.25972

DMB

DATA MANAGEMENT
AND
BIOMETRICS

STUDY OF SELF-DRIVING FUNCTIONALITY: FROM SDC TO STEREO VISUAL ODOMETRY-BASED IMPLEMENTATION IN SMALL-SCALE KARTS

Gayathri Dhanapal

MASTER'S ASSIGNMENT

Committee:
dr.ir. L.J. Spreeuwers
ing. G.J. Laanstra
dr.ir. M. Abayazid

October, 2024

2024DMB0009
Data Management and Biometrics
EEMathCS
University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands

UNIVERSITY OF TWENTE. | DIGITAL SOCIETY INSTITUTE

STUDY OF SELF-DRIVING FUNCTIONALITY:
FROM SDC TO STEREO VISUAL ODOMETRY-BASED IMPLEMENTATION IN SMALL-SCALE KARTS

G. (Gayathri) Dhanapal

MSC ASSIGNMENT

Committee:
dr. ir. L.J. Spreeuwers
dr. ir. M. Abayazid
ing. G.J. Laanstra

October, 2024

073RaM2024
Robotics and Mechatronics
Computer Science/Data Management & Biometrics
EEMCS
University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands

Self-Driving Challenge: Implementation of Vision-Only Based Autonomous Driving in Karts

Raj Kumar Ashokan   Gayathri Dhanapal

***Abstract*—This study is based on our participation in the Self Driving Challenge 2023 edition, which was aimed at the study of basic autonomous functionality behaviour in cars. The purely vision-based, classical autonomous driving approaches implemented during the challenge are discussed here. This includes an unsuccessfully attempted monocular visual odometry based approach, in which relative localization was successfully obtained using feature extraction, feature matching, and 2D-2D motion estimation but absolute world scale could not be successfully retrieved. A lane boundary marker detection based approach was then successfully implemented and utilized at the challenge. This approach enabled the provided electric go-kart to autonomously traverse a distance of approximately 1 km. The algorithm processing speed was 30 FPS in real-time. This approach fetched us the runner-up position at the challenge. The observed outcome at the challenge is presented including noted undesirable behaviours. However, the behaviours could not be analyzed or studied owing to the short development stint of the challenge. The shortcomings at the challenge are also identified for both the approaches along with the need for follow-up study. A video of our demonstration of the autonomous driving at the SDC finale event can be found at: https://tinyurl.com/4ycet5de.**

***Index Terms*—self driving challenge, RDW, autonomous cars, monocular visual odometry, relative localization, motion estimation, lane boundary detection, steer estimation**

I. Introduction

Self-driving car technology has been making great strides towards becoming a reality and has been changing day to day lives in terms of road safety [1], ease in mobility, increased travel comfort, reduced emission and pollution levels [2], improved transport inter-connectivity etc. since its advent. Vehicles are increasingly equipped with advanced sensors, vision and control systems, providing them with autonomous capabilities [3] [4]. It has become inevitable to further probe into this field and research into the latest advancements in order to expand the knowledge in smart mobility.

In line with this, being the regulatory organization in approving cars for use on public roads, the Netherlands Vehicle Authority (RDW) has been organizing an annual competition, 'Self Driving Challenge (SDC)', since 2019. The RDW organizes this challenge with the futuristic goal of preparing itself for expanding its knowledge about autonomous vehicles, especially cars, and about the complex choices those vehicles make [5]. SDC being an open challenge for the student teams in The Netherlands, we participated in the SDC 2023 edition as a part of the team from the University of Twente. The main aim of this edition was to build a software stack to autonomously drive a lap as fast as possible at the specified track using an electric go-kart that is provided by the RDW.

Fig. 1: University of Twente team at SDC2023. Image credit: RDW/Self Driving Challenge

Autonomous racing on karts provides a valuable testing field for algorithmic approaches related to autonomous driving. As this field is emerging and relatively new, a direct transfer of autonomous racing software to the autonomous passenger cars has not yet been accomplished [6]. However, increasingly, more autonomous racing challenges are organized using karts, such as the EV Grand Prix Autonomous Challenge [7] and Formula Student Driverless competitions [8]; these challenges induce valuable research that can be scaled up to passenger cars. Therefore, study of basic autonomous functionality behaviour using karts can be a good starting point in the study of the same in cars, aligning with the motive of the SDC [5].

Each participating team at the SDC had limited access to two identical electric go-karts, equipped with the required hardware, and thus the extent of the challenge differed only in the development of the autonomous software and interfacing with the hardware [9]. The whole challenge took place at the TT-Junior track, at Assen in the Netherlands, which is a racing circuit of 1 km length and varying width. Being a single-lane circuit with boundary markers on both sides, it also contained turns, intersections, splits, crossings, curbs and inner loops.

A total of six teams participated in the edition and our team secured the second prize. At the end of the challenge, our developed code was able to provide autonomous functionality to the provided go-kart, but extensive testing could not be performed to evaluate its autonomous behaviour. This extensive testing and evaluation was done as a follow-up study at the University of Twente, by using a rebuilt small-scale kart. This follow-up study is not discussed in this paper. The approaches we used for participating in the challenge and the performance at the challenge is primarily discussed here. Overall, the objective of this Master's thesis work was to implement and study the behaviour of basic autonomous functionality in cars using karts and small-scale karts. Fig. 1 shows the SDC electric go-kart during one of its autonomous manoeuvres demonstrated by the University of Twente team.

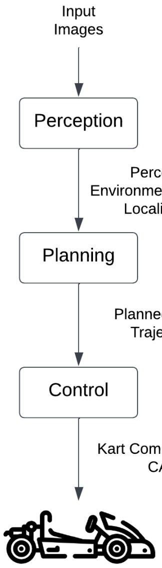


Fig. 2: Basic autonomous driving pipeline

The pipeline for a vehicle to drive itself autonomously from a point A to point B includes three major modules - perception, planning and control as in Fig. 2. Perception is the ability of a vehicle to perceive its surroundings and to know its own position in the environment using the data from its sensors. Planning is the process of generating the best path for the vehicle to traverse and reach its destination, based on the perceived environment taking into account the dynamic capabilities of the vehicle, presence of obstacles etc. Control is the process of converting the intended decisions into actions by sending commands to the actuators to obtain the desired movement [10].

In the SDC kart, cameras were meant to be the primary sensor setup, limiting the scope to visual perception based systems. Obstacle avoidance was not included in the scope and hence is not considered in the entirety of this work. One of the challenging perception tasks for an autonomous vehicle is estimating its current ego-pose or localization [11] [12]. Visual-based localization systems can be broadly based on traditional, learning or hybrid approaches. The state-of-the-art learning-based or hybrid approaches require a lot of data and processing power [6] [13] for considerable performance. However, during the SDC, owing to the limited processing capabilities of the kart, we could not rely on that as the primary method. Nevertheless, two other members of our team were trying to develop an end-to-end steering prediction, which was not progressively fruitful during the competition and, therefore, is not discussed here.

Considering traditional approaches, Visual Odometry (VO), which is a process of estimating the translational and rotational movements of the camera using images, is often used for localization in autonomous vehicles [14] [15]. Therefore, we initially considered monocular visual odometry for localization, as the kart did not include a stereo setup. The involved steps are: feature extraction, feature matching, outlier rejection, and relative pose estimation. Relative ego localization was successfully obtained using this approach; to integrate the absolute scale, an image retrieval-based method was tried to be implemented, but in vain. This involved constructing a database of compressed geo-tagged images and fetching matched images using image retrieval. Thus, the scale factor issue of the monocular camera could not be successfully resolved utilizing any other additional information, resulting in unsuccessful localization. For the planning and control steps in the pipeline, a waypoint follower utilizing a geometric lateral controller was planned to be implemented in case of successful localization using odometry.

Therefore, in the later part of the challenge, we a adopted lane boundary detection-based road following approach, which is another commonly used approach; this was the approach we used for the final race. The steps involved in the perception module here includes feature extraction, detection of lane boundary markers, and lateral localization relative to the lane markings. In the planning module, steer angle is calculated in order to generate the trajectory for lane following, i.e., for the kart to be positioned at the center of the lane. The lateral control module executes the movements, including error handling process, to maintain the planned path.

Traditionally, the study of related works would be carried out at the beginning of the research. However, as this work directly started with the development phase due to the stringent schedule of the competition, the study of related works was done at the beginning of the follow-up study and is not elaborated in this paper. The contributions discussed in this paper focuses on the design-development approach and the implementations carried out to satisfy the requirements of the challenge, which were primarily given by:

- The kart should begin from the start position and run autonomously through the track. The teams would not compete against each other at the same time, but one after the other.
- Each team would get a time-slot of 15 minutes for multiple trials. A trial would be immediately considered disqualified if the kart either goes off-track or even touches either of the lane boundary markers with one of its tyres.
- After disqualification, a new trial would begin again from the start position, only within the provided time-slot.
- The kart should follow the right path at the intersections or crossings.

The team that makes the most metres on the track in the shortest lap time possible, satisfying the above requirements

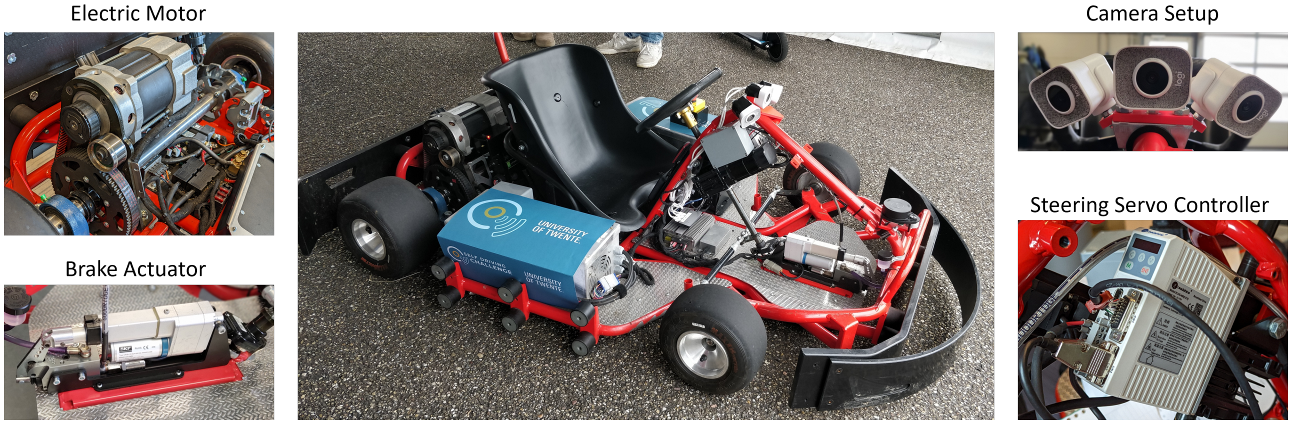


Fig. 3: The SDC electric Go-Kart

would emerge as the winner. More details of the challenge can be found at [9].

The paper is further structured as follows: Section II describes the hardware used. Section III talks about the data collection process. Section IV details the approaches used. Section V discusses the results and the performance at the challenge. Section VI concludes the work with highlights and shortcomings at the challenge, and talks about the follow-up study.

II. Go-Kart

The parts of the electric go-kart relevant to the challenge, as shown in Fig. 3, include the computing unit (a 16GB Intel i5 processor with no GPU), actuators, and sensors, apart from the chassis. The actuation of the go-kart is primarily made via throttle, steering, and braking. The original version of the go-kart consisted of interface modules such as steering wheel and pedals which were human driven. For the sake of autonomous driving, a servo motor (for steering) and a linear actuator(replacing the braking module), which are controlled via ECUs, were integrated in the provided kart. Communication between the ECUs is carried out over a CAN (Controller Area Network) bus, emulating the standard communication system within a conventional car [16]. The kart comes equipped with a 3.5 kW electric motor for longitudinal actuation with maximum speed modes of either 5, 15, 30 or 60km/h. Apart from autonomous control capability, manual control via a wired Xbox controller was also possible.

Existing research works concerning autonomous driving involve fusion of data from sophisticated sensors such as the 3D LiDAR, RADAR, GNSS, IMU, encoders, and cameras in their perception module [17] [18]. In contrast to this, the SDC kart was equipped with a basic sensor suite with three regular USB web-cameras and a 2D planar LiDAR. The cameras were clamped together at a height of 60 cm from the ground in the front part of the kart and had negligible overlap between their field of views. The 2D planar LiDAR, also placed at the front part of the kart, primarily finds its usage in obstacle avoidance task and hence was of little use to us. For safety reasons, an emergency transmitter-receiver pair was interfaced for easy manual intervention.

III. Dataset

Data collection from all the three cameras was performed in parallel, by manually driving the kart throughout the track in lane-centered, lane-edged, and zig-zag manoeuvres. Data was captured in three different speed modes, during different times of the day, and by different drivers. Cameras were calibrated to obtain the intrinsics. In addition to the image recordings, the set of throttle, steer, and brake commands given to the actuators via the CAN network were also recorded in the form of a CSV file. Both the CSV and image data are time synchronized utilizing the concept of multi-threading. This is significant as it facilitates better understanding and correspondence between images and actuator commands, especially during algorithm development and analyses.

IV. Approaches

A. Visual Odometry Based Approach

1) Feature Extraction: The overview of the processes in the initial perception module for monocular VO based approach is as shown in Fig. 4. In this module the center camera images are utilized to obtain the kart's localization. Extraction of features or keypoints in the image is the primary significant step in the process. Features can be extracted using various methods like SURF, Harris corner detector, ORB, FAST etc. ORB (Oriented FAST and Rotated BRIEF) [19] can detect and describe (vectors of size 32) more features, that are invariant to scale, rotation, and small affine changes, quickly than many such feature detector-descriptors [12] [20]. Hence ORB was initially chosen for this step primarily for the sake of efficient computation. An example image with features extracted using ORB is as shown in Fig. 5a. Later in the further stages of the pipeline, when the results of the obtained motion estimation were not as expected (discussed in the Results section), another detector-descriptor known as SIFT (Scale Invariant Feature Transform) [21] was finally used. SIFT is a more accurate method, which detects stable features that are robustly invariant to scale, rotations or small affine changes than ORB [17], and provides descriptors (vectors of size 128). But this comes with a higher

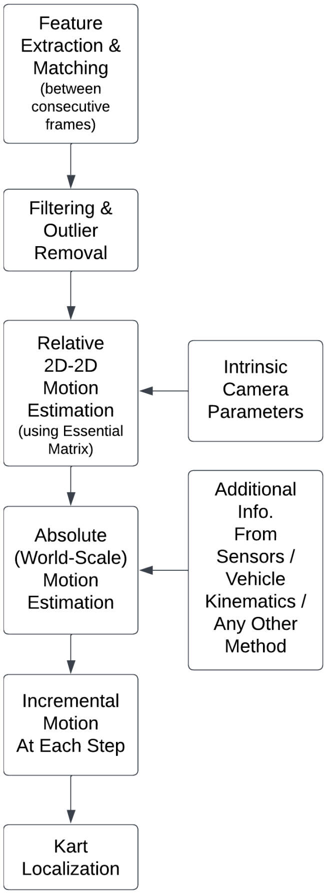
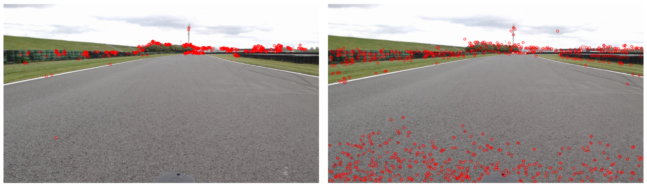

Feature Extraction & Matching (between consecutive frames)
Filtering & Outlier Removal
Relative 2D-2D Motion Estimation (using Essential Matrix)
Intrinsic Camera Parameters
Absolute (World-Scale) Motion Estimation
Additional Info. From Sensors / Vehicle Kinematics / Any Other Method
Incremental Motion At Each Step
Kart Localization

Fig. 4: Processes in the perception module

computational cost [20]. Features extracted using SIFT for the same example image are as shown in Fig. 5b.



