocular camera and odometry, significantly enhancing the system's accuracy and reliability. This integration will provide redundancy, allowing the system to maintain precise localization even in scenarios where individual sensors may fail or provide unreliable data.

5.1.2. Kalman Filter Tuning

One of the primary challenges in refining the SLAM system is the tuning of measurement and process noise parameters within the Kalman Filter. Accurate tuning is essential to balance the trust between sensor measurements and motion estimates, ensuring optimal filter performance. Future work will focus on developing systematic methods for testing and analyzing sensor data to fine-tune these parameters, thereby enhancing the accuracy and stability of the SLAM system.

5.1.3. Integrated Cone Detection

Relying solely on a monocular camera for cone detection introduces significant noise and uncertainty in cone location estimation. To address this, future efforts will involve integrating cone detection with additional sensing modalities, such as LiDAR and stereo cameras. This integration will improve the robustness of cone detection, providing more accurate and reliable landmark data for the SLAM system. Enhanced cone detection algorithms will also be developed to better handle occlusions and varying lighting conditions, ensuring consistent performance across diverse track scenarios.

5.1.4. Improved Data Association Algorithm

As highlighted in the results, the similarity of cones poses challenges for accurate data association, particularly during sharp turns where the camera cannot capture both sides of the track. This can lead to significant odometry errors due to missing landmarks. Future work will explore advanced data association techniques that leverage both spatial proximity and unique identifiers for cones. Implementing algorithms that combine Nearest Neighbor matching with statistical validation methods, such as the Chi-Square test, will enhance the reliability of landmark matching. Additionally, introducing unique markers, such as larger orange cones for loop closures, can aid in accurately recognizing and associating cones during complex maneuvers, thereby preventing incorrect mappings and improving overall system accuracy.

By addressing these areas, the SLAM system will become more robust and capable of handling the dynamic and challenging conditions of autonomous racing. These enhancements will not only improve localization accuracy but also ensure the system's scalability and reliability for future competitions and real-world applications.

5.2. Acknowledgements

First and foremost, I would like to express my sincere gratitude to my supervisor Ph.D Shoudong Huang who has helped me to complete this work successfully. Under his helpful guidance and advice, I was able to overcome obstacles in the process of doing my research work as well as writing my thesis. Additionally, I would like to thank all members of the committee for making my defense an enjoyable experience, and for their insightful feedback and outstanding suggestions, sincerely thank to you.

I would also like to give my warmest appreciation to my family as a whole for their generous assistance and understanding when me undertaking my research work and writing my thesis. Your prayer for me has motivated me this far.

Lastly, I would like to acknowledge my own dedication and perseverance, which have been crucial in bringing this graduation project to fruition.

6. REFERENCE

Barrois, B., Stela Hristova, Wöhler, C., Franz Kummert, & Hermes, C. (2009). 3D pose estimation of vehicles using a stereo camera. *IEEE Intelligent Vehicles Symposium*. https://doi.org/10.1109/ivs.2009.5164289

Bian, J., Lin, W.-Y., Matsushita, Y., Yeung, S.-K., Nguyen, T.-D., & Cheng, M.-M. (2017). GMS: Grid-based motion statistics for fast, ultra-robust feature correspondence. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 2828–2837). https://doi.org/10.1109/CVPR.2017.302

Cadena, C., Carlone, L., Carrillo, H., Latif, Y., Scaramuzza, D., Neira, J., Reid, I., & Leonard, J. J. (2016). Past, Present, and Future of Simultaneous Localization And Mapping: Towards the Robust-Perception Age. *IEEE Transactions on Robotics*, *32*(6), 1309–1332. https://doi.org/10.1109/TRO.2016.2624754

Caporale, D., Settimi, A., Massa, F., Amerotti, F., Corti, A., Fagiolini, A., Guiggiani, M., Bicchi, A., & Pallottino, L. (2019). *Towards the Design of Robotic Drivers for Full-Scale Self-Driving Racing Cars*. IEEE Xplore. https://doi.org/10.1109/ICRA.2019.8793882

Chen, L., Hermans, A., Papandreou, G., Schroff, F., Wang, P., & Adam, H. (2018). MaskLab: Instance Segmentation by Refining Object Detection with Semantic and Direction Features. *Computer Vision and Pattern Recognition*. https://doi.org/10.1109/cvpr.2018.00422

Choi, S., Joung, J. H., Yu, W., & Cho, J. I. (2011). What does ground tell us? Monocular visual odometry under planar motion constraint. In *International Conference on Control Automation and Systems* (pp. 1480–1485)

Costante, G., Mancini, M., Valigi, P., & Ciarfuglia, T. A. (2016). Exploring Representation Learning With CNNs for Frame-to-Frame Ego-Motion Estimation. *IEEE Robotics and Automation Letters*, *1*(1), 18–25. https://doi.org/10.1109/lra.2015.2505717

Dalal, N., & Triggs, B. (2005). Histograms of Oriented Gradients for Human Detection. *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, *1*, 886–893. https://doi.org/10.1109/cvpr.2005.177

Dhall, A., Dai, D., & Van Gool, L. (2019, June 1). *Real-time 3D Traffic Cone Detection for Autonomous Driving*. IEEE Xplore. https://doi.org/10.1109/IVS.2019.8814089

Engel, J., Schöps, T., & Cremers, D. (2014). LSD-SLAM: Large-scale direct monocular SLAM. In *European Conference on Computer Vision*

Gosala, N., Bühler, A., Prajapat, M., Ehmke, C., Gupta, M., Sivanesan, R., Gawel, A., Pfeiffer, M., Bürki, M., Sa, I., Dubé, R., & Siegwart, R. (2019). *Redundant Perception and State Estimation for Reliable Autonomous Racing*. IEEE Xplore. https://doi.org/10.1109/ICRA.2019.8794155

