FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Perception system for monitoring of several autonomous racing vehicles

Vasco Machado Perdigão

Mestrado em Engenharia Eletrotécnica e de Computadores

Supervisor: Prof.Luís Tiago Paiva

July 31, 2026

Perception system for monitoring of several autonomous racing vehicles

Vasco Machado Perdigão

Mestrado em Engenharia Eletrotécnica e de Computadores

July 31, 2026

Resumo

Esta dissertação apresenta o desenvolvimento e a avaliação de um sistema de perceção visual off-board para rastreamento de veículos RC num ambiente de corrida em pequena escala. O sistema utiliza uma câmara externa fixa, colocada sobre a pista, para observar a área de teste a partir de uma perspetiva global. A imagem é adquirida através de uma configuração baseada numa câmara USB ligada a uma Raspberry Pi, sendo o processamento principal executado num computador externo.

A solução desenvolvida combina deteção de veículos com YOLO, apoio de identidade através de AprilTags, associação de dados, gestão de tracks e estimação de estado baseada num Filtro de Kalman. O YOLO(You Only Look Once) é usado como principal fonte de localização visual, fornecendo as bounding boxes dos veículos, enquanto as AprilTags são integradas como informação auxiliar de identidade quando o marcador é visível e corretamente descodificado. O Filtro de Kalman permite manter uma representação contínua do estado do veículo, incluindo posição, velocidade e aceleração, mesmo perante pequenas variações nas medições visuais.

A avaliação experimental foi realizada numa pista RC(Radio-Controlled) de aproximadamente 1.95 m $\times$ 2.25 m. Os resultados mostram que o sistema consegue manter uma track estável usando apenas deteções visuais, mesmo quando a AprilTag de 5 cm não é detetada. Numa segunda run, com uma AprilTag maior, de aproximadamente 12 cm, o sistema validou o funcionamento híbrido, mantendo o identificador interno da track estável enquanto o marcador era usado como apoio de identidade. Os resultados também mostram que a deteção de AprilTags depende fortemente do tamanho aparente do marcador na imagem, da resolução da câmara e das condições de aquisição.

No geral, o trabalho demonstra a viabilidade de uma arquitetura de perceção off-board para rastreamento e estimação de estado em plataformas RC. A solução desenvolvida constitui uma base para trabalho futuro em cenários multi-veículo, calibração métrica mais rigorosa, aquisição de maior resolução e integração com módulos de planeamento e controlo.

Palavras-chave: Perceção off-board, visão computacional, rastreamento de veículos, YOLO, AprilTag, Filtro de Kalman, veículos RC.

Abstract

This thesis presents the development and evaluation of an off-board visual perception system for tracking RC(Radio-Controlled) vehicles in a small-scale racing environment. The system uses a fixed external camera, positioned above the track, to observe the test area from a bird's-eye view. The image is acquired through a setup based on a USB camera connected to a Raspberry Pi, with the main processing performed on an external computer.

The developed solution combines vehicle detection using YOLO(You Only Look Once), identity support via AprilTags, data association, track management, and state estimation based on a Kalman Filter. YOLO is used as the primary source of visual localization, providing the bounding boxes of the vehicles, while AprilTags are integrated as auxiliary identity information when the tag is visible and correctly decoded. The Kalman Filter allows for maintaining a continuous representation of the vehicle's state, including position, velocity, and acceleration, even in the face of small variations in visual measurements.

The experimental evaluation was conducted on an RC track measuring approximately 1.95 m $\times$ 2.25 m. The results show that the system is able to maintain a stable track using only visual detections, even when the 5 cm AprilTag is not detected. In a second run, using a larger AprilTag of approximately 12 cm, the system validated the hybrid operation, maintaining a stable internal track identifier while the marker was used as an identity backup. The results also show that AprilTag detection strongly depends on the marker's apparent size in the image, the camera resolution, and the acquisition conditions.

Overall, this work demonstrates the feasibility of an off-board perception architecture for tracking and state estimation on RC platforms. The developed solution serves as a foundation for future work on multi-vehicle scenarios, more rigorous metric calibration, higher-resolution acquisition, and integration with planning and control modules.

Keywords: Off-board perception, computer vision, vehicle tracking, YOLO, AprilTag, Kalman Filter, RC vehicles.

UN Sustainable Development Goals

The United Nations Sustainable Development Goals (SDGs) provide a global framework to achieve a better and more sustainable future for all. It includes 17 goals[1] to address the world's most pressing challenges, including poverty, inequality, climate change, environmental degradation, peace, and justice.

The specific Sustainable Development Goals related to this dissertation are:

SDG 4 Ensure inclusive and equitable quality education and promote lifelong learning opportunities for all.

SDG 9 Build resilient infrastructure, promote inclusive and sustainable industrialization and foster innovation.