(a) Using ORB

(b) Using SIFT

Fig. 5: Sample outputs showing extracted features

2) Feature Matching: Secondly, feature matching between the extracted SIFT features of two consecutive images was performed using a Brute Force(BF) matcher. It considers a descriptor in one image and tries to find a match among all the descriptors in the other, based on the smallest distance. For binary string-based descriptors like ORB, the Hamming distance is considered, whereas for SIFT, the L2 Norm works good [22].

For each feature, two best matches were retrieved and a distance ratio thresholding based on [21] was performed, to filter out unreliable matches. This requires the closest match to be significantly better than the second closest match by checking the ratio of their distances. This ratio threshold was varied from 0.6 to 0.9 to see which threshold was more reliable. Feature matching using BF in two consecutive frames is visualized as shown in Fig. 6. BF-based matcher is highly accurate but requires high computation time. FLANN-based matcher(Fast Library for Approximate Nearest Neighbors) is another technique that finds matches by approximating the nearest neighbours instead of computing them exactly. This is optimized and quicker than BF for larger datasets [22] [23]. Therefore, for the sake of efficiency, this matching technique was also attempted but the quantity of matches obtained were significantly reduced and thus FLANN was not chosen over BF.

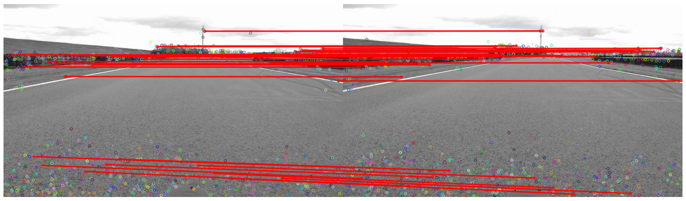


Fig. 6: Matched features between sample consecutive frames. Only a few matches are shown for illustration

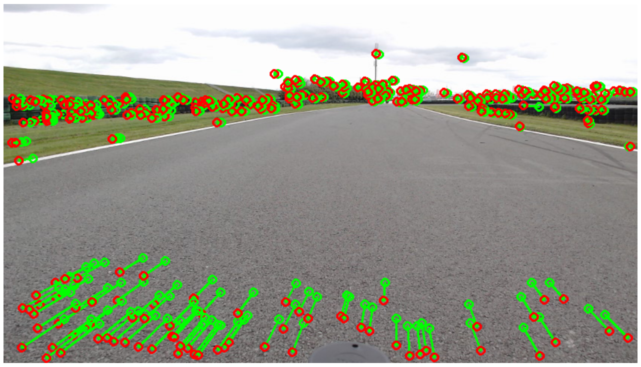


Fig. 7: Estimated motion between sample consecutive frames

Another commonly used technique to improve efficiency is to perform feature tracking instead of matching. Feature tracking using Lucas-Kanade optical flow method was tried. Features generated in a frame are searched for in the consecutive frame using search windows and a pyramid level search approach. Features that are not successfully tracked are dropped; new features are regenerated if the number of retained features drops below a set limit. Again owing to improper results (as discussed later), feature matching using BF was the final chosen approach to find correspondences between consecutive frames.

3) Relative Motion Estimation: From the obtained correspondences or matches between the two consecutive frames and the intrinsic parameters of the camera, 2D-to-2D camera motion was estimated by computing the Essential matrix. This step also incorporates additional outlier removals using RANdom SAmple Consensus (RANSAC), as the filtered matches might still contain outliers. RANSAC iteratively selects random subsets of data and fits a model hypothesis, identifying inliers that align with the model. Assuming known intrinsics, the essential matrix encodes the epipolar

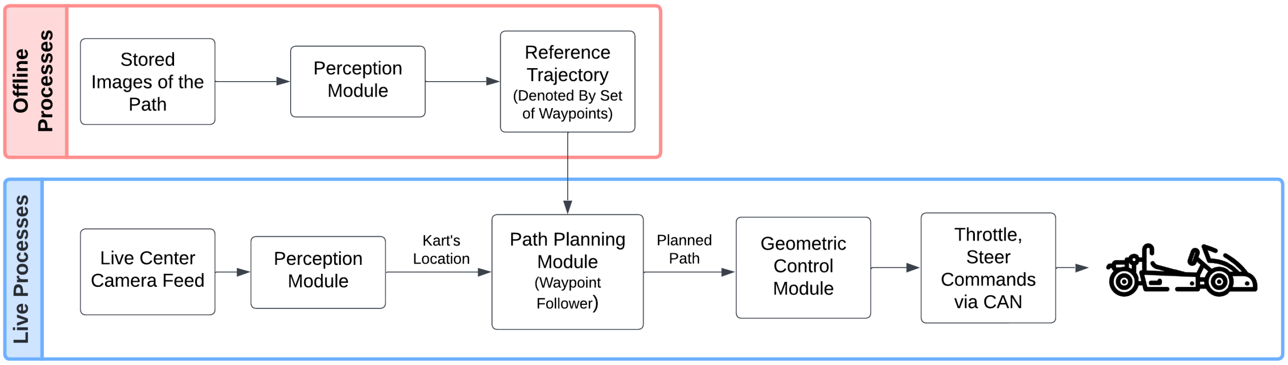

Offline Processes
Stored Images of the Path
Perception Module
Reference Trajectory (Denoted By Set of Waypoints)
Live Processes
Live Center Camera Feed
Perception Module
Kart's Location
Path Planning Module (Waypoint Follower)
Planned Path
Geometric Control Module
Throttle, Steer Commands via CAN

Fig. 8: Overview of the offline and the live processes for the monocular VO-based approach

geometry between two camera frames. This is further decomposed into the relative rotation (R) and translation (T) using Singular Vector Decomposition(SVD) and chirality check, resulting in relative pose between the consecutive frames [24].

The estimated motion between features of two consecutive frames is as depicted in Fig. 7. By accumulating this incremental motion at each step, the trajectory followed by the camera can be generated. Notably, this trajectory is not in absolute scale i.e., real-world units. As the camera is rigidly attached to the kart, the motion of the camera corresponds to the motion of the kart. Resolving this scale issue using any other additional information, trajectory in absolute scale can be generated. By using the collected dataset, a reference trajectory can be generated in this manner;during the live run, a waypoint follower can be used to follow this trajectory by estimating and accumulating the camera poses over time. The overview of the offline and the live processes for the monocular VO based approach with respect to the perception-planning-control pipeline is as shown in Fig. 8.

4) Absolute world-scale estimation: Subsequently, for obtaining the absolute scale, we tried to apply a global localization method partly as in [11], wherein the authors try to obtain the global pose using image retrieval method and a mapping database of geo-tagged images.The purpose of image retrieval is to fetch images similar to the query image from the database. In order to reduce the computational cost of this retrieval, it is necessary to have compact image representations in the database. For creating this offline database, the first essential step is to build a visual vocabulary from the images. A complete set of approximately 14000 reference dataset images was used for this purpose and their SIFT feature descriptors were extracted. Using these descriptors as input data, a k-means classifier was trained to partition the descriptors into 64 clusters. The centres of these 64 clusters are representative feature descriptors and hence form the visual vocabulary.

Secondly, a residual error was calculated between every descriptor of the image assigned to the k-th cluster and the center of that cluster. These residual errors were summed up, giving a total residual error for each cluster. Computing this for all the 64 clusters and stacking up, a 64×128 matrix (as the SIFT descriptor size is 128-dim) was obtained for each image. This matrix was L2 normalized to obtain a Vector of Locally Aggregated Descriptors(VLAD) matrix, which is detailed in [25] and [26]. VLAD matrices of all the images in the considered dataset was generated to form a mapped database. During the live run, in order to fetch the best match for a query image, a BallTree nearest neighbour algorithm trained on this mapped database is used.

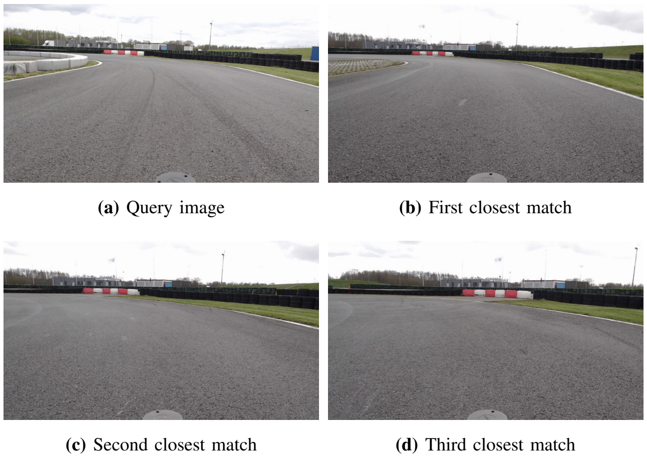

(a) Query image
(b) First closest match
(c) Second closest match
(d) Third closest match

Fig. 9: Sample outputs of global localization module

The image fetched with the lowest distance is chosen as the best match. A sample query image of the Assen track, along with its three closest matches retrieved via this method is as shown in Fig. 9. Visibly, the retrieved matches are properly indicative of the location despite the fact that majority part of the image contained only the road surface, sky, and other less distinctive information.

For tagging the location corresponding to the mapped database images, in [11], a GPS is used in the offline process and a 3D LiDAR is also used to obtain the 3D coordinates of the features. Using this information and a particle filter, the relative world position of the query image, with respect to the retrieved three best matches can be obtained. In our case, we tried to use a constant velocity assumption or to

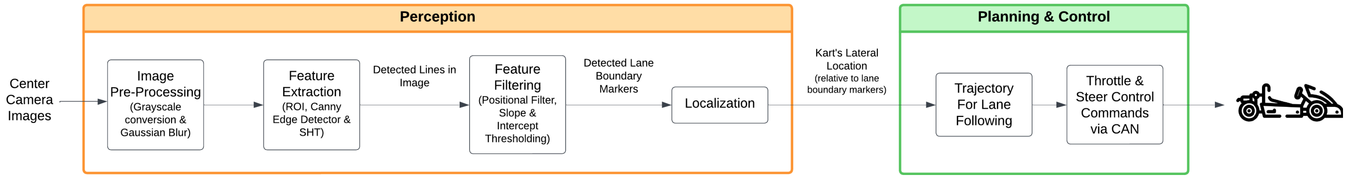
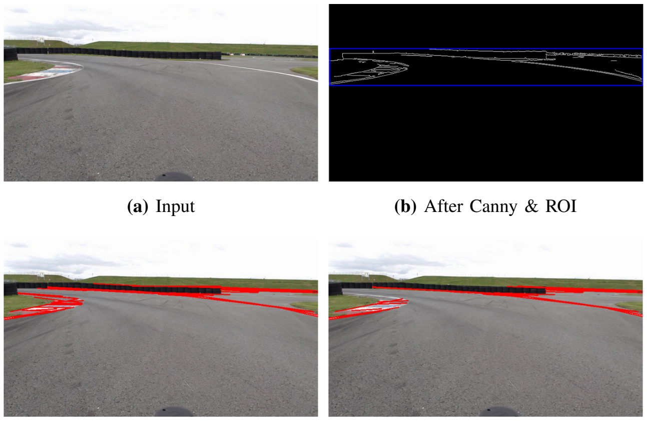

Perception
Planning & Control
Center Camera Images
Image Pre-Processing (Grayscale conversion & Gaussian Blur)
Feature Extraction (ROI, Canny Edge Detector & SHT)
Detected Lines in Image
Feature Filtering (Positional Filter, Slope & Intercept Thresholding)
Detected Lane Boundary Markers
Localization
Kart's Lateral Location (relative to lane boundary markers)
Trajectory For Lane Following
Throttle & Steer Control Commands via CAN

Fig. 10: Steps involved in perception, planning and control modules of lane boundary detection based road following approach



(a) Input **(b)** After Canny & ROI

(c) After SHT **(d)** After Positional Filter

Fig. 11: Sample outputs at intermediate steps

tag the location using any other independent positioning device. However, unfortunately, we could not succeed with this step, owing to cancellation of some of the track days and unforeseen practical hindrances during the test slots.

Furthermore, as only a couple of track days were pending then, which would not be sufficient to do the testing and to implement the following adaptations required for this approach, this approach was dropped; an alternative lane detection-based approach, was attempted to be implemented in the remaining days to the challenge, which is detailed as follows.

B. Lane Boundary Detection Based Approach

This approach builds on the detection of the lane boundaries, using the center camera images as the primary feed. Based on this, the steering angle for lateral control is calculated.

1) Pre-Processing and Feature Extraction: Initially, as the first step in perception, we tried to project the input images using Inverse Perspective Mapping to correct for the perspective distortion [27]. However, in our case, the transformed image did not contain meaningful lane information as the road was very wide. The other option was to perceive using the camera perspective view.

For line detection, grayscale conversion and Gaussian blurring were typically carried out on the input feed to reduce the computation costs and the noise resulting from low camera height [2] [28]. Following this, lane boundary features were detected using Canny edge detector, as it preserves necessary structural information while removing unwanted intense data. In the resulting image, geometric model fitting Standard Hough Transform(SHT) was applied on a chosen Region of Interest(ROI) to characterize the line segments belonging to the road boundaries. The obtained detected lines often included irrelevant lines or false positives as in Fig. 11c; for removing these, a series of line filtering and refinement steps were applied next.

2) Feature Filtering and Localization: Consequently, lines that could be a potential left boundary but present in the right half of the image and vice versa were removed, using a positional filtering that checks the sign of the slope and the image co-ordinates, as in Fig. 11d. This was based on the analysis that unless the kart is off-track, at least one-third portion of the left boundary falls in the left half of the image and similarly for the right boundary. Filtering based on the geometric properties like slope thresholding and intercept thresholding was performed next. Different sample images showing the impact of the above steps are shown in Fig. 12a - Fig. 12d.