He, B., Zhang, S., Yan, T., Zhang, T., Liang, Y., & Zhang, H. (2011). A Novel Combined SLAM Based on RBPF-SLAM and EIF-SLAM for Mobile System Sensing in a Large Scale Environment. *Sensors*, *11*(11), 10197–10219. https://doi.org/10.3390/s111110197

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., & Adam, H. (2017). *MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications*. ArXiv.org. https://arxiv.org/abs/1704.04861

Huang, S., Wang, Z., & Gamini Dissanayake. (2009). *Iterated SLSJF: A Sparse Local Submap Joining Algorithm with Improved Consistency*. https://www.researchgate.net/publication/228373606_Iterated_SLSJF_A_Sparse_Local_Submap_Joining_Algorithm_with_Improved_Consistency

Iagnemma, K., & Buehler, M. (2006). Editorial for Journal of Field Robotics—Special Issue on the DARPA Grand Challenge. *Journal of Field Robotics*, *23*(9), 655–656. https://doi.org/10.1002/rob.20154

Kabzan, J., Valls, M. I., Reijgwart, V. J. F., Hendrikx, H. F. C., Ehmke, C., Prajapat, M., Bühler, A., Gosala, N., Gupta, M., Sivanesan, R., Dhall, A., Chisari, E., Karnchanachari, N., Brits, S., Dangel, M., Sa, I., Dubé, R., Gawel, A., Pfeiffer, M., & Liniger, A. (2020). AMZ Driverless: The full autonomous racing system. *Journal of Field Robotics*. https://doi.org/10.1002/rob.21977

Lee, C.-J., Tseng, T.-H., Huang, B.-J., None Jun-Weihsieh, & Tsai, C.-M. (2015). *Obstacle detection and avoidance via cascade classifier for wheeled mobile robot*. https://doi.org/10.1109/icmlc.2015.7340955

Mathieu Labbé, & Michaud, F. (2017). Long-term online multi-session graph-based SPLAM with memory management. *Autonomous Robots*, *42*(6), 1133–1150. https://doi.org/10.1007/s10514-017-9682-5

Redmon, J., & Farhadi, A. (2016). *YOLO9000: Better, Faster, Stronger*. ArXiv.org. https://arxiv.org/abs/1612.08242

Roy, A., & Todorovic, S. (2016). Monocular Depth Estimation Using Neural Regression Forest. *IEEE Conference on Computer Vision and Pattern Recognition*, *1*(1), 18–25. https://doi.org/10.1109/cvpr.2016.594

Tardos, J. D. (2016). ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo and RGB-D Cameras. *ArXiv*. https://doi.org/10.1109/TRO.2017.2705103

Terzakis, G., & Manolis Lourakis. (2020). A Consistently Fast and Globally Optimal Solution to the Perspective-n-Point Problem. *Lecture Notes in Computer Science*, 478–494. https://doi.org/10.1007/978-3-030-58452-8_28

Thrun, S., Burgard, W., & Fox, D. (2005, January 1). *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. https://www.researchgate.net/publication/224773190_Probabilistic_Robotics_Intelligent_Robotics_and_Autonomous_Agents

Thrun, S., Montemerlo, M., Dahlkamp, H., Stavens, D., Aron, A., Diebel, J., Fong, P., & Gale, J. (2006). Stanley: The robot that won the DARPA Grand Challenge. *Journal of Field Robotics*, *23*(9), 661–692. https://doi.org/10.1002/rob.20147

Varghese, R., & Sambath M. (2024). *YOLOv8: A Novel Object Detection Algorithm with Enhanced Performance and Robustness*. https://doi.org/10.1109/adics58448.2024.10533619

Vödisch, N., Dodel, D., & Schötz, M. (2022). FSOCO: The Formula Student Objects in Context dataset. *SAE International Journal of Connected and Automated Vehicles, 5*(12-05-01-0003).

Xue, Z., & Schwartz, H. (2013). *A comparison of several nonlinear filters for mobile robot pose estimation*. https://doi.org/10.1109/icma.2013.6618066

Zhan, H., Garg, R., Chamara Saroj Weerasekera, Li, K., Agarwal, H., & Reid, I. R. (2018). Unsupervised Learning of Monocular Depth Estimation and Visual Odometry with Deep Feature Reconstruction. *Computer Vision and Pattern Recognition*. https://doi.org/10.1109/cvpr.2018.00043

7. APPENDIX

7.1. Appendix 1: Communication Log with supervisor

Date	Event	Topic of Communication	Outcome
16/08/2024	In-person Meetings	Introduction	General expectation given for this semester
23/08/2024	In-person Meetings	Progress update	Summary of work done so far and demo
30/08/2024	In-person Meetings	Progress update	Presented the latest findings and received valuable feedback to refine the research approach.
06/09/2024	In-person Meetings	Progress update	Presented the latest findings and received valuable feedback to refine the research approach.
20/09/2024	In-person Meetings	Progress update	Presented the latest findings and received valuable feedback to refine the research approach.
27/09/2024		Progress update	Shared a comprehensive update on project milestones and discussed the next phases of the study.
04/10/2024	In-person Meetings	Progress update	Presented the latest findings and received valuable feedback to refine the research approach.
11/10/2024	In-person Meetings	Progress update	Reviewed and refined the research methodology to enhance the robustness of the study.
18/10/2024	In-person Meetings	Progress update	Presented the latest findings and received valuable feedback to refine the research approach.

01/11/2024	In-person Meetings	Progress update	Direction for Report Writing.
08/11/2024	In-person Meetings	Progress update	Review on First Draft.