SDG 11 Make cities and human settlements inclusive, safe, resilient and sustainable.

Table 1: Sustainable Development Goals related to this dissertation.

SDG	Target	Contribution	Performance Indicators and Metrics
4	4.4	Development of technical skills relevant for engineering education and future work in autonomous and intelligent systems.	Implementation of a working prototype; development of detection, tracking, and state estimation modules; experimental validation using data collected from the RC track.
9	9.5	Contributes to research and innovation in robotic perception by implementing and evaluating an off-board vehicle tracking and state estimation prototype in a small-scale racing environment.	Experimental evaluation of vision-based and hybrid YOLO–AprilTag runs; analysis of track continuity, missed detections, trajectory estimation, velocity, and acceleration.
11	11.2	Addresses perception and tracking problems that are relevant to intelligent transport systems and autonomous mobility research.	Analysis of hybrid identity support; evaluation of trajectory and motion estimates from an off-board camera view.

[1] https://sdgs.un.org/goals

Acknowledgements

Em primeiro lugar, gostaria de agradecer aos meus pais e ao meu irmão. Obrigado pelo apoio constante, pela paciência e pelo incentivo ao longo de todos estes anos. A vossa confiança em mim foi sempre uma fonte de força, especialmente nos momentos mais exigentes deste percurso.

Gostaria também de agradecer aos meus avós e aos meus tios, pelo carinho, pela presença e pelo apoio que sempre me deram. Cada um, à sua maneira, teve um papel muito importante na minha vida e neste caminho.

À Maria, obrigado por estares sempre ao meu lado durante este processo. Obrigado pela paciência, pela motivação e por me ajudares a manter a calma e a perspetiva nos momentos mais difíceis. O teu apoio tornou este percurso mais leve e mais especial.

Agradeço também ao meu orientador, Luís Tiago Paiva, pela orientação, disponibilidade e feedback ao longo desta dissertação. O seu acompanhamento foi essencial para desenvolver este trabalho e para o melhorar em cada fase.

Quero ainda agradecer ao Conrado Guimarães da Costa, pela ajuda, orientação técnica e disponibilidade durante o desenvolvimento deste projeto. O seu apoio foi muito importante para a implementação e para o trabalho experimental realizado no laboratório.

A todos os que trabalharam comigo no laboratório, obrigado pela colaboração, pelas discussões, pela ajuda nos problemas que foram surgindo e pelos momentos partilhados que tornaram o trabalho mais agradável.

Aos meus amigos da FEUP, obrigado por terem feito parte destes anos. Obrigado pelas horas de estudo, pelas conversas, pelo stress partilhado e por tornarem este percurso académico muito mais memorável.

Por fim, agradeço também aos meus amigos fora da FEUP. Obrigado pela amizade, pelo apoio e por me lembrarem que existe vida para além dos prazos, exames e dissertações.

A todos os que, de alguma forma, contribuíram para este percurso, obrigado.

Vasco Perdigão

"Engineering is the art of directing the great sources of power in nature for the use and convenience of man."

Thomas Tredgold

Declaration of honour

I declare that this dissertation is my own work and has not previously been Submitted or assessed at this or any other institution. References to other authors (statements, ideas, thoughts), as well as the use of published works, strictly comply with the rules of attribution and are duly identif ied in the text and in the bibliographic references, in accordance with the applicable referencing standards. I am aware that the practice of plagiarism or self-plagiarism constitutes an academic offence, as established in the U.Porto Code of Ethics.
I also declare that I have used Artificial Intelligence tools to complete part(s) of this dissertation, and that this use and its results are duly noted and have been carefully checked for errors, inaccuracies, unfounded attributions or other forms of bias in content and arguments, as well as determining authorship.

Vasco Machado Perdigão

Contents

List of Figures

List of Tables

List of Acronyms

ArUco	Augmented Reality University of Cordoba
CLEAR MOT	Classification of Events, Activities and Relationships Multi-Object Tracking
CNN	Convolutional Neural Network
DETR	Detection Transformer
IDF1	Identification F1 Score
MOT	Multi-Object Tracking
MOTA	Multiple Object Tracking Accuracy
MOTP	Multiple Object Tracking Precision
OC-SORT	Observation-Centric Simple Online and Realtime Tracking
OpenCV	Open Source Computer Vision Library
RC	Radio-Controlled
R-CNN	Region-based Convolutional Neural Network
SORT	Simple Online and Realtime Tracking
SSD	Single Shot MultiBox Detector
USB	Universal Serial Bus
YOLO	You Only Look Once

Chapter 1

Introduction

Autonomous racing has become an important experimental field for the development and testing of perception, planning, control, and decision-making algorithms. In this context, small-scale racing platforms based on Radio-Controlled (RC) vehicles provide a practical and accessible environment for studying problems that also appear in full-scale autonomous systems. These platforms make it possible to test algorithms in real physical conditions while keeping the cost, risk, and complexity of the experiments relatively low.