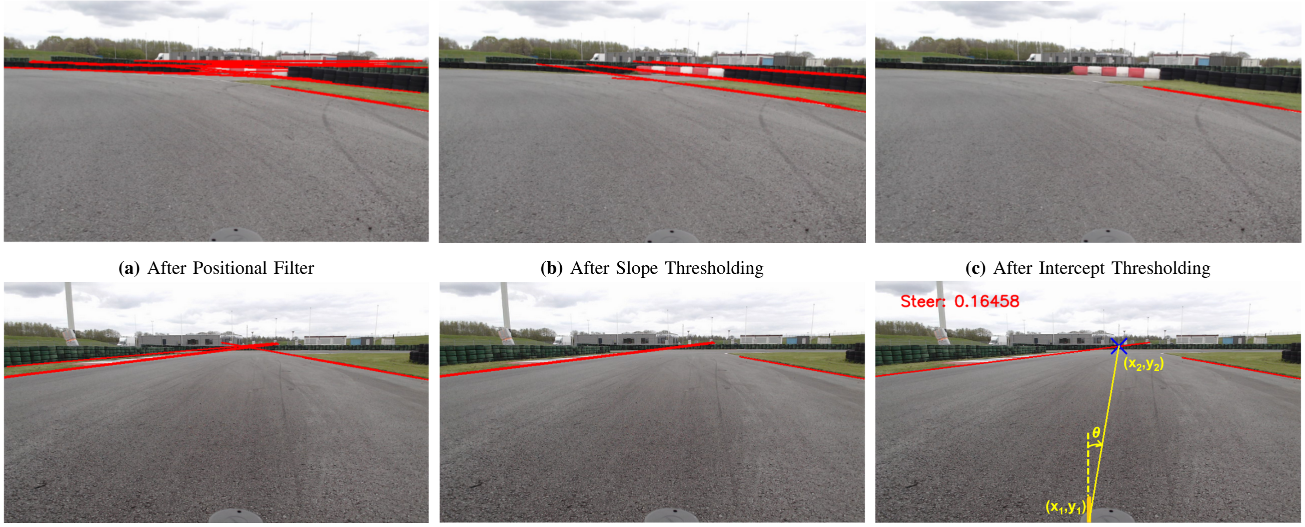
Subsequently, the obtained lines were classified into potential left and right boundaries using the sign of the slope. Owing to the camera's perspective and the wide roads, to avoid false detections of the close-by road edges and baseline of tyre barrier walls, a clustering operation was then carried out by selecting only the lines with a intercept value greater than the mean intercept value on each side as in Fig. 12e. Merging the resulting individual line segments together, their end coordinates were averaged on each side to form the final left and right lane boundaries. If no lines were present during filtering process, next frame was grabbed and processed.

3) Lane Following: Next, in the planning module, steer value was computed for the kart to run in the lane's center. This was calculated using the point of intersection of the detected lane boundaries [13] [28]; this point of intersection denotes the desired heading as shown in Fig. 12f, and is used in the steer(ϕ) computation as given by:

$$\theta = \tan^{-1}(\frac{y_1 - y_2}{x_1 - x_2}) - 90^\circ \tag{1}$$

$$\phi = \frac{\theta}{S_{max}} \begin{cases} > 1 \rightarrow \text{right steer} \\ < 1 \rightarrow \text{left steer} \end{cases} \tag{2}$$

where (x_1, y_1) and (x_2, y_2) are the coordinates of the image base centre point and the point of intersection of the detected lane boundaries respectively; (θ) denotes the deviation of the vehicle's heading, S_{max} denotes the maximum possible steer of the kart.


(a) After Positional Filter

(b) After Slope Thresholding

(c) After Intercept Thresholding

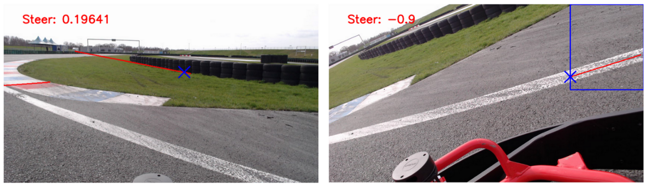
(d) After Intercept Thresholding

(e) After Clustering

(f) After Merge & Steer Calculation

Fig. 12: Sample outputs at intermediate steps (cont.) for two scenarios (First scenario - Row 1; Second scenario - Row 2)

Crucially, in cases where either the left or right boundary was not present or not detected, we considered the extreme column of the image in that particular side as the detected boundary. Mostly such cases were encountered in wide turns, where this manner of handling seemed to work. When both boundaries remain undetected, after a few frames of reusing the previous steer value, the kart comes to a halt. The steering values obtained thus and the fixed throttle values are provided to the actuators in the control module. With the intention to avoid abrupt turns of the kart, we used a slower throttle value during the turns.

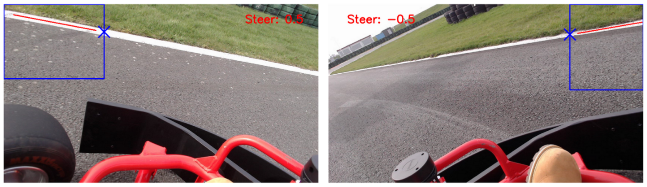


(a) Perceived from center camera

(b) Perceived from right camera

Fig. 13: Need for off-track avoidance

4) Off-Track Avoidance: Nevertheless, due to unforeseen issues, the kart might run off-track. As seen in Fig. 13, despite all the filtering, presence of curbs generates a critical undesired steer at the boundary. Hence we introduce a significant fail-safe 'off-track avoidance' module using the left and right feed. For this, the zig-zag and lane-edged run images were analyzed to identify an optimal Region of Search(RoS), each in the left and right feed, in which boundary lines will appear only when the kart nears it. During live runs, this RoS will be checked for presence of lines and a steer calculation is proportionally derived based on the closeness to the boundary. Line detection was performed using the same steps as the center camera. As these two cameras were downward facing and close to the boundary, strong noiseless edges were detected thereby eliminating the need for major filtering operations.



(a) From left camera

(b) From right camera

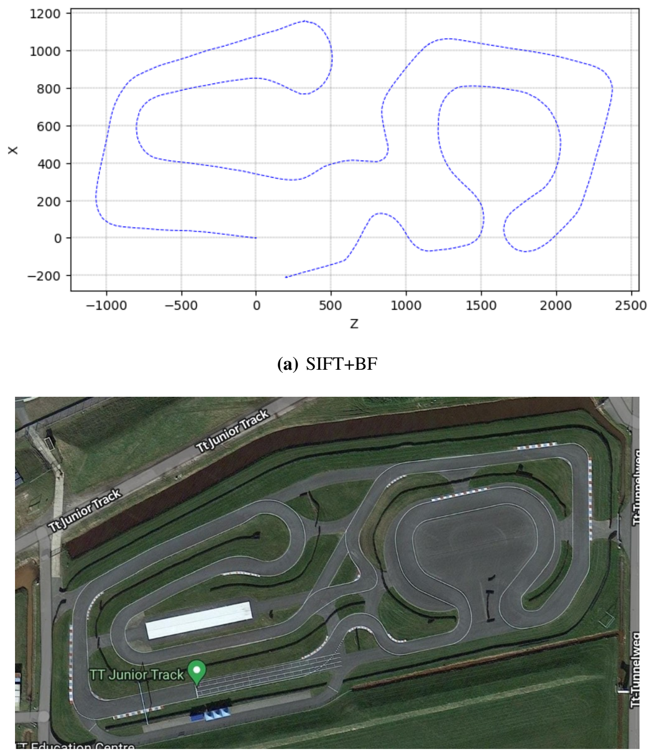
Fig. 14: Sample outputs from off-track avoidance module

The point of intersection of this resulting final left or right lane boundary line with the inner vertical boundary of the RoS is found. As this point moves from the top to the bottom of the vertical boundary, in case of left camera feed, a positive steer value S_{Right} (ranging from 0 to 1) is proportionally devised to move the kart away from the left boundary; similarly a negative steer S_{Left} is devised for the right camera feed, demonstrated clearly as shown in Fig. 14. If no left or right boundary line is detected, then S_{Right} or S_{Left} value is zero respectively. This module simultaneously performs the check in both the cameras and outputs both S_{Left} and S_{Right}.

Combining this module along with the center camera module in a uni-modal sensor fusion, this module's outputs receive the highest priority. In case of non-zero values of either S_{Left} or S_{Right}, it is directly given to the kart to immediately steer it away from the boundary. If both S_{Left} and S_{Right} are zero, the steer from the center camera module is passed to the kart. If both are non-zero, we considered the S_{Right} to be given to the kart, specifically because of the rare false-positive scenarios that we faced due to the tire-markings on the tarmac on one side of the lane.

V. Results & Discussion

The results of the offline processes performed for both the VO and the lane boundary detection-based approaches are



(a) SIFT+BF

(b) Original track from google maps

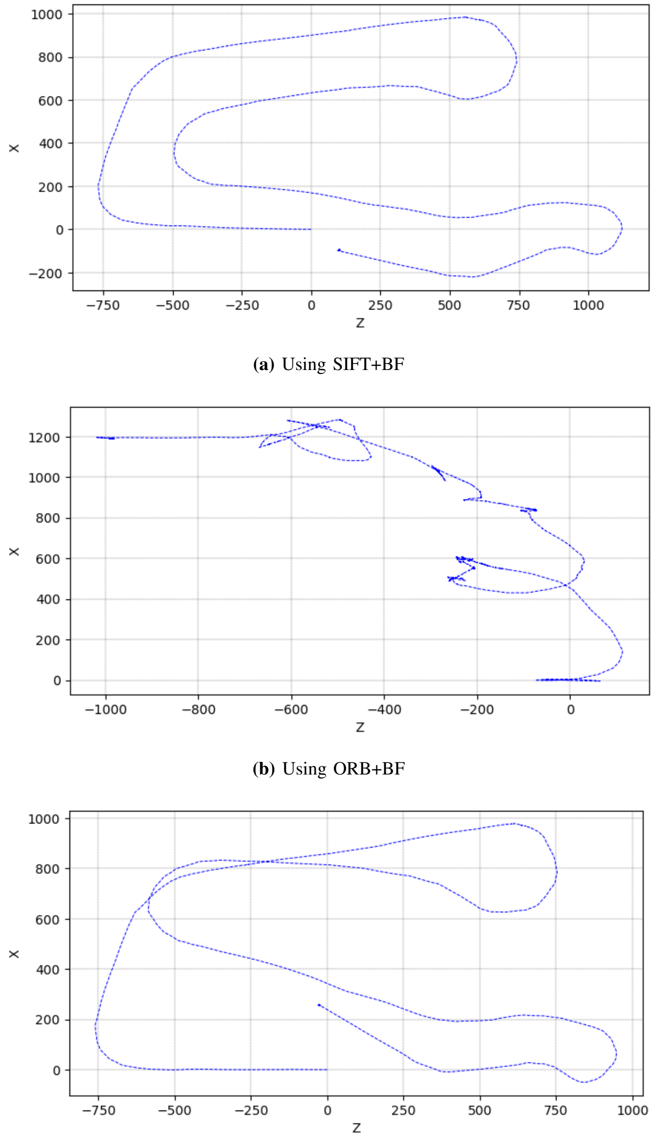
Fig. 15: Comparison of reference trajectory generated using SIFT+BF and the original track

discussed here in addition to the real-time performance of the kart at the challenge.

A. Visual Odometry Based Approach

For this approach, the offline reference trajectory generated utilizing SIFT+BF, for a distance ratio threshold of 0.7, is as shown in Fig. 15a. Though this trajectory is scale ambiguous, the shape retrieved exhibits similarity with the original track as shown in Fig. 15b. However, it failed to achieve loop closure with slight drifts over time; this is consistent with the classic characteristic of VO process and is usually corrected using batch corrective techniques such as bundle adjustment [29]. For the sake of comparison, trajectories for half the track were generated using SIFT+BF, ORB+BF, SIFT+KLT (feature tracking) as shown in Fig. 16. ORB failed to perform, even for varying thresholds. This is presumably because SIFT features are generally detected across the entire image in a scattered manner, whereas, ORB features tend to concentrate more around corners, which is less applicable in our images as ground plane dominates the scene; this also resonates with the feature extraction step shown in Fig. 5a - Fig. 5b. In the shown sample image, the number of initial features extracted using ORB and SIFT were 499 and 903 respectively, and after distance ratio thresholding, the number of obtained features were 178 and 302 respectively.

In contrast to the ORB+BF method, SIFT+KLT method did not completely fail, but rather showed a drift accumulated at some parts of the trajectory. This is mainly due to build up of minor errors that occur while continuously re-estimating the position of the features. This is more significant when tracking is to be done for long sequences as in our case [30].



(a) Using SIFT+BF

(b) Using ORB+BF

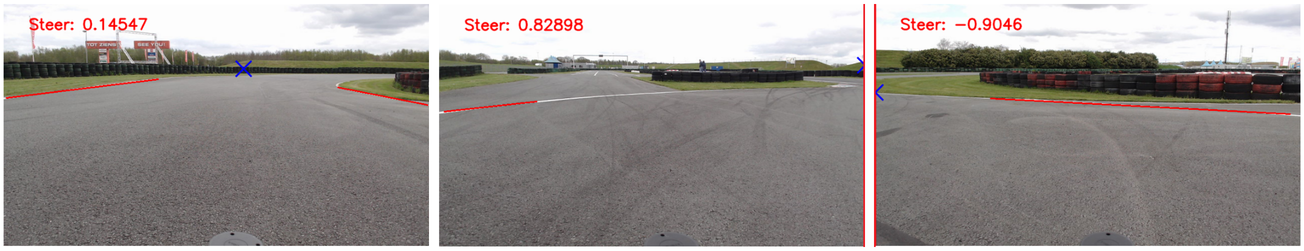
(c) Using SIFT+KLT

Fig. 16: Comparison of reference trajectories generated for first half of the racing track

Thus, SIFT+BF performs better compared to using ORB+BF and SIFT+KLT. While tuning the optimal distance ratio threshold using SIFT, a lesser value of 0.5 or 0.6 filtered out too many features including inliers, resulting in a improper trajectory. On the contrary, a high value of 0.9 retained too many outliers in the matches, which largely remained even after applying RANSAC; this was again indicated by the improper shape of the trajectory obtained.

B. Lane Boundary Detection Based Approach

For lane boundary detection based approach, the final predicted steer results for some of the different scenarios are shown in Fig. 17. Owing to very short development span,



(a) Both lane boundaries visible **(b)** Only left lane boundary visible **(c)** Only right lane boundary visble

Fig. 17: Sample steer outputs for different scenarios

testing was initially carried out fully offline by assuming the possible problematic scenarios at different parts of the track. Possibility of simulation was ruled out due to rendering issues with the provided server and incorrect simulation map configuration. Real-time testing was performed directly at the pre-qualification and finale events. During the pre-qualification, in the initial runs, the kart drove autonomously but crossed weak-edged lane boundaries at two places. Tweaking the line detection parameters, the kart ran autonomously twice in the following runs, traversing the entire length in approximately 15 minutes at an execution speed of 30 FPS. However, it was exhibiting oscillating movements with left/right drags at some places of the track, specifically, at the turns. Notably, the kart also did not follow the path along the center of the lane in some segments of the track. Another significant point is that the off-track avoidance module seemed to work perfectly, with the kart staying on the track. On places where the kart neared the boundaries, a notable push was observed which steered the kart inwards, away from the boundary, right in time. Including such a fail-safe module is often overlooked by many, which was also noteworthy during most other teams' performance at the challenge.

At the finale slot, unfortunately, due to a malfunction, the initial version of the program (before the tweaking of the line detection parameters) had to be run, which made the kart to cross the weak-edged lane boundary at the same place as witnessed during the pre-qualification. As the slot duration was only 15 minutes, this was the only run that the kart made. Later the same day, when the malfunction was rectified and the correct program versions were used, the kart managed to run throughout the track twice autonomously in the first two speed modes. The kart exhibited the same behaviour as exhibited during the full autonomous runs performed at the pre-qualification. Since these runs were made outside the finale slot, our team won the second position.

VI. Conclusion

Overall, the VO based approach was unsuccessful because it could not obtain localization in absolute world scale, as no additional information could be employed; hence this approach could not be utilized for the challenge. On the other hand, the lane boundary detection based approach was implemented and succeeded at the challenge, despite the shorter development span. The kart managed to autonomously run a distance of approximately 1 km with a processing speed of 30 FPS. Still, no quantitative evaluation or analysis could be carried out, except for observing the kart's behaviour during the run, as no data was recorded for the sake of efficiency during the challenge. The reasoning behind the kart's observed oscillating behaviour could not be figured out. The reason for the the kart to follow a path which was deviated from the lane center could not be identified. Whether it was the center camera or the left-right pair that influenced the kart's behaviour could not be ascertained either, highlighting these as the shortcomings at the challenge. Despite the shortcomings, the approach shows potential to work. Similarly, if successfully implemented, VO based approach can be a more generalized one. Thus, the follow-up studies focus on overcoming the shortcomings arising from both the approaches used at the challenge and studying the autonomous behaviour in more detail.

References

[1] I. Kostavelis, E. Boukas, L. Nalpantidis, and A. Gasteratos, "Stereo-based visual odometry for autonomous robot navigation," *International Journal of Advanced Robotic Systems*, vol. 13, 2016. [Online]. Available: https://api.semanticscholar.org/CorpusID:62266727

[2] V. Umamaheswari, S. Amarjyoti, T. Bakshi, and A. Singh, "Steering angle estimation for autonomous vehicle navigation using hough and euclidean transform," in *Proceedings of the IEEE International Conference on Signal Processing, Informatics, Communication and Energy Systems (SPICES)*, 2015, pp. 1–5.

[3] A. rose Harman, "The environmental benefits of self-driving cars," https://greenerideal.com/news/vehicles/driverless-cars-environmental-benefits/, 2024, accessed: 2024-08-22.

[4] "Self-driving vehicles," https://www.government.nl/topics/mobility-public-transport-and-road-safety/self-driving-vehicles, Government of the Netherlands, 2024, accessed: 2024-08-22.

[5] About RDW. RDW. Accessed: 2024-08-22. [Online]. Available: https://www.selfdrivingchallenge.nl/about-rdw

[6] J. Betz, H. Zheng, A. Liniger, U. Rosolia, P. Karle, M. Behl, V. Krovi, and R. Mangharam, "Autonomous vehicles on the edge: A survey on autonomous vehicle racing," *IEEE Open Journal of Intelligent Transportation Systems*, vol. 3, pp. 458–488, 2022.

[7] evGrandPrix. Purdue University. [Online]. Available: https://engineering.purdue.edu/evGrandPrix/autonomous

[8] Formula Student Driverless. SAE International. Accessed: 2024-08-22. [Online]. Available: https://www.fsaeonline.com/

[9] Self Driving Challenge 2023. RDW. Accessed: 2024-08-22. [Online]. Available: https://www.selfdrivingchallenge.nl/previous-editions/edition-2023

[10] S. Pendleton, H. Andersen, X. Du, X. Shen, M. Meghjani, Y. H. Eng, D. Rus, and M. H. Ang, "Perception, planning, control, and coordination for autonomous vehicles," *Machines*, vol. 1,6, 2017. [Online]. Available: https://api.semanticscholar.org/CorpusID:114862052

[11] E. Joa, Y. Sun, and F. Borrelli, “Monocular camera localization for automated vehicles using image retrieval,” 2021, arXiv preprint. [Online]. Available: https://doi.org/10.48550/arXiv.2109.06296

[12] W. Gates, G. Jati, M. Pratama, W. Jatmiko *et al.*, “A modest system of feature-based stereo visual odometry,” in *Proceedings of the IEEE 6th International Workshop on Big Data and Information Security (IWBIS)*, 2021, pp. 47–52.

[13] J. Sujatha *et al.*, “Computer vision based novel steering angle calculation for autonomous vehicles,” in *Proceedings of the IEEE 2nd International Conference on Robotic Computing (IRC)*, 2018, pp. 143–146.

[14] M. Aladem and S. A. Rawashdeh, “Lightweight visual odometry for autonomous mobile robots,” *Sensors*, vol. 18, no. 9, pp. 2837–2851, 2018.

[15] Y. Tanaka, A. Semmyo, Y. Nishida, S. Yasukawa, J. Ahn, and K. Ishii, “Evaluation of underwater vehicle’s self-localization based on visual odometry or sensor odometry,” in *Proceedings of the IEEE 14th Conference on Industrial and Information Systems (ICIIS)*, 2019, pp. 384–389.

[16] Z. Qiao, M. Zhou, T. Agarwal, Z. Zhuang, F. Jahncke, P.-J. Wang, J. Friedman, H. Lai, D. Sahu, T. Nagy *et al.*, “Av4ev: Open-source modular autonomous electric vehicle platform to make mobility research accessible,” 2023, arXiv preprint. [Online]. Available: https://doi.org/10.48550/arXiv.2312.00951

[17] A. Hernandez-Gutierrez, J. I. Nieto, T. A. Vidal-Calleja, and E. Nebot, “Large scale visual odometry using stereo vision,” in *Proceedings of the Australasian Conference on Robotics and Automation (ACRA)*, 2009.

[18] F. Sauerbeck, S. Huch, F. Fent, P. Karle, D. Kulmer, and J. Betz, “Learn to see fast: Lessons learned from autonomous racing on how to develop perception systems,” *IEEE Access*, vol. 11, pp. 44 034–44 050, 2023.

[19] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, “Orb: An efficient alternative to sift or surf,” in *Proceedings of the IEEE International conference on computer vision*, 2011, pp. 2564–2571.

[20] S. A. K. Tareen and Z. Saleem, “A comparative analysis of sift, surf, kaze, akaze, orb, and brisk,” in *Proceedings of the IEEE International conference on computing, mathematics and engineering technologies (iCoMET)*, 2018, pp. 1–10.

[21] D. G. Lowe, “Distinctive image features from scale-invariant keypoints,” *International journal of computer vision*, vol. 60, pp. 91–110, 2004.

[22] Feature Matching. Accessed: 2024-08-22. [Online]. Available: https://docs.opencv.org/4.x/dc/dc3/tutorial_py_matcher.html

[23] Feature extraction and matching. Accessed: 2024-08-22. [Online]. Available: https://pyflowopencv.readthedocs.io/en/latest/tutorial04.html

[24] Camera Calibration and 3D Reconstruction. Accessed: 2024-08-22. [Online]. Available: https://docs.opencv.org/4.x/d9/d0c/group__calib3d.html

[25] R. Arandjelovic and A. Zisserman, “All about vlad,” in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2013, pp. 1578–1585.

[26] H. Jégou, M. Douze, C. Schmid, and P. Pérez, “Aggregating local descriptors into a compact image representation,” in *Proceedings of the IEEE computer society conference on computer vision and pattern recognition*, 2010, pp. 3304–3311.

[27] J. Á. B. Palma, M. N. I. Bonilla, and R. E. Grande, “Lane line detection computer vision system applied to a scale autonomos car: Automodelcar,” in *Proceedings of the IEEE 17th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE)*, 2020, pp. 1–6.

[28] V. S. Dev, V. S. Variyar, and K. Soman, “Steering angle estimation for autonomous vehicle,” in *Proceedings of the IEEE International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, 2017, pp. 871–876.

[29] A. Howard, “Real-time stereo visual odometry for autonomous ground vehicles,” in *Proceedings of the IEEE/RSJ international conference on intelligent robots and systems*, 2008, pp. 3946–3952.

[30] H. Halmaoui and A. Haqiq, “Feature detection and tracking for visual effects: Augmented reality and video stabilization,” in *Artificial Intelligence and Industrial Applications: Smart Operation Management*. Springer, 2021, pp. 291–311.

Study of Self-Driving Functionality: Stereo Visual Odometry-Based Implementation in Small-Scale Karts

Gayathri Dhanapal

***Abstract*—In this paper, we present a basic study of autonomous driving functionality using implementation of pure stereo-vision-based traditional approaches in small-scale karts. This is performed as a follow-up study based on our participation in the Self Driving Challenge (SDC) 2023 edition. A small-scale kart is rebuilt and is used as the test platform. A complete autonomous driving pipeline is explored including localization using stereo-based triangulation, path planning using a waypoint follower and lateral control using a geometric Stanley controller. The suitability of various techniques are checked for this. The influence of the distribution of features, presence of noise or outliers, and handling them pertinent to the challenges that arise from a very small stereo baseline coupled with low camera heights are investigated and discussed. In this regard, analysis is done by placing traffic cones along the track at regular gaps and using these features to check the feasibility of 2D-3D motion estimation. Autonomous driving is achieved in real-time, for distances of 10-20 metres outdoors at a processing speed of 12-13 FPS. The autonomous behaviour exhibited during the runs are presented, and the results are analyzed and discussed. A sample video showcasing the autonomous run performed using the work implemented in this study can be accessed at https://tinyurl.com/bksnf2ku.**

***Index Terms*—self driving challenge, autonomous cars, stereo visual odometry, stereo triangulation, 2D-3D motion, Stanley controller, small-scale outdoor karts, world-scale localization**

I. Introduction

Self-driving car technology has been making great strides towards becoming a reality and has been changing day to day lives in terms of road safety [1], ease in mobility, increased travel comfort, reduced emission and pollution levels [2], improved transport inter-connectivity etc. since its advent. Vehicles are increasingly equipped with advanced sensors, vision and control systems, providing them with autonomous capabilities [3] [4]. It has become inevitable to further probe into this field and research into the latest advancements in order to expand the knowledge in smart mobility.

In line with this, the Netherlands Vehicle Authority (RDW) has been organizing an annual competition, 'Self Driving Challenge (SDC)', since 2019. The RDW organizes this challenge with the futuristic goal of preparing itself for expanding its knowledge about autonomous vehicles, especially cars, and about the complex choices those vehicles make [5]. We participated in the SDC 2023 edition as a part of the team from the University of Twente. The main aim of this edition was to build a software stack to autonomously drive a lap as fast as possible at the specified track using an electric go-kart that is provided by the RDW.

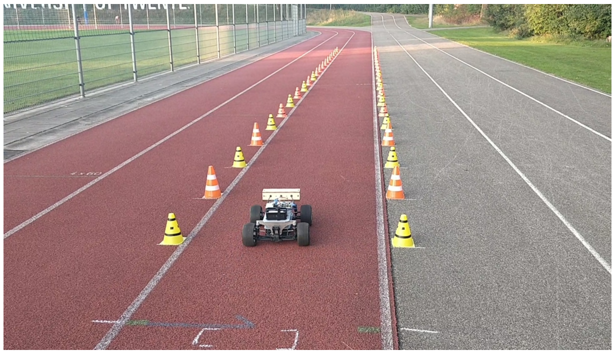


Fig. 1: Small-scale kart during an autonomous manoeuvre at the UTrack

Autonomous racing on karts provides a valuable testing field for algorithmic approaches related to autonomous driving. As this field is emerging and relatively new, a direct transfer of autonomous racing software to the autonomous passenger cars has not yet been accomplished [6]. However, increasingly, more autonomous racing challenges are organized using karts, such as the EV Grand Prix Autonomous Challenge [7] and Formula Student Driverless competitions [8]; these challenges induce valuable research that can be scaled up to passenger cars. Therefore, study of basic autonomous functionality behaviour using karts and small-scale karts can be a good starting point in the study of the same in cars, aligning with the motive of the SDC [5].

A total of six teams participated in the edition and our team secured the second prize. At the end of the challenge, our developed code was able to provide autonomous functionality to the provided go-kart but extensive testing was necessary in order to evaluate its autonomous behaviour. As the challenge was of a short stint, and as the kart was inaccessible after the challenge, in order to study this further, a small-scale kart, mimicking the basic functionalities of the SDC go-kart, was re-built at our niversity and the autonomous functionality was further developed, deployed, and its behaviour was studied. The approaches we used for participating in the challenge is discussed in a separate preliminary paper included in the first part of this Master's thesis work. The follow-up comprehensive study using the small-scale kart was carried out as the second part of this Master's thesis work, which is primarily discussed in this paper. Overall, the objective of this Master's thesis work was to implement and study the behaviour of basic autonomous functionality in cars using karts and small-scale karts. Fig. 1 shows the small-scale kart during one of its autonomous runs.

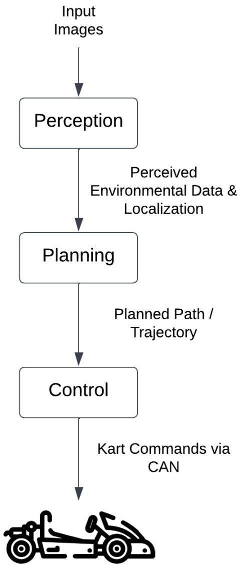

Input Images
Perception
Perceived Environmental Data & Localization
Planning
Planned Path / Trajectory
Control
Kart Commands via CAN

Fig. 2: Basic autonomous driving pipeline

The pipeline for a vehicle to drive itself autonomously from a point A to point B includes three major modules: perception, planning and control as in Fig. 2. Perception is the ability of a vehicle to perceive its surroundings and to know its own position in the environment using the data from its sensors. Planning is the process of generating the best path for the vehicle to traverse and reach its destination, based on the perceived environment taking into account the dynamic capabilities of the vehicle, presence of obstacles etc. Control is the process of converting the intended decisions into actions by sending commands to the actuators to obtain the desired movement [9].

In the SDC kart, cameras were meant to be the primary sensor setup, limiting the scope to visual perception based systems. Obstacle avoidance is also not considered in the scope of this work. One of the challenging perception tasks for an autonomous vehicle is estimating its current ego-pose or localization [10] [11]. Visual-based localization systems can be broadly based on traditional, learning or hybrid approaches. The state-of-the-art learning-based or hybrid approaches require a lot of data and processing power [6] [12] for considerable performance. However, during the SDC, owing to the limited processing capabilities of the kart, we resorted to using only traditional approaches.

Considering traditional approaches, Visual Odometry (VO), which is a process of estimating the translational and rotational movements of the camera using images, is often used for localization in autonomous vehicles [13] [14]. At the SDC, we attempted a monocular visual odometry-based approach due to the lack of a stereo setup. Relative localization was obtained using this approach but integrating the absolute scale could not be successfully performed due to lack of any other additional information. Therefore, in the later part of the challenge, we adopted a lane boundary detection-based road following approach, which is another commonly used approach; we successfully utilized this approach at the SDC to make the kart autonomously drive a distance of 1 km. Although, the exhibited behaviours of the kart could not be studied including significant oscillations of the kart, unintended deviations from the center of the lane etc. More details about the approaches used, the results obtained, and the highlights and shortcomings of our work at the challenge can be found in the preliminary paper included in the first part of this Master's thesis work.

After the challenge, in this follow-up study, the VO-based localization approach was considered to study further and evaluate, the behaviour of the basic autonomous functionality, in line with the primary goal of this work. Addressing the limitations of this approach, it was decided to have a stereo camera setup to deal with lack of additional information when using monocular cameras. To act as the test platform, we re-built a small-scale kart with the same basic capabilities as the RDW kart, slightly adapting the camera setup to be a stereo pair. For small-scale kart build purposes, RC cars are generally adapted with necessary hardware alterations. These can be likened to go-karts, as they achieve high speeds and rapid accelerations for their size. [6].

Thus, stereo-based VO was performed following the basic odometry steps of feature extraction, feature matching, noise filtering, and motion estimation; however, for obtaining the absolute scale, stereo-based triangulation is performed. For minimising the outliers, stereo range-based and epipolar correspondence-based filtering are applied in addition to the RANSAC scheme. Absolute ego-motion estimation is then obtained by minimising the re-projection error for the triangulated features.

Notably, as the cameras are at very low heights in the small-scale kart, presence of distinctive features in the obtained images is very crucial for the success of odometry and this was investigated by placement of traffic cones on the track. Impact of noisy features at such lower camera heights is also an influencing factor especially in real-time autonomous driving and was explored. For planning and control, a waypoint follower using Stanley lateral geometric controller was implemented. Other than in a few works like [15], [16], to our knowledge, approaches based solely on VO for localization in autonomous driving or racing, especially with applicability to small-scale karts are less studied. Even works that do mostly use simulation datasets rather than performing real-time testing. Additionally, most of the works deal with only a part of the pipeline. In our work, we present the entire pipeline along with real-time results. The combination of these factors can be attributed as the novel contributions of this work. Therefore, this research and analysis explores the implementation of stereo VO-based lateral control in small-scale karts, especially relevant to low camera heights, with a focus on the following research questions:

1) What techniques can estimate localization in world units, using only stereo vision, for small-scale karts?
2) What is the impact of presence or lack of distinctive

features on stereo visual odometry for small-scale karts?