In autonomous racing, perception is one of the core components of the system. The vehicle or the external infrastructure must be able to identify relevant objects, estimate their position, and track their motion over time. This information is then used by higher-level modules, such as planning and control, to support safe and consistent vehicle operation. Even in a controlled small-scale track, perception remains challenging because image measurements can be noisy, detections may be intermittent, and the vehicle can move quickly relative to the camera frame rate.

This dissertation focuses on an off-board vision-based perception system for small-scale RC racing. Instead of relying on a camera mounted on the vehicle, the system uses a fixed external camera to observe the racing area from above. This configuration provides a global view of the track and allows the vehicle motion to be monitored from a common reference frame. The perception pipeline detects the vehicle using a learning-based object detector, uses AprilTags as auxiliary identity cues when available, maintains tracks over time, and estimates motion-related quantities using a Kalman Filter.

1.1 Context and Motivation

Autonomous vehicles require reliable perception systems to understand their surrounding environment. In racing scenarios, this requirement becomes especially relevant because the vehicle motion can be fast, the available reaction time is limited, and errors in state estimation can affect the behaviour of downstream planning and control modules. For this reason, autonomous racing has become a useful testbed for evaluating algorithms under dynamic conditions.

Small-scale RC racing platforms offer a controlled way to study these problems. They allow experiments to be carried out in a laboratory environment, while still preserving important aspects of autonomous driving, such as vehicle detection, motion estimation, trajectory tracking, and real-time processing. Compared with full-scale vehicles, RC platforms are easier to instrument, safer to test, and more flexible for rapid development.

Perception in autonomous racing can be approached from an onboard or off-board perspective. In onboard perception, sensors are placed directly on the vehicle, and the system observes the environment from the vehicle's own point of view. In off-board perception, one or more external sensors observe the track and provide information about the vehicle from a global viewpoint. This dissertation follows the off-board approach.

The off-board configuration is particularly suitable for a small-scale racing track. A fixed camera placed above the track can capture a large part of the environment at once, making it possible to estimate vehicle motion in a shared coordinate system. This also reduces the sensing and computational burden on the RC vehicle itself, since the perception pipeline can run on an external processing computer. The approach is therefore useful for experimental setups where the main objective is to monitor vehicle motion, test tracking logic, and obtain state estimates for later use by other modules.

However, an off-board vision-based system still has several practical challenges. The vehicle must be detected reliably in the image, even when its appearance changes with orientation or lighting. The system must maintain the same identity over time, instead of treating each detection as an independent observation. It must also handle short missed-detection intervals without immediately losing the vehicle track. Finally, the system should estimate quantities such as position, velocity, and acceleration in a way that is useful for analysing the vehicle motion.

1.2 Problem Statement

The problem addressed in this dissertation is the development of a real-time off-board perception pipeline capable of detecting, tracking, and estimating the state of RC vehicles on a small-scale racing track.

A simple frame-by-frame detection approach is not sufficient for this purpose. Detections can be missed due to motion blur, changes in illumination, reflections, partial visibility, or temporary instability of the detector. If each frame is processed independently, these interruptions can cause the vehicle identity to be lost or duplicated. A tracking system is therefore required to associate detections over time and maintain a consistent internal representation of the vehicle.

The system must also estimate motion-related quantities. The camera directly provides image-based position measurements, but velocity and acceleration are not measured directly. These quantities must be inferred from the temporal evolution of the vehicle position. Direct numerical differentiation of noisy image measurements can produce unstable estimates, especially when detections are not perfectly smooth or when the frame interval varies. A filtering approach is therefore needed to obtain more consistent state estimates.

Another challenge is vehicle identity. In a single-vehicle scenario, identity preservation is simpler, but it is still important to avoid track fragmentation. In future multi-vehicle scenarios, identity becomes more difficult because several vehicles may be visible at the same time. This dissertation explores the use of AprilTags as auxiliary identity markers. When visible and decoded, an AprilTag can provide an explicit identifier for the vehicle. However, marker detection depends on factors such as tag size, camera resolution, motion blur, and viewing angle. For this reason, AprilTags are treated as identity support rather than as the only tracking mechanism.

The main problem can therefore be summarized as follows: how to implement and evaluate an off-board perception system that combines visual vehicle detection, auxiliary marker-based identity information, multi-object tracking logic, and Kalman-based state estimation in a real-time RC racing setup.