3) What effect does the presence of noise or errors has on the performance of the stereo based visual localization and what are the ways to mitigate them?

The paper is further structured as follows: Section II discusses the related works that are relevant to the research questions. Section III describes the re-build of the small-scale kart. Section IV talks about the data collection process and Section V details the methods used in this research. Section VI presents the results obtained and discusses the relevancy to the research questions. Section VII concludes the study along with suggestions for improvements.

II. Related Works

Works concerning classical stereo VO for autonomous driving with a relevancy to the above research questions are discussed here. Most of the works in this category solely deal with the perception module, with a special focus on localization.

Hernandez-Gutierrez et al. [16] focus on a egomotion estimation system in their work, solely using a stereo head camera, and based on feature detection and tracking between consecutive video frames. They present a comparison between two algorithms - one that applies triangulation but states that the localization was seriously affected by the non-isotropic noise in the process and hence a second one that directly works in the disparity space. They also quickly mention the impact of features' distribution but do not discuss any experimentation related to it. For mitigating the effect of noise, a refinement process is performed by utilizing either all the feature points or only a random sample of 7 points or using disparity space homography. They perform testing on video sequences collected using a full-size car in urban environments but do not mention any real-time autonomous driving tests on the vehicle.

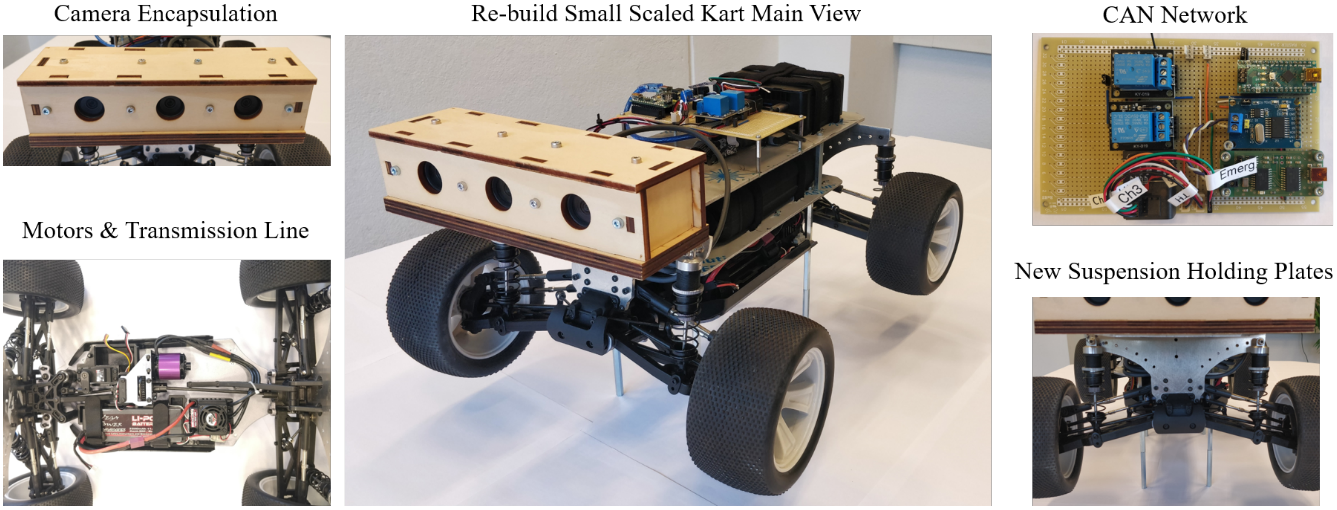
In the work by Agarwal and Konolige [17], a real-time system to localize a mobile robot is presented. This work utilizes a pair of stereo cameras with 12cm baseline and camera height of 0.5m but complements it with inertial sensor suite in cases of failures, and an inexpensive GPS. Relative motion is estimated using feature tracking; and stereo correspondences are triangulated to estimate absolute motion. Disparity space homography is made use of to evaluate the inliers. A Kalman filter fuses the GPS measurements with the global pose to avoid long term drifts. Here, only three points are used here to generate a motion hypothesis; and notably this work discusses the need to ensure that these three points are equally spaced out to avoid bad estimates. Feature locations in the image are divided into equally spaced bins and each feature point is selected from a different bin. Testing is performed on several outdoor terrains in closed loops of over 50-100 metres. Percentage errors in localization and trajectory results are presented in their work. Also, integration with GPS is shown to outperform vehicle odometry or raw VO, for loop closures.

Kitt et al. [18] employ stereo based motion estimation based on trifocal geometry between triples, solely relying on visual inputs. They use feature matching between consecutive frames as feature tracking requires re-initialization procedures. To handle outliers, they utilize a RANSAC based outlier rejection and couple it with a Iterated Sigma Point Kalman Filter to cope with measurement non-linearities. They focus on the distribution of features using a bucketing procedure. They divide the image into non-overlapping rectangles and keep a maximal number of points in every bucket. This is done to have a good distribution of the features along the roll-axis of the vehicle in order to utilize both near and far features. Bucketing ensures having a uniform distribution over the image, guaranteeing that majority of the features lie on the static backgrounds. Owing to this, they claim to observe reduced drift rates, on experiments with simulated data.

Gates et al. [11] build a feature-based simple stereo VO system based on feature matching and linear triangulation. For validation, they propose checking the plausible scenarios depending on the vehicle and the environment that the vehicle is in i.e., given these scenarios and the corresponding conditions, translation between any two time steps cannot exceed a limit and similarly rotation cannot be above a degree threshold. Their overall system is evaluated on a KITTI public odometry dataset but report less accuracy than other such systems; although they report improved speed performance as a plus.

Howard [15] describes a near-pure stereo VO using combination of stereo ranging and consecutive stereo pairs. This work utilizes feature matching for consecutive frames and constructs disparity images for stereo matching. For noisy features or error removal, the author proposes a inlier detection rather than a outlier rejection scheme; the matched points are iteratively analyzed to form cliques, where clique is a subset of mutually consistent matches. A pair of matches is consistent if the distance(world-units) between the two features is identical in the consecutive frames. Practically, at least ten points are needed in the clique for proper ego motion estimation. The author discusses that this approach can easily cope with frames containing 90% outliers but the generalization of the same to handle triangulation is beyond the scope of the work. Also, significance is given to the feature spread in the image. Co-linearity of the features is validated by computing the eigenvalues of the feature distribution and computing the maximal ratio. The algorithm in this work has been tested on many platforms including the DARPA LAGR and Biodynotics BigDog and the same has been discussed in the work. The LAGR vehicle equips two unsynchronized stereo pairs but falls back on wheel encoder and IMU in case of VO failure. One of the limitations highlighted is that VO will work only in environments where stereo works. The work also concludes that the pure VO algorithm is intended to augment some form of proprioceptive sensing rather than be a standalone.

In summary, for motion estimation, most of the approaches utilize either feature matching or tracking for the initial


Camera Encapsulation

Re-build Small Scaled Kart Main View

CAN Network

Motors & Transmission Line

New Suspension Holding Plates

Fig. 3: Rebuilt small-scale kart

steps and then proceed with either triangulation or to operate directly in disparity space. For outlier rejection, RANSAC based scheme is mostly used; for noisy features or error removal, each work focuses on a different validation method, depending on the application and the environment. Works that focus on feature distribution re-iterate the significance of feature spread. However, there are relatively few works that focus on classical, vision-only stereo odometry approaches with an emphasis on real-time testing and that also present a complete pipeline for autonomous driving. The applicability to small-scale karts and lower heights is even more limited. Hence we further investigate the feasibility of these techniques in the following sections, addressing the research questions.

III. Re-build of Small-Scale Kart

We considered a 1:8 scale RC HIMOTO kart for the rebuild framework. For better control, the original brushless DC motor was replaced with a new one to reduce the speed from 80 km/hr to approximately 23 km/hr. Servo motor for steering and electronic speed control for braking is used. A Four-Wheel-Drive (4WD) mechanism drives the kart. Separated power supplies for the motors and the computing unit were used. Intel i5 Processor 16GB RAM was chosen, which is the same as the one used in the SDC kart.

A CAN communication network was used with single ECU control for both the throttle ESC and the steer servo. The kart was made controllable using RC mode (2.4GHz frequencies) or command mode (either autonomous or keyboard commands from a mobile device connected via hotspot) and included an emergency stop mechanism considering safety aspects.

The chassis was remodeled with newly designed suspension plates and increased load capacity, to accommodate two platforms, one for the power supply and the communication circuit and the other for the cameras and the computing unit. Initially left, center, right cameras (Logitech Streamcam) were mounted directly on the top platform in a stereo arrangement with baseline between the left-right cameras as 14cm and that from the center to left/right cameras as 7 cm. However, the less rigidity of the setups caused image instability and inconsistency in calibrated stereo parameters; due to this, an encapsulation setup was designed to house the cameras making the extrinsic calibrations between the cameras as constant. This setup was attached to the platform with the center of the camera at a height of 19 cm from the ground. The final rebuilt kart is as shown in Fig. 3.

IV. Data Collection

The UTrack athletic track at the University of Twente was chosen as the test field. Data was collected using RC or keyboard control commands. Additionally, traffic cones and compact plastic cups resembling traffic cones were placed along the lane boundaries at regular gaps and data was captured. The captured data also included CSV files containing throttle and steer data. The left-right stereo pairs were calibrated to obtain the the extrinsic and the intrinsic parameters. Data was collected during different times of the day at different lighting conditions, for which, the camera settings were adjusted to avoid motion artifacts. Velocity values were measured using manual distance measurements and timers, when needed.

V. Methodologies

In this approach, the perception module involves estimating the pose of the kart in absolute (world) scale, the planning module generates a reference trajectory denoted by waypoints in terms of 3D coordinates and velocities, and the control module implements a controller to utilize the obtained pose and generates actuator commands for the kart to follow the reference trajectory.

1) Pre-Processing and Stereo Rectification: For perception, a stereo camera has the advantage of depth information retrieval over monocular cameras. This helps in the absolute motion estimation in real world distances [11]. We couple this stereo depth information with VO for the motion estimation.

We utilize the left and right camera images from the kart as the stereo pair. Even though the stereo setup is encapsulated, it is not an industrial depth camera which would be more perfectly aligned. Therefore, for real-time stereo algorithms, the initial significant step after grayscale conversion is to perform stereo rectification by warping the left-right images, so that the epipolar lines are aligned with the image rows. These rectified images correspond to images obtained from a virtual pair of perfectly aligned stereo cameras and all the further processing steps are performed using these rectified images [15].

2) Disparity and Depth Estimation: For performing odometry, the 3D co-ordinate values (X,Y,Z) of the features need to be estimated i.e., depth values need to be estimated first. For stereo pairs, disparity and depth computation using stereo matching algorithms are often used for this purpose. Disparity at each pixel in the left image is the difference in X value between the pixel and the corresponding matching pixel in the right image; disparity is hence proportional to the inverse of the depth. The epipolar constraint makes this disparity estimation efficient.

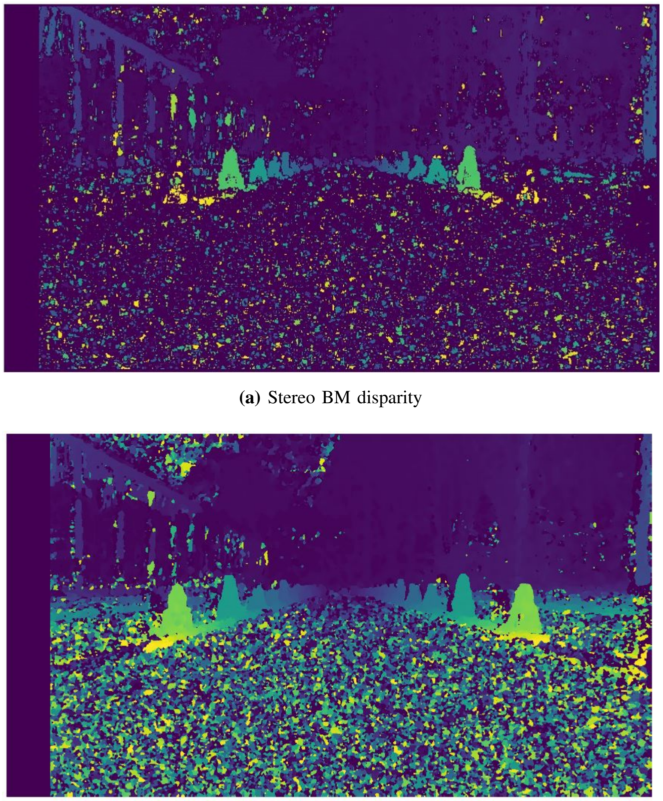
Stereo matching algorithms such as StereoBM (Block Matching) and StereoSGBM (Semi-Global Block Matching) were attempted. These produce a disparity map output based on a scoring method such as SAD (Sum of Absolute Differences) which considers a small window around the pixel of interest and computes the score [15]. StereoBM is computationally less intensive, while stereoSGBM offers higher accuracy by considering global context but at the cost of increased computational complexity [19]. Parameters, including the window size and the disparity range considered for the search, require extensive tuning to find the optimal combination of values suitable for the application. For instance, a larger block size gives a less accurate but smoother map, while a smaller block size provides more details but the chances for the algorithm to find a wrong correspondence is higher.

Depth maps are obtained from disparity maps using the formula:

$$Depth = \frac{(f * B)}{Disparity} \tag{1}$$

where f is the focal length of the camera in pixels and B is the stereo baseline in world units.

However, the obtained disparity and depth maps contained too much noise and less useful information. To improve the disparity maps, before disparity computation, edge-aware Gaussian filtering process was applied on the rectified images to smoothen out noise but this resulted in no significant improvement. An important aspect behind this is that owing to a short baseline, the depth range is small. Computing using Eq. 1, for a baseline of 0.14m and a (rectified) focal length of 743 pixels, the depth is ∼100m only when the disparity is as less as 1 pixel, ∼50 m for 2 pixels and ∼35m for 3.5 pixels. A disparity of 1 pixel cannot be considered reliable due to noise sensitivity and sub-pixel inaccuracies. Therefore, we considered the practical maximum reliable depth range to be around 50 metres. Owing to the lower height of the camera, this is a shorter range given that most of the pixels captured are from road surfaces and not many distinctive features can be obtained from this perspective. Hence, in order to tackle this, traffic cones were placed along the lane boundaries of the track at gaps of 1 metre or 2 metres, as a way of introducing more features. Resolution was also increased from 848×480 to 960×540. After this, disparity was again computed using StereoBM and StereoSGBM, which are as shown in Fig. 4a and Fig. 4b respectively. The corresponding depth maps were also computed.



(a) Stereo BM disparity

(b) Stereo SGBM disparity

Fig. 4: Comparison of disparity map generated using BM and SGBM matchers

As can be seen, the resulting images were still notably noisy overall but retrieved the structure of the cones properly. The estimated depth values corresponding to the cones were closely matching with the ground truth values that were approximated based on the placement arrangement, in terms of meter level precision but the centimeter precision for the ground truth is unknown. Nevertheless, filtering only the valid pixels and avoiding the noisy values in these images, would be a daunting task. A commonly used disparity post processing filter is the Weighted Least Squares filter; this removes noise and small errors, while performing edge-preserving smoothing, using gradient information [19] and a original guided source image. This filter was also tried to

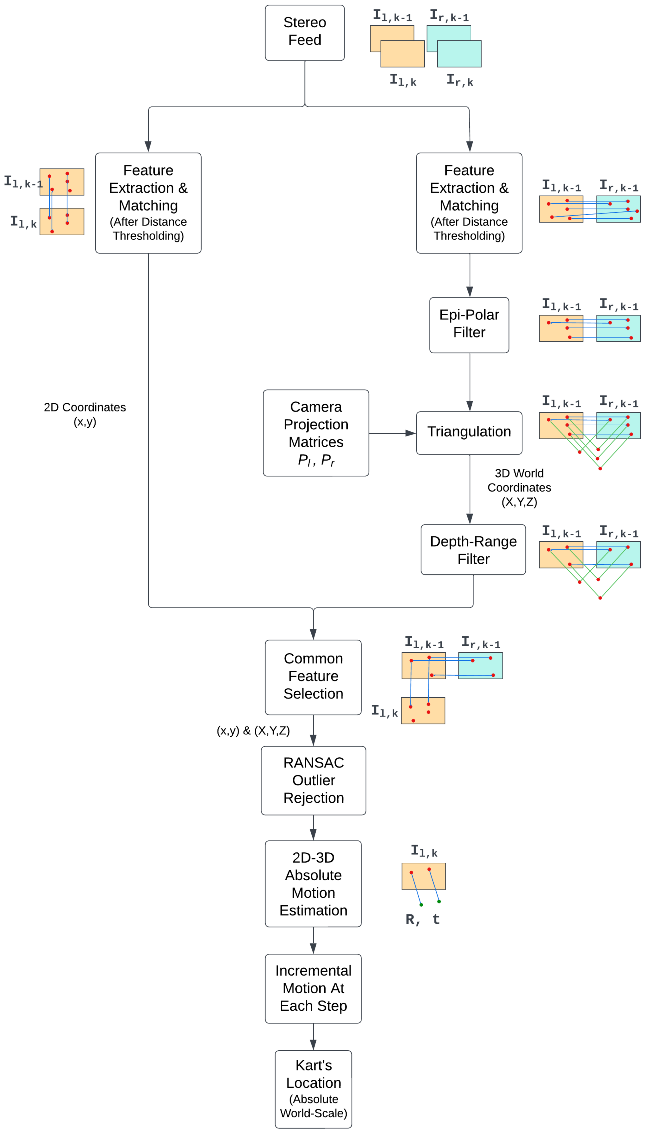
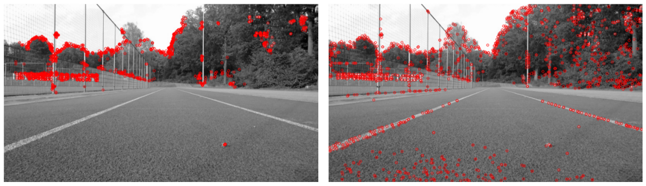

Stereo Feed
I_{l,k-1}
I_{r,k-1}
I_{l,k}
I_{r,k}
Feature Extraction & Matching (After Distance Thresholding)
Feature Extraction & Matching (After Distance Thresholding)
Epi-Polar Filter
Camera Projection Matrices P_l, P_r
Triangulation
3D World Coordinates (X,Y,Z)
Depth-Range Filter
2D Coordinates (x,y)
Common Feature Selection
(x,y) & (X,Y,Z)
RANSAC Outlier Rejection
2D-3D Absolute Motion Estimation
R, t
Incremental Motion At Each Step
Kart's Location (Absolute World-Scale)

Fig. 5: Processes in the perception module

be implemented but in vain.

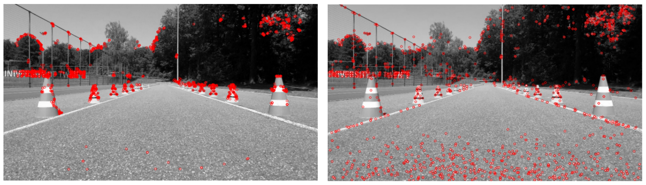
3) Features Extraction and Matching: Alternatively, instead of these block based matchers, another way of finding the 3D coordinates from a stereo pair is to focus on prominent features and perform feature based triangulation. Triangulation determines a point in 3D space given its projections in two or more images. Feature points of the left-right stereo images are utilized for this process only after being validated through a series of filtering steps. A complete overview of the processes in the perception module we implemented utilizing stereo triangulation is as shown in Fig. 5.

Consequently, for feature extraction, SIFT extractor is considered. Based on experimentation, we choose the number of features per image to be 1500, in order to balance performance and real-time computation. As the placement of the cones introduces additional corner features, ORB extractor is also tried [20]. At each step let $I_{l,k}$, $I_{r,k}$, $I_{l,k-1}$, $I_{r,k-1}$ denote the current and previous left-right rectified pair of images



(a) Using ORB **(b)** Using SIFT

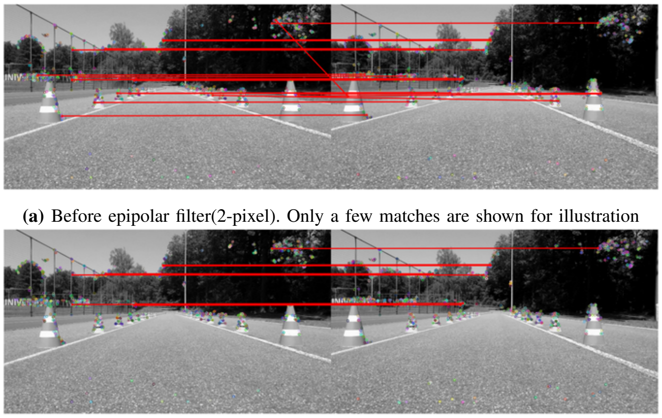
Fig. 6: Sample extracted features in UTrack images in the absence of cones



(a) Using ORB **(b)** Using SIFT

Fig. 7: Sample extracted features in UTrack images with cones placed at 1m gaps

respectively and let (x_i,y_i) denote the 2D coordinates of the i_{th} feature point extracted in the images. Features extracted using SIFT and ORB in sample left images with and without cones is as shown in Fig. 6 and Fig. 7. It can be seen that the ORB also fetches more meaningful features due to the placement of cones.



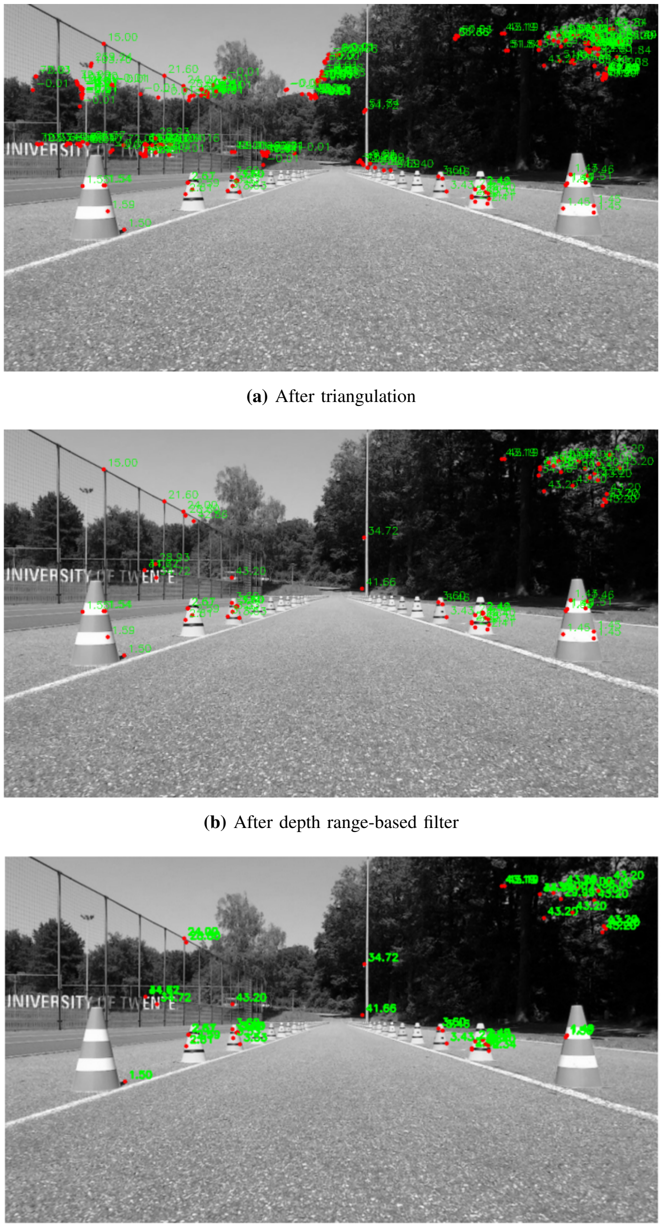
(a) Before epipolar filter(2-pixel). Only a few matches are shown for illustration

(b) After epipolar filter(2-pixel)

Fig. 8: Matched features

At each estimation step, the 3D coordinate values of the feature points in $I_{l,k-1}$ are needed. For this, correspondences between the features of the stereo pair $I_{l,k-1}$ and $I_{r,k-1}$ are obtained by using Brute force feature matching technique, utilizing a distance threshold ratio of 0.7 (chosen via experimentation). In order to further remove outliers among these matches, validation is performed both before and after triangulation.

4) Filtering and Triangulation: For this, utilizing the property of epipolar geometry, we define a pixel-based filter that removes matches that do not adhere to the property. This is performed by checking that the y values of each pair of the matches do not differ by more than 1 pixel or 2 pixels;



(a) After triangulation

(b) After depth range-based filter

(c) Final selected points

Fig. 9: Comparison of the obtained 3D points using ORB at stages of filtering

this difference is considered acceptable because practically, epipolar correspondences would still have slight flaws due to imperfections in rectification. Results of the epipolar filtering process is as shown in Fig. 8. By making use of these filtered matches and the projection matrices of the left and right cameras P_l and P_r, linear triangulation step is performed and 3D coordinates of the feature points in $I_{l,k-1}$ are obtained.

The output of this step is as shown in Fig. 9a. For evaluation, no ground truth 3D values are available for the features due to lack of additional sensors like 3D Lidar. However, the cone features obtained were randomly chosen and evaluated using the approximate ground truth (considered from the cone placement arrangement as mentioned previously).

These triangulated feature points are validated next based on the obtained depth range. Per the computed reliable depth range of 1m-50m, features with depth outside this range are filtered out to increase the reliability of the motion estimation. The output of this step for the sample image is shown in Fig. 9b. Ranges were varied between a maximum value of 30m and 50m to check for changes in performance and 45m was chosen as the suitable range.

5) Motion Estimation in World-Scale: Now, to estimate the motion using odometry, the 2D features in $im_{l,k}$ corresponding to the these triangulated points in $im_{l,k-1}$ are obtained by feature matching with distance ratio threshold of 0.7 i.e., features are matched between the consecutive left frames $im_{l,k}$ and $im_{l,k-1}$, and finally only the triangulated 3D points in $im_{l,k-1}$ which also have a corresponding 2D match in $im_{l,k}$ are chosen. The final chosen triangulated feature points using ORB (range 45m max) are as shown in Fig. 9c. The frame-to-frame motion can be estimated by minimizing the image re-projection error between the obtained 2D-3D correspondences. The translation and rotation can be solved for using the standard Levenberg-Marquardt minimization (least-squares) algorithm [21] [22]. RANSAC [23] based scheme is used to further tolerate outliers in the estimation process.

We set another validation criteria for this step. Theoretically, at least 6 points are needed to generate a unique motion estimation [24] but practically we set at least 8 or 10 points as needed for the estimation.

This gives the estimated motion (X, Y, Z) between each step in terms of absolute world scale. By accumulating this incremental motion at each step, the trajectory followed by the left camera can be generated. As the camera setup is rigidly attached to the kart, camera's motion equates to the kart's motion. We have considered the origin of the left camera coordinate system as the origin of the world coordinate system.

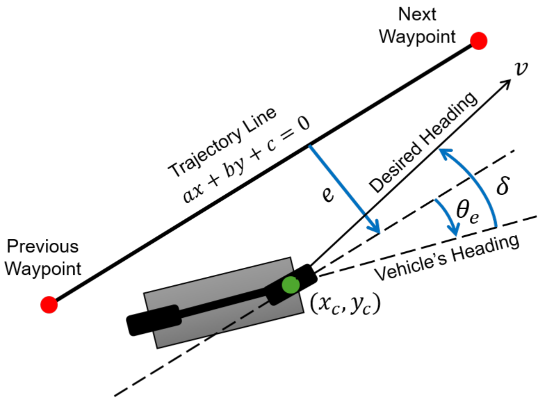


Fig. 10: Stanley controller for a simple kinematic vehicle model. Here δ is the steering angle, θ_e is the heading error, e is the lateral error, v is the vehicle's speed and (x_c, y_c) is reference point taken at the front axle center

6) Path Planning and Control: Performing this localization and trajectory generation process offline on a collected dataset, a list of waypoints that corresponds to the trajectory

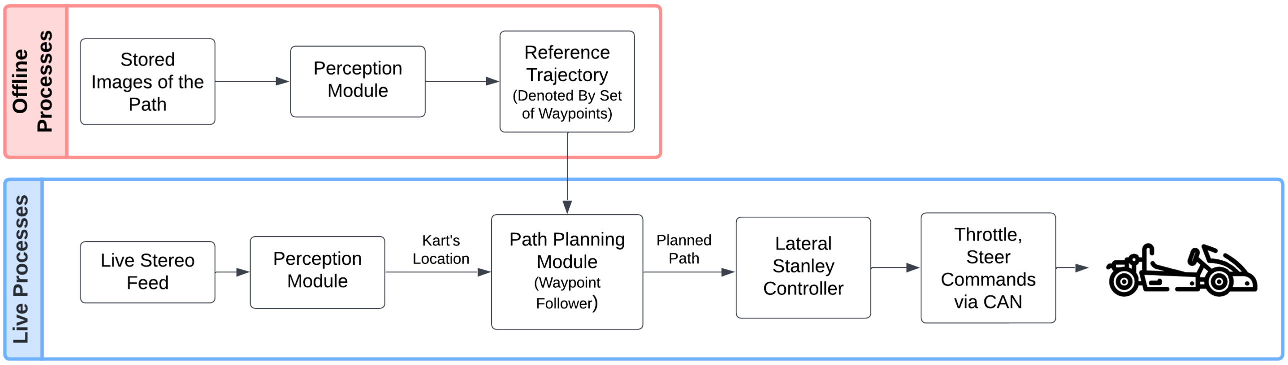

Offline Processes
Stored Images of the Path
Perception Module
Reference Trajectory (Denoted By Set of Waypoints)
Live Processes
Live Stereo Feed
Perception Module
Kart's Location
Path Planning Module (Waypoint Follower)
Planned Path
Lateral Stanley Controller
Throttle, Steer Commands via CAN

Fig. 11: Overview of the offline and the live processes for the stereo VO-based approach

of the driven path can be generated. This waypoints list usually consists of entries of ((X,Y,Z), velocity, orientation) accumulated at each step from source to destination. In our case, we assume constant velocity and for the sake of simplicity coupled with the lack of additional sensors, we construct each waypoint as (X,Y,Z).