1.3 Objectives

The main objective of this dissertation is to design, implement, and evaluate an off-board vision-based perception pipeline for RC vehicle tracking and state estimation. More specifically, the dissertation aims to design an off-board acquisition architecture using a fixed external camera and a processing computer; to implement You Only Look Once (YOLO)-based vehicle detection from the overhead camera view; to integrate AprilTag detections as auxiliary identity information when the marker is visible; to develop a tracking module capable of maintaining consistent vehicle tracks over time; to implement data association and track management strategies to handle missed detections and avoid unnecessary track fragmentation; to estimate vehicle position, velocity, and acceleration using a Kalman Filter associated with each active track; to log relevant detection, tracking, and state estimation outputs for later analysis; and to evaluate the system using experimental sequences collected on a small-scale RC racing track.

These objectives focus on the perception and tracking components of the system. The dissertation does not aim to implement a complete autonomous racing stack including path planning or closed-loop vehicle control. Instead, it provides the perception layer that can support such modules in future work.

1.4 Proposed Approach

The proposed system follows an off-board vision-based architecture. A fixed camera is positioned above the experimental RC racing track, providing a global view of the racing area. The camera stream is acquired through a Raspberry Pi and transmitted to an external processing computer, where the perception pipeline is executed.

The processing pipeline begins with frame acquisition and validation. Each frame is processed by a YOLO-based detector trained to identify the RC vehicle from the external camera viewpoint. The detector returns bounding boxes and confidence values, and the center of each bounding box is used as the main visual position measurement for the tracking module.

AprilTag detection is also included in the pipeline. Each tag can provide a decoded identifier when the marker is visible and sufficiently clear in the image. In the proposed system, AprilTags are not used as the only localization source. Instead, they provide auxiliary identity information that can support the track associated with a vehicle. This design allows the system to continue operating even when the marker is temporarily unavailable.

The tracking module maintains active vehicle tracks across frames. For each frame, existing tracks are predicted forward using their Kalman Filter state model. Current detections are then associated with predicted tracks using a data association strategy based on spatial compatibility, detection information, and available tag identity cues. Tracks that are not matched in a given frame are not immediately deleted. Instead, they are kept alive for a limited number of frames, allowing the system to handle short detection gaps.

Each active track is associated with a Kalman Filter. The filter estimates the vehicle state over time, including position, velocity, and acceleration. Position measurements are obtained from the visual detections, while velocity and acceleration are inferred from the temporal evolution of the state. This provides a continuous motion estimate that is more structured than a sequence of independent detections.

The system also generates visual outputs and structured logs. The visual output is useful for debugging and qualitative inspection, while the logs allow the system behaviour to be analysed after each experimental run. These logs are used to evaluate detection confidence, track continuity, missed frames, trajectory estimation, and motion estimates.

1.5 Main Contributions

The main contributions of this dissertation are mainly related to the design, implementation, and experimental validation of the proposed perception pipeline. A first contribution is the implementation of an off-board vision-based perception architecture for RC vehicle tracking on a small-scale racing track. This contribution is implemented in Chapter 4 through the software architecture, camera stream integration, real-time execution loop, and processing structure used to connect the external camera, the Raspberry Pi, and the processing computer. It is evaluated in Chapter 5 through the experimental setup, the acquisition configuration, and the real-time performance indicators defined for the complete pipeline. The corresponding results are presented in Chapter 6 through the evaluated sequences and the analysis of the system behaviour during execution. Chapter 7 concludes that this architecture provides a feasible basis for off-board perception in small-scale RC racing experiments.

A second contribution is the integration of YOLO-based vehicle detection with AprilTag-based auxiliary identity support. This contribution is implemented in Chapter 4 through the detection modules, where YOLO is used as the main source of vehicle localization and AprilTags are used as optional identity cues when the marker is visible and correctly decoded. It is evaluated in Chapter 5 using detection-related indicators, including YOLO confidence values, detection availability, decoded AprilTag identifiers, and qualitative inspection of the annotated output. The

results are analysed in Chapter 6 through the vision-based run, the hybrid YOLO–AprilTag run, and the two-vehicle run with 5 cm and 10 cm markers. Chapter 7 concludes that YOLO provides the main localization information, while AprilTags can support identity assignment when their apparent size in the image is sufficient.

A third contribution is the development of a tracking strategy combining data association, track management, and Kalman-based prediction. This contribution is implemented in Chapter 4 through the track data structures, detection-to-track association logic, track creation and removal rules, and missed-detection handling mechanisms. It is evaluated in Chapter 5 using tracking indicators such as active tracks, track identifiers, missed frames, duplicated tracks, identity preservation, and recovery after temporary detection losses. The results are presented in Chapter 6 through the analysis of track continuity in the vision-based, hybrid, and two-vehicle sequences. Chapter 7 concludes that the implemented tracker is able to maintain stable single-vehicle tracks and two simultaneous vehicle tracks under the tested conditions.

A fourth contribution is the estimation of vehicle position, velocity, and acceleration from image-based measurements. This contribution is implemented in Chapter 4 through the Kalman Filter state vector, prediction and update cycle, noise modelling, and coordinate conversion procedure. It is evaluated in Chapter 5 through state estimation indicators, including measured-versus-estimated position, trajectory plots, velocity profiles, acceleration profiles, and filter behaviour during missed detections. The corresponding results are shown in Chapter 6 through the trajectory and state estimation analysis for the evaluated runs. Chapter 7 concludes that the Kalman Filter provides temporally consistent position and velocity estimates, while acceleration remains more sensitive to image noise and frame-timing variations.

A fifth contribution is the creation of structured logs and visual outputs for analysing detection, tracking, and state estimation behaviour after each experimental run. This contribution is implemented in Chapter 4 through the visualization, debugging, output generation, and logging modules. It is evaluated in Chapter 5 by defining the logged variables and qualitative assessment criteria used to inspect the system behaviour. These logs support the plots and analyses presented in Chapter 6, including detection confidence, tag availability, active tracks, missed frames, trajectories, and motion estimates. Chapter 7 concludes that these outputs are essential for interpreting the behaviour and limitations of the prototype.

Finally, the dissertation contributes with an experimental evaluation of the complete system under vision-based, hybrid YOLO–AprilTag, and two-vehicle conditions. This contribution is prepared in Chapter 5 through the definition of the experimental scenarios, data collection procedure, and evaluation indicators. The results are presented and discussed in Chapter 6, where the role of each perception component is analysed across the different runs. Chapter 7 uses these results to conclude that the proposed system is a working prototype adapted to the constraints of an RC racing environment, while also identifying the need for further evaluation in more demanding multi-vehicle scenarios.

Part of the work developed in this dissertation was also submitted as a conference paper to CONTROLO 2026 [1].

These contributions are mainly practical and experimental. The work does not propose a new object detection algorithm or a new filtering theory. Instead, it brings together existing perception and estimation techniques into a working prototype whose implementation, evaluation, results, and conclusions are explicitly connected throughout Chapters 4 to 7.

1.6 Dissertation Structure

This dissertation is organized into seven chapters.

Chapter 1 introduces the context and motivation of the work, defines the problem addressed in the dissertation, presents the main objectives, summarizes the proposed approach, and lists the main contributions.

Chapter 2 presents the background and state of the art related to autonomous racing, off-board vision-based perception, camera-based detection, fiducial markers, multi-object tracking, data association, and Kalman-based state estimation.

Chapter 3 describes the proposed system architecture and methodology. It explains the acquisition setup, perception pipeline, detection strategy, hybrid YOLO–AprilTag approach, tracking logic, data association method, and state estimation model.

Chapter 4 presents the implementation of the proposed system. It describes the software architecture, main processing loop, detection modules, tracking implementation, visualization, logging, and practical development details.

Chapter 5 describes the experimental setup and evaluation methodology. It presents the RC racing track, camera and acquisition configuration, vehicle and marker setup, data collection procedure, and evaluation indicators.

Chapter 6 presents and discusses the experimental results. It analyses the vision-based and hybrid runs, YOLO detection behaviour, AprilTag detection behaviour, track continuity, missed detections, trajectory estimation, Kalman Filter behaviour, and velocity and acceleration estimates.

Chapter 7 concludes the dissertation by summarizing the main findings, identifying the limitations of the current prototype, and suggesting directions for future work.

Chapter 2

Background and State of the Art

This chapter introduces the main concepts and related work required to understand the perception problem addressed in this dissertation. First, the background section presents the general context of autonomous racing, small-scale RC platforms, off-board vision-based perception, camera calibration, and recursive state estimation. Then, the state-of-the-art section reviews existing approaches for vehicle detection, fiducial marker-based identification, learning-based object detection, Multi-Object Tracking (MOT), motion estimation, and real-time perception. Finally, the chapter identifies the main research gaps that motivate the development of a modular perception pipeline for small-scale autonomous racing.

2.1 Background

2.1.1 Autonomous Racing and RC Platforms

Autonomous racing has become an important research benchmark in robotics and autonomous systems due to the demanding conditions it imposes on perception, planning, and control algorithms [2], [3], [4], [5]. Unlike conventional autonomous driving, where safety, comfort, and traffic-rule compliance are dominant objectives, autonomous racing focuses on high-speed operation, fast decision-making, and accurate state estimation under strict timing constraints. These characteristics make racing platforms useful for testing robotic systems that must operate reliably in dynamic and time-critical scenarios.