During the live run, this reference waypoint list was used as the intended global reference path in the planning module. Local path planning was not necessary in our case as obstacle avoidance is not considered in the scope. At each step in the live run, the actual location of the kart outputted by the perception module is used to find the nearest reference waypoint in the list. A waypoint follower then calculates the lateral and heading error between the actual and the reference locations (waypoints). In the control module, for minimizing this lateral and heading error, a lateral control law stated by a commonly used Stanley controller [25] was chosen. As shown in Fig. 10, this relies on the geometric relationship between the vehicle's heading and the path to provide a steer value in the form of :

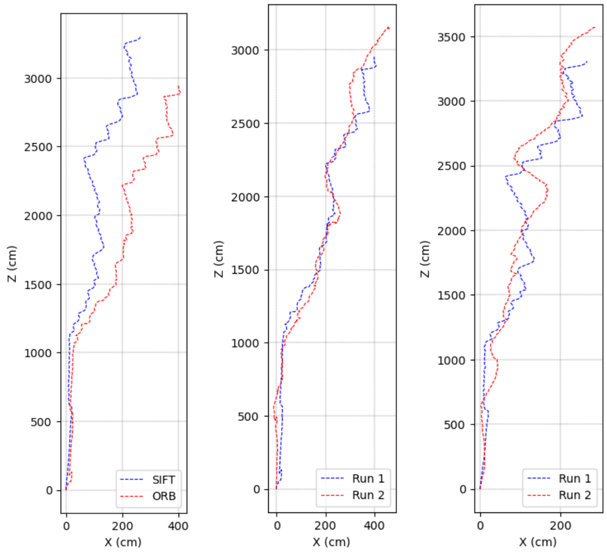
$$\delta = (k_h \cdot \theta_e) + \arctan\left(\frac{k_e \cdot e}{v}\right) \tag{2}$$

where δ is the steering angle, k_h is a gain parameter for the heading error, θ_e is the heading error, k_e is a gain parameter for the lateral error, e is the lateral error, v is the vehicle's speed.

Pure pursuit controller is another often used geometric controller but Stanley is computationally efficient for less complex real-time situations [26]. No longitudinal controller was implemented and instead a constant throttle was provided. The final throttle and steer commands are passed to the actuators via the CAN network. The basic architecture of the implemented overall pipeline is as shown in Fig. 11.

VI. Results and Discussions

The results primarily obtained from each module during the offline and the live processes and the performance of the kart during real-time tests are discussed here, along with the findings and discussion of the research questions.



(a) SIFT vs ORB **(b)** Using ORB runs **(c)** Using SIFT runs

Fig. 12: Generated offline trajectories

1) Offline Processes: Using the methods discussed above, the offline reference trajectories were generated using both ORB and SIFT as shown in Fig. 12. Data collected using manual runs in straight segments of the track lanes, for a distance of 30m, and with cones placed along the track was used for this. Fig. 13 shows the placement of the cones on the track. The obtained trajectories could not be validated quantitatively as exact ground truth could not be determined owing to the absence of an IMU or even a basic velocity sensor in the kart. Therefore, manually measured longitudinal distance and the known shape of the trajectory were considered for comparison. Fig. 12a shows comparison between the trajectories generated using the ORB and SIFT feature detectors. It can be seen that ORB gave a closer longitudinal distance match (approximately 29m) compared to SIFT (approximately 32m), though both the trajectories retained almost the same shape in this case, except for orientation errors at some regions. In the absence of precise ground truth, another way to validate is to test using data from

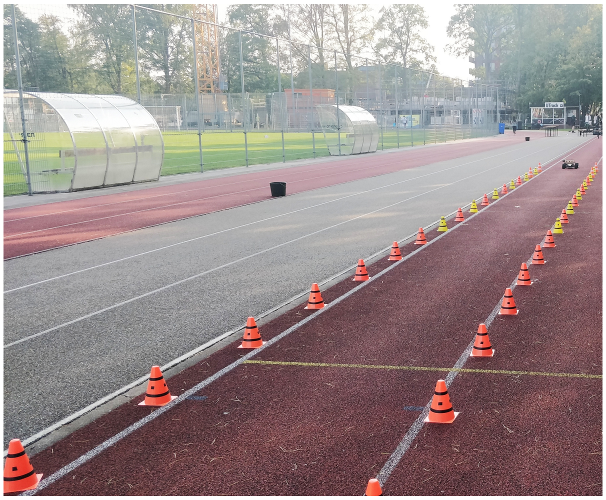


Fig. 13: Traffic cones placement

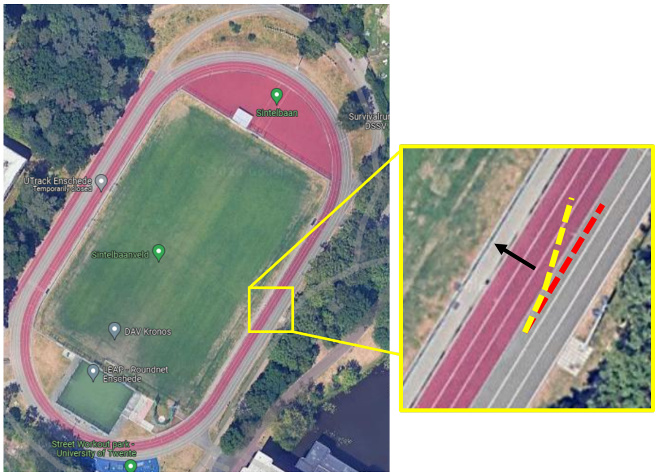


Fig. 14: Zoom-in map showing the slant direction (black arrow), absolute zero steer run (yellow dotted) and expected run (red dotted)

different runs. The offline trajectories generated for different runs using ORB and SIFT can be seen in Fig. 12b and Fig. 12c respectively and ORB showed better repeatability than SIFT.

Notably, the shape of the trajectories obtained using both the methods did not result in a straight path (compared to the manual ground truth). Rather, both resulted in curved trajectories with significant lateral shifts (3-4 meters). Accumulation of drifts over time, especially lateral drifts, is a natural characteristic of VO processes. However, in the manual drives, as the kart was approximately driven throughout in the center of the lane (relative to the lane) and as the total width of the lane itself was only 1.2 m, this drift was perceived as impossible. Despite checking for errors in the localization process and improvements in the algorithm, similar shapes of trajectory were retrieved in numerous runs at different places of the track and for different distances. On probing, we determined that this is attributed to combination of the following reasons:

1) The kart is originally a hobby-purpose RC car and has been rebuilt. It suffers from significant wheel misalignment of the front wheels, owing to which it has a natural drag towards left side.
2) The Utrack is an athletic track and the lanes of the track are slightly slanted inwards at varying degrees. This was informally tested by placing spirit levels and fluid-filled containers on the track.
3) Naturally occurring accumulation of drifts using the VO processes without any adjustment techniques.

This reasoning was confirmed by driving the kart in the track using absolute zero steer and giving only throttle values. Though placed at approximately zero degree start-orientation relative to the lane, instead of running straight in the lane, the kart ran significantly left. This can be seen as in Fig. 14, where the red dotted line denotes the expected run of the kart and the yellow dotted line denotes the actual run of the kart. However, while driving manually, the driver controls the kart by giving right steer then and there, to keep the kart in the center of the lane. These right steers cause the odometry algorithm to believe that the kart is taking a turn as the algorithm checks for the relative motion of the features only between the consecutive frames. When accumulated, this causes the perceived lateral drift shown in the generated trajectories. This left drag and slant was tried to be compensated by giving a constant right steer, but this value could not be pre-defined as the drag was varying each time.

Nevertheless, this does not denote any failures with the odometry process and the localization output of the perception module itself. Additionally, considering only the offline processes of the perception module, ORB was able to process the frames at around 15 FPS whereas, using SIFT, the FPS was reduced to approximately 8. Hence, ORB was chosen to be used for real-time tests.

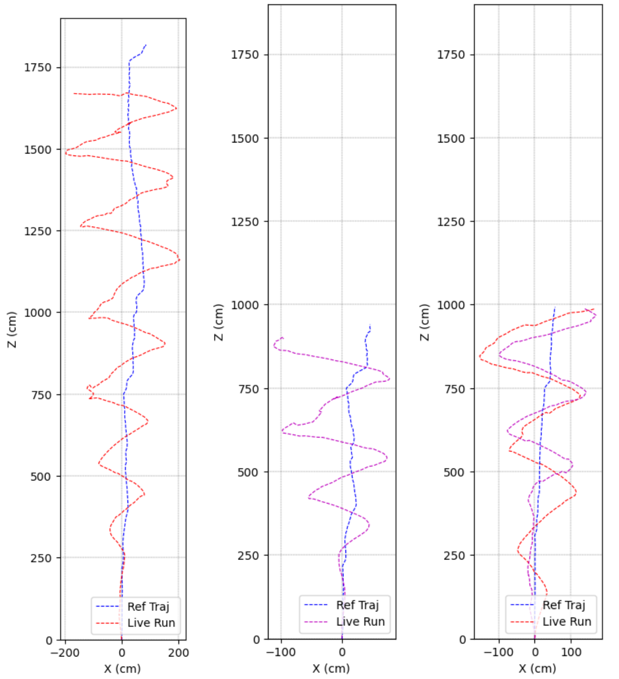
2) Real-time runs: Using the offline trajectories generated as above (using ORB) as reference trajectories, real-time autonomous live runs were tested. For the real-time runs to work, all the three modules in the pipeline should work successfully. The perception module choices were retained the same as that of the offline trajectory generation process, including feature detection and matching, triangulation and motion estimation, filtering steps, and the constant parameters. Traffic cones were placed on both sides along the track at gaps of 1 meter. For the planning and control modules, velocity of the kart is an important input parameter as can be seen in Eq. 2 of the Stanley controller. Incorrect velocity values lead to erroneous steer commands. Therefore, manual measurements of the kart's velocity had to be taken frequently during the experiments, as even small changes in battery levels resulted in noticeable drops in speed. The exposure of the cameras was manually set before each run depending on the lighting conditions at the moment.

During the tests, owing to lack of ground truth data, each step of the live run could not be precisely validated. However, at each step, the obtained ego-localization was checked to see if the values were sensible according to the visual evaluation of the kart's movement. In the initial tests, the processing speed of the algorithm seemed too low for the kart to run. Hence, the efficiency of the processes were partly improved by enhancements in the implementation. In the proceeding runs, the processing speed was slightly improved and the kart's velocity was set to the minimum possible (approximately 62 cm/s) in accordance with the processing speed. However, the kart ran a distance of only 2 m, and then curved towards the left or right, and bombarded with the cones. On analyzing the data, the localization estimates were good, implying that the perception processes were working fine. The closest waypoint estimates fetched by the planning module were checked for correctness too.

However, the controller module did not seem to produce the required steer commands. The parameters k_e was set as 0.5 and k_h was set as 1 during these runs. Tuning these parameters according to the vehicle and the environment was extensively carried out. Despite this, owing to frequent heading angle changes in the considered reference trajectory, the heading error changed at a high frequency during the live run; as a result, the controller produced frequent and sudden steer oscillations. To reduce this impact, the weightage of the heading error (k_h) had to be lowered. On the contrary, the weightage of the cross-track error had to be increased, due to the very small lane width (120 cm). As the kart was 40 cm wide, it had less than 40 cm movable lateral space on each side before it would hit the cones placed on the lane boundaries. This space was practically reduced to approximately 15-20 cm on each side when the kart diagonally nears the cones, because the Field of View of the cameras used enables information capture only from 50 cm ahead. Due to this, as the kart nears the cones, there will be no information except unidentifiable blur or plain cone surfaces; this causes the odometry to fail and consequently the motion estimation and the control fails. Ultimately, only when the k_h was set as 0.5 and the k_e was set as 1.2, the kart was able to carry out autonomous manoeuvres.

3) Performance of the kart: Runs were performed for distances upto 10 m and 20 m. The same set of tuned parameters enabled the kart to traverse these distances autonomously. The trajectories generated from the data collected during these live runs along with the reference trajectory are as shown in Fig. 15a and Fig. 15b.

It can be seen that the kart exhibited oscillating behaviour although it successfully traversed the distance. The behaviour was also visible during the runs on the track. On analyzing, one of the primary reasons is that though the controller does not produce a left steer the kart moves to the left due to its natural left drag as discussed earlier. The controller module does not incorporate this information about the mechanics and the consequent further left drag, and hence produces a small right steer to compensate. However as the left drag increases, the controller produces a higher or continuous right steer to compensate, which explains the drift to the right. The pattern repeats again resulting in oscillating behaviour. This is also caused by the impact of the oscillations in the heading errors despite reducing it by a weightage factor of 0.5. As seen in Fig. 15a, the oscillations begin in the live run when the reference trajectory shows significant changes in the heading direction. The processing power and in turn the execution speed also influences the reaction time of the controller. The entire pipeline was capable of live execution at ∼ 12-13 FPS.



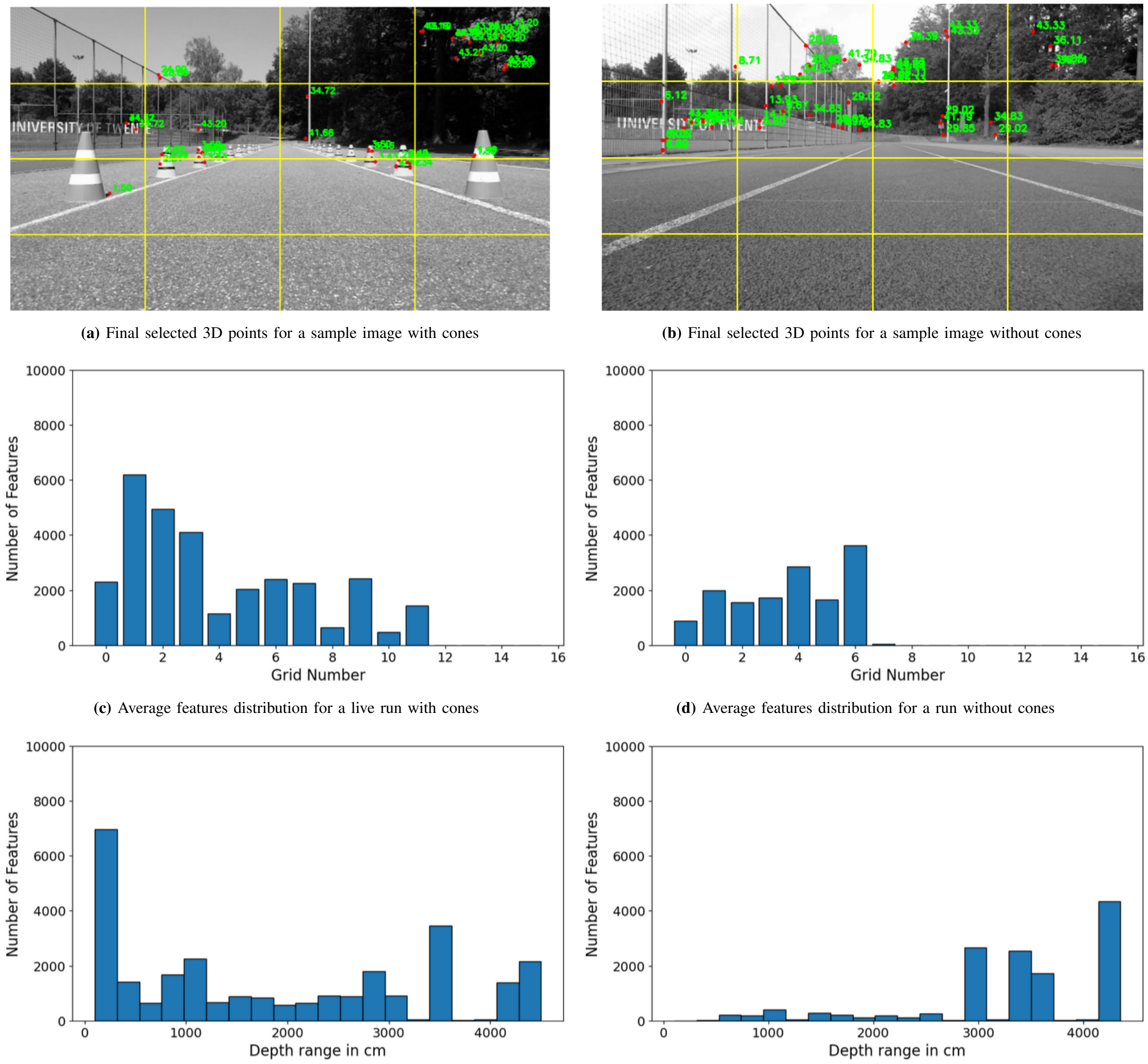
(a) For a straight segment of ∼20 m distance **(b)** For a straight segment of ∼10 m distance **(c)** For a different segment of distance ∼ 10 m

Fig. 15: Live runs vs reference trajectories

Nevertheless, despite the oscillations and a stringent lateral space on each side, the kart was able to achieve the autonomous runs. Repeatability was tested by performing the live runs at a different straight segment of the track and on different days. The trajectories generated corresponding to two different runs in this segment compared until a distance of 10 m are plotted as shown in Fig. 15c. along with the reference trajectory. The longitudinal error for the run considered in Fig. 15a was found to be ∼1.9 m. The average longitudinal error for all the autonomous runs performed was ∼ 1.5 m; it is important to note again that all the measurements are completely manual.