Scaled RC vehicles are widely used in academic autonomous racing research because they provide a safe, affordable, and repeatable environment for experimentation [2], [3], [6], [7]. Although these platforms do not reproduce the full complexity of real autonomous vehicles, they capture several relevant challenges, such as fast vehicle motion, noisy visual measurements, limited sensing conditions, and the need for low-latency processing. Their small scale also allows experiments to be performed in controlled indoor environments, enabling rapid iteration during system development and validation.

Several research platforms have explored this idea at different scales. Liniger et al. [6] used 1:43 scale RC cars to study optimization-based autonomous racing, while AutoRally provides a

larger experimental platform for aggressive autonomous driving research [7], [8]. These examples show that scaled vehicles can be useful not only for control experiments, but also for studying perception, localization, and state estimation under demanding motion conditions.

RC racing platforms can support different sensing configurations. Some systems rely on onboard sensing, where each vehicle perceives the environment from its own perspective. Other setups use external sensing infrastructure, such as fixed overhead cameras, to monitor the racing area from a global viewpoint [2]. Off-board sensing is particularly useful in controlled laboratory environments because it allows multiple vehicles to be observed within a common reference frame. However, this configuration also introduces specific challenges, including camera calibration, image resolution limitations, perspective effects, acquisition latency, and possible detection failures caused by motion blur or partial occlusions.

2.1.2 Off-board Vision-Based Perception

Vision-based perception is widely used in robotics because cameras provide rich spatial information using relatively low-cost and flexible sensing hardware [9]. Visual data can support several perception tasks, including object detection, localization, tracking, mapping, and motion estimation [9], [10]. For this reason, cameras are commonly used in autonomous systems where the environment must be interpreted continuously and in real time.

In autonomous vehicle applications, visual perception pipelines typically process image sequences to detect relevant objects and estimate their state over time [10]. In autonomous racing, this requirement becomes more demanding, as perception systems must provide timely and consistent information about vehicle positions and motion under strict latency constraints [2], [3]. Even small delays or noisy estimates may affect downstream planning and control decisions, especially when vehicles move quickly or interact closely with each other.

Vision-based perception can be implemented using onboard cameras or external sensing infrastructure, as illustrated in Fig. 2.1. Onboard perception provides a vehicle-centered view of the environment and is more representative of full-scale autonomous driving [10]. However, in small-scale RC racing platforms, off-board perception using a fixed overhead camera is often a practical alternative. This configuration provides a global view of the track and allows multiple vehicles to be observed within the same reference frame [2]. This configuration simplifies multi-vehicle monitoring and state comparison, particularly in controlled laboratory environments.

Despite these advantages, off-board vision-based perception also introduces practical constraints. The image stream must be acquired with sufficiently low latency, and the perception pipeline must remain robust to perspective effects, limited image resolution, motion blur, illumination changes, and partial occlusions [9]. These constraints are particularly relevant in small-scale racing environments, where vehicles may occupy a limited number of pixels and where small errors in image measurements can affect the estimated vehicle state.

Practical computer vision systems often rely on established software libraries for image acquisition, processing, and visualization. Open Source Computer Vision Library (OpenCV) is one of the most widely used libraries for this purpose and provides tools for camera access, image processing,

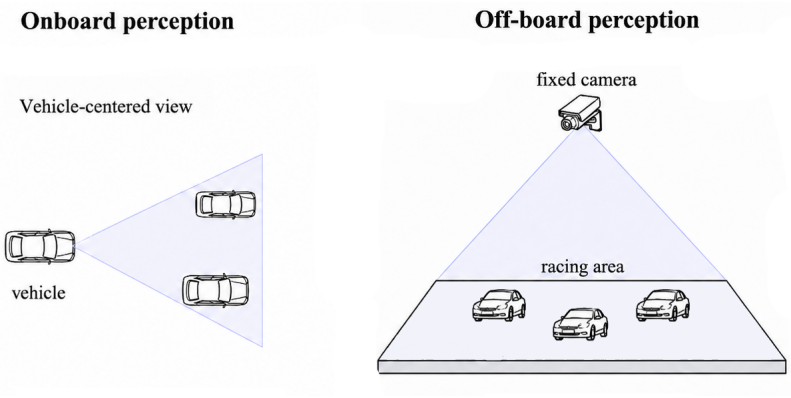

Figure 2.1: Comparison between onboard and off-board vision-based perception. In onboard perception, each vehicle observes the environment from its own perspective, while in off-board perception a fixed external camera provides a global view of the racing area and allows multiple vehicles to be monitored within a shared reference frame. Figure generated with the support of ChatGPT and manually reviewed by the author

calibration, and feature extraction. In this dissertation, this type of software infrastructure is relevant because the perception pipeline depends on real-time frame acquisition, image processing, and visualization of tracking results.

2.1.3 Camera Calibration and Coordinate Transformation

Vision-based perception systems typically operate first in image coordinates, where object positions are represented in pixels. However, for state estimation, trajectory analysis, and control-oriented applications, these measurements often need to be expressed in a metric reference frame. This requires a calibration procedure that relates image coordinates to physical coordinates in the environment [9], [11], [12].

Since visual detections are first obtained in image coordinates, a coordinate transformation is required to express vehicle positions in a metric reference frame. Fig. 2.2 illustrates this transformation conceptually, using two reference points on the racing plane to define the pixel-to-meter scale.

Camera calibration can involve estimating intrinsic parameters, such as focal length and distortion coefficients, as well as extrinsic parameters, which describe the pose of the camera relative to the observed scene [11], [12]. In planar environments, such as small-scale racing tracks, simpler geometric transformations may also be used when vehicle motion is assumed to occur on a flat surface [9], [12]. These transformations allow image measurements to be converted into coordinates that are more meaningful for motion analysis.

For overhead camera setups, calibration is especially important because all vehicle positions are inferred from visual measurements. Errors in calibration can propagate to the estimated position, velocity, and acceleration. Therefore, even when the perception system is designed primarily for monitoring, a consistent relationship between pixel coordinates and physical distances is necessary to interpret vehicle motion correctly.

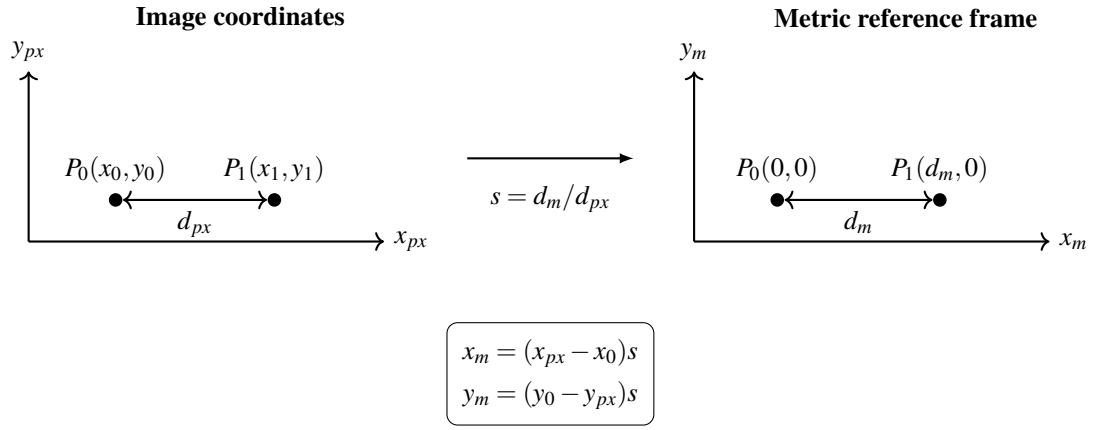

Figure 2.2: Conceptual transformation from image coordinates to a metric reference frame defined on the racing plane. Pixel measurements obtained from visual detections are converted into metric quantities using calibration reference points.

2.1.4 Recursive State Estimation

Visual detections provide discrete measurements of object position, but these measurements may be noisy, intermittent, or affected by detection errors. Recursive state estimation addresses this problem by combining measurement information with a dynamic motion model over time [13]. This allows the estimated state to remain temporally consistent even when individual measurements are imperfect.

The Kalman Filter is one of the most widely used recursive estimation methods for linear systems with Gaussian noise assumptions [13]. In visual tracking, Kalman filtering is commonly used to estimate position and velocity from noisy position measurements [14]. The prediction step propagates the current state using a motion model, while the update step corrects the prediction using the latest available measurement.

Recursive filtering is particularly relevant in vision-based racing scenarios because velocity and acceleration are not usually measured directly. Instead, these quantities must be inferred from position measurements over time. By including motion variables in the estimated state, filtering methods can reduce the instability associated with direct numerical differentiation and provide smoother estimates for downstream interpretation or control.

2.2 State of the Art

2.2.1 Vehicle Detection in Controlled Racing Environments

Multi-object detection is a central problem in computer vision, involving the identification and localization of multiple objects within the same image or video frame [9], [10]. In autonomous racing environments, this task corresponds to detecting all vehicles present in the scene, often under conditions involving fast motion, close interactions, partial occlusions, and strict real-time

constraints [2], [3]. Reliable detection is therefore a fundamental requirement for subsequent tracking, data association, and state estimation stages.

In controlled RC racing environments, simple detection strategies such as color-based segmentation, background subtraction, or blob detection may be used due to their low computational cost and straightforward implementation [9], [15]. These methods can be effective when the visual appearance of the vehicles and the lighting conditions remain stable. However, their performance is strongly affected by illumination changes, shadows, reflections, motion blur, and variations in object appearance. As a result, purely classical detection methods may not provide sufficient robustness for dynamic multi-vehicle racing scenarios.