The lateral error was considered to be the ∼40 cm lateral space as mentioned above. However, in the plotted trajectories, the lateral drifts are calculated to be higher which is not possible as the kart did not go out of the lane of a total width of 120 cm. The shown lateral error is again perceived due to the accumulation of drift because of frequent changes in the heading.

The tests failed at around 10 meters in case of notable changes in the natural lighting conditions; this is because as the lighting varied, the exposure of the cameras had to be adjusted and more exposure introduces motion artifacts and affects the frame rates. A varied frame rate alters



(a) Final selected 3D points for a sample image with cones

(b) Final selected 3D points for a sample image without cones

(c) Average features distribution for a live run with cones

(d) Average features distribution for a run without cones

(e) Average histogram distribution for a live run with cones

(f) Average histogram distribution for a run without cones

Fig. 16: Comparison of the obtained 3D points using ORB at stages of filtering

the available information and impacts the feature matching process. Another trouble faced was the vertical vibrations of the kart as it is a simple, small-scale, non-industrial grade vehicle, and not equipped with robust mechanics. This also highly influences the process as visual odomtery is dependent on the change in information between the frames.

4) RQ1: Addressing the primary research question, the techniques that work for implementing the autonomous driving using only vision-based odometry can be inferred from the elaborate discussion of the results presented above. The heading error oscillations and its effect can be mitigated by the use of a look-ahead distance using the fetched reference waypoints while computing the heading error. This takes into account the heading of the global trajectory that is further ahead and reduces the impact of the local oscillations. A Kalman filter can also be used to smoothen the heading error oscillations.

Failures were reported in the localization process in a few frames. A failure is reported when there are no valid points available to estimate the motion after all the filtering processes. For such cases, constant velocity assumption can be used to compute the motion. However, we could not attempt it in our implementation because the velocity of our kart changes non-linearly with decline in battery levels.

The reaction time of the controller can also be improved by improving the performance of the algorithm. Feature extraction is one of the steps that occupy a lot of the processing power. This step was performed sequentially for the left and right camera images. Performing this in parallel using multi-threading can further fasten the performance. Similarly the implementation of the step where the common features are selected from the left image of the stereo pair and the consecutive left frame was computationally expensive and can be made efficient. These improvements can possibly increase the execution speed to 15 FPS.

5) RQ2: For analyzing the impact of the presence of features in the obtained runs, we have utilized two visualization forms: dividing the image into grids and mapping the number of features per grid in a bar graph, and considering the chosen depth filter range and checking the distribution of the features across this range via a histogram. For this analysis, we consider the run corresponding to the trajectory in Fig. 15a. Fig. 16a denotes the final 3D points obtained using ORB, in a sample image segmented into grids. Fig. 16c denotes the distribution of these 3D points across the grids computed for the data of the whole run, considering the top left grid as start and moving right; whereas Fig. 16e shows the chosen depth range (1m-45m) and the spread of the 3D points across these continuous ranges as computed for the data of the whole run. These histogram distributions were computed for all the autonomous runs performed and similar patterns were observed for the different runs. Whereas, the Fig. 16b shows a sample image from a manual run in the absence of cones and Fig. 16d shows the average feature distribution across grids for data collected from the images of the manual run. Fig. 16f shows the average histogram distribution for the same run. Similar patterns were observed in other runs carried out in the absence of cones as well. Comparing these average histogram distributions, it can be seen that there is a stark difference in the form of peaks in the near-depth ranges around 2-3 metres in the autonomous runs that succeeded. However, in the runs without cones, the distribution of features in the near-depth ranges is too scarce. This explains the reason why the runs succeeded only in the presence of cones and failed drastically in their absence. The availability of distinctive features in the near-depth ranges has a high impact on the motion estimation process, especially in the translation estimation. The average feature distributions of the runs with and without cones also show that presence of distinctive features distributed in the grids corresponding to the near-depth ranges is crucial for the success of the runs. The far-depth ranges should also contain at-least a part of the distribution, though not high peaks as in the near-depth range. This assists in the rotation computation in the motion estimation process.

6) RQ3: Considering the impact of the outlier or noise filtering steps on the overall process, the epipolar filter was found to have a greater impact for better motion estimation. Before the introduction of this filter in the process, a significant number of features were having erroneously estimated 3D values; this was removed by the introduction of this filter. Having the pixel-difference as '2' was ideal for this filter; as 1-pixel difference retained only too less features. The next impactful process was the depth range filter. Consideration of the optimal range that the cameras can handle helped in removing outliers and unreliable features belonging to the far ranges, as can be seen in Fig. 9b. Finally removing out the features that were not commonly present in the left consecutive frame pairs, was necessitated by the process. The combination of these filters drastically reduced the number of final points obtained, especially the unreliable features belonging to the noisy road textures or the distant trees, and increased the selection of valid features as that of the cones. Without this, autonomous runs could not have been achieved.

Overall, combining the appropriate filtering steps along with the discussed suitable techniques, is highly essential to achieve autonomous driving using "pure" stereo visual odometry on small-scale karts under constrained operating conditions. No other related work demonstrates classical approaches for real-time autonomous driving in small-scale karts using only a single-stereo pair and a camera setup height as low as 19cm from the ground. All works explore camera-based driving augmented with any other basic or advanced sensors or using learning-based approaches. Only the works such as in [15] explore nearly pure stereo autonomous driving which was applied in the DARPA LAGR and the BigDog applications. But in the work, two stereo pairs are coupled to explore the autonomous driving. Nevertheless, stereo visual odometry coupled with an IMU is more powerful and fail-safe. Considering that our implemented techniques can achieve camera-only-based autonomous driving under stringent operating conditions as discussed, it has the potential to perform well in improved hardware and better operating domain conditions.

Relevant to the SDC, if the SDC kart were equipped with a stereo pair instead of multi-cameras with no overlap, this approach would have worked better than with the small-scale kart, owing to the comparative high camera placements and more stability of the go-kart. This has the potential to explore more techniques in the study of autonomous driving behaviour when coupled with a basic IMU. The organizers of the SDC can take this study into account as the findings align with the motive of the challenge.

VII. Conclusion

As a part of the follow-up study addressing the shortcomings of the monocular VO approach after the SDC, real-time autonomous driving using a single stereo-pair placed at a very low height on a small-scale kart was achieved in this work. Incorporating the basic functionalities of the SDC kart, this small-scale kart was re-built successfully as the intended test platform to implement and study the autonomous behaviour. The study was focused on the investigation of the techniques that can be relevant to small-scale karts, with analyses about the impact of the presence and distribution of distinctive features and the ways to mitigate outliers throughout the processes. In accordance with this, a complete pipeline including ego-localization in the perception module, waypoint follower in the planning module, and a geometric Stanley controller in the control module is presented. In order to study the impact of features, traffic cones were placed along the lanes at regular gaps and the autonomous runs were made using this data. Ego-localization was achieved using feature extraction, matching, and filtering coupled with stereo-based triangulation. Depth and disparity-based stereo block matching techniques were also initially attempted instead of triangulation process, but these techniques failed due to the noisy environments arising from the low camera heights. A series of filtering steps was incorporated to include only valid points for the 3D points estimation. 2D-3D correspondences were used to estimate the kart's motion. Autonomous runs were achieved for distances of 10-20 meters with an average longitudinal error of 1.5 meters and a execution speed of 12-13 FPS. No other sensors were used in the runs. The exhibited oscillating behaviour during the autonomous runs were studied and suggestions for improvements were proposed. Possible failure cases of the VO approach were also discussed with techniques to handle the failures. Future works can focus on an advanced implementation of the control module, additionally taking into consideration the mechanics and dynamics of the kart. VO is a more generalized process for vision-based autonomous driving and can work in scenarios where even IMUs fail. However, VO is intended to augment other sensors; in case of VO failures, fusing the data from other sensors such as a basic IMU, if available, can make the algorithm more reliable and this would be the most suitable approach for real-time driving.

References

[1] I. Kostavelis, E. Boukas, L. Nalpantidis, and A. Gasteratos, "Stereo-based visual odometry for autonomous robot navigation," *International Journal of Advanced Robotic Systems*, vol. 13, 2016. [Online]. Available: https://api.semanticscholar.org/CorpusID:62266727

[2] V. Umamaheswari, S. Amarjyoti, T. Bakshi, and A. Singh, "Steering angle estimation for autonomous vehicle navigation using hough and euclidean transform," in *Proceedings of the IEEE International Conference on Signal Processing, Informatics, Communication and Energy Systems (SPICES)*, 2015, pp. 1–5.

[3] A. rose Harman, "The environmental benefits of self-driving cars," https://greenerideal.com/news/vehicles/driverless-cars-environmental-benefits/, 2024, accessed: 2024-08-22.

[4] "Self-driving vehicles," https://www.government.nl/topics/mobility-public-transport-and-road-safety/self-driving-vehicles, Government of the Netherlands, 2024, accessed: 2024-08-22.

[5] About RDW. RDW. Accessed: 2024-08-22. [Online]. Available: https://www.selfdrivingchallenge.nl/about-rdw

[6] J. Betz, H. Zheng, A. Liniger, U. Rosolia, P. Karle, M. Behl, V. Krovi, and R. Mangharam, "Autonomous vehicles on the edge: A survey on autonomous vehicle racing," *IEEE Open Journal of Intelligent Transportation Systems*, vol. 3, pp. 458–488, 2022.

[7] evGrandPrix. Purdue University. [Online]. Available: https://engineering.purdue.edu/evGrandPrix/autonomous

[8] Formula Student Driverless. SAE International. Accessed: 2024-08-22. [Online]. Available: https://www.fsaeonline.com/

[9] S. Pendleton, H. Andersen, X. Du, X. Shen, M. Meghjani, Y. H. Eng, D. Rus, and M. H. Ang, "Perception, planning, control, and coordination for autonomous vehicles," *Machines*, vol. 1,6, 2017. [Online]. Available: https://api.semanticscholar.org/CorpusID:114862052

[10] E. Joa, Y. Sun, and F. Borrelli, "Monocular camera localization for automated vehicles using image retrieval," 2021, arXiv preprint. [Online]. Available: https://doi.org/10.48550/arXiv.2109.06296

[11] W. Gates, G. Jati, M. Pratama, W. Jatmiko *et al.*, "A modest system of feature-based stereo visual odometry," in *Proceedings of the IEEE 6th International Workshop on Big Data and Information Security (IWBIS)*, 2021, pp. 47–52.

[12] J. Sujatha *et al.*, "Computer vision based novel steering angle calculation for autonomous vehicles," in *Proceedings of the IEEE 2nd International Conference on Robotic Computing (IRC)*, 2018, pp. 143–146.

[13] M. Aladem and S. A. Rawashdeh, "Lightweight visual odometry for autonomous mobile robots," *Sensors*, vol. 18, no. 9, pp. 2837–2851, 2018.

[14] Y. Tanaka, A. Semmyo, Y. Nishida, S. Yasukawa, J. Ahn, and K. Ishii, "Evaluation of underwater vehicle's self-localization based on visual odometry or sensor odometry," in *Proceedings of the IEEE 14th Conference on Industrial and Information Systems (ICIIS)*, 2019, pp. 384–389.

[15] A. Howard, "Real-time stereo visual odometry for autonomous ground vehicles," in *Proceedings of the IEEE/RSJ international conference on intelligent robots and systems*, 2008, pp. 3946–3952.

[16] A. Hernandez-Gutierrez, J. I. Nieto, T. A. Vidal-Calleja, and E. Nebot, "Large scale visual odometry using stereo vision," in *Proceedings of the Australasian Conference on Robotics and Automation (ACRA)*, 2009.

[17] M. Agrawal and K. Konolige, "Real-time localization in outdoor environments using stereo vision and inexpensive gps," in *Proceedings of the IEEE 18th International conference on pattern recognition (ICPR'06)*, vol. 3, 2006, pp. 1063–1068.

[18] B. Kitt, A. Geiger, and H. Lategahn, "Visual odometry based on stereo image sequences with ransac-based outlier rejection scheme," in *Proceedings of the IEEE intelligent vehicles symposium*, 2010, pp. 486–492.

[19] Disparity map post-filtering. Accessed: 2024-08-22. [Online]. Available: https://docs.opencv.org/4.x/d3/d14/tutorial_ximgproc_disparity_filtering.html

[20] S. A. K. Tareen and Z. Saleem, "A comparative analysis of sift, surf, kaze, akaze, orb, and brisk," in *Proceedings of the IEEE International conference on computing, mathematics and engineering technologies (iCoMET)*, 2018, pp. 1–10.

[21] E. Eade, "Gauss-newton/levenberg-marquardt optimization," Technical Report, Tech. Rep., 2013. [Online]. Available: https://mat.uab.cat/~alseda/MasterOpt/optimization.pdf

[22] K. Madsen, H. B. Nielsen, and O. Tingleff, "Methods for non-linear least squares problems (2nd ed.)," Lecture Notes, Informatics and Mathematical Modelling, Technical University of Denmark, 2004. [Online]. Available: https://www2.imm.dtu.dk/pubdb/views/publication_details.php?id=3215

[23] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[24] Perspective-n-point (PnP) pose computation. Accessed: 2024-08-22. [Online]. Available: https://docs.opencv.org/3.4/d5/d1f/calib3d_solvePnP.html

[25] S. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, J. Diebel, P. Fong, J. Gale, M. Halpenny, G. Hoffmann *et al.*, "Stanley: The robot that won the darpa grand challenge," *Journal of field Robotics*, vol. 23, no. 9, pp. 661–692, 2006.

[26] J. Liu, Z. Yang, Z. Huang, W. Li, S. Dang, and H. Li, "Simulation performance evaluation of pure pursuit, stanley, lqr, mpc controller for autonomous vehicles," in *Proceedings of the IEEE international conference on real-time computing and robotics (RCAR)*, 2021, pp. 1444–1449.