These classical methods remain relevant as lightweight baselines, especially in constrained laboratory setups. Nevertheless, their dependence on stable visual conditions limits their applicability when the scene contains changing lighting, fast-moving vehicles, or visual ambiguity between different objects. This has motivated the use of more structured identification methods and learning-based detectors in robotic perception systems.

2.2.2 Fiducial Marker-Based Identification

Marker-based approaches, using visual fiducials such as AprilTags and Augmented Reality University of Cordoba (ArUco) markers, provide explicit and reliable identity information and are widely used in robotics research [16], [17], [18], [19]. These markers are designed to be detected through standardized image processing and geometric decoding procedures, allowing each physical object to be associated with a unique identifier when the marker is visible. This property simplifies vehicle identification and can reduce ambiguity in multi-object experimental testbeds.

Fiducial markers are particularly attractive in controlled robotics environments because they provide a direct link between visual detections and physical entities. When a marker is detected successfully, the identity assignment problem becomes significantly simpler than in appearance-based tracking. This can reduce the risk of identity switches and improve interpretability in experimental systems.

However, fiducial marker-based systems also present important limitations. Their detection performance depends on camera resolution, marker size, viewing angle, motion blur, occlusions, and the specific detector implementation [20]. In small-scale racing scenarios, these factors become particularly relevant because vehicles move quickly, markers occupy a limited number of pixels, and the camera may observe them from non-ideal angles. Moreover, fiducial markers require physical modification of the vehicles and may fail temporarily during close interactions or high-speed motion. For this reason, relying exclusively on marker detection may not be sufficient for continuous vehicle localization.

2.2.3 Learning-Based Object Detection

Learning-based object detectors, particularly Convolutional Neural Network (CNN) detectors, provide an alternative approach for vehicle detection under more variable visual conditions [21].

Instead of relying on predefined colors or visible fiducial patterns, these detectors learn object appearance from annotated examples and output bounding boxes with confidence scores. This makes them suitable for detecting objects whose appearance may vary due to orientation, lighting, or partial visibility.

Object detection research has evolved from two-stage detectors, such as Faster Region-based Convolutional Neural Network (R-CNN), to single-stage detectors, such as Single Shot MultiBox Detector (SSD) and YOLO. Single-stage detectors are often more suitable for real-time applications because they reduce the detection process to a more direct inference pipeline [22], [23], [24]. More recent approaches, including YOLOv4 and transformer-based detectors such as Detection Transformer (DETR), further illustrate the range of possible trade-offs between accuracy, speed, and model complexity [25], [26]. In this dissertation, the focus is not on developing a new detector, but on using a practical learning-based detector as part of a complete tracking pipeline.

YOLO-based detectors are commonly used when real-time object detection is required, since they are designed to perform detection in a single forward pass through the network [24], [27]. Modern implementations, such as Ultralytics YOLO, provide practical tools for training and deploying object detectors in custom environments [28]. Reviews of the YOLO family also show how the architecture has evolved through different versions while keeping real-time detection as a central goal [29]. In controlled small-scale racing scenarios, custom datasets collected from the target environment can reduce domain-specific detection errors and improve detector suitability.

Despite their advantages, learning-based detectors also have limitations. They require annotated training data, sufficient computational resources, and careful evaluation to ensure reliable performance under the expected operating conditions [27], [28]. Additionally, standard object detectors provide bounding boxes and confidence scores, but do not inherently preserve object identity across frames. Therefore, detection alone is insufficient for multi-vehicle tracking; it must be combined with data association and state estimation.

These approaches show that multi-object detection in small-scale autonomous racing involves a trade-off between computational efficiency, detection robustness, and identity information. Classical image processing methods are efficient and simple to deploy, but sensitive to visual changes [9], [15]. Fiducial markers provide explicit identity cues, but depend on marker visibility and imaging conditions [17], [19], [20]. Learning-based detectors provide more flexible appearance-based detection, but require annotated data and sufficient computational resources for real-time operation [24], [27], [28]. Understanding these trade-offs is essential for designing robust perception pipelines for multi-vehicle RC racing environments.

Table 2.1 summarizes the main differences between fiducial marker-based approaches and learning-based detection methods. These two families are particularly relevant in controlled small-scale racing environments because they provide complementary types of information: fiducial markers can provide explicit identity information, while learning-based detectors provide appearance-based vehicle detections.

As shown in Table 2.1, fiducial markers are particularly useful when explicit vehicle identity is required, since each detected marker can be directly associated with a physical vehicle. However,

Table 2.1: Comparison between fiducial marker-based and learning-based detection approaches.

Criterion	**Fiducial markers**	**Learning-based methods**
Typical output	ID + position	Bounding boxes
Main strength	Explicit identity	Robust detection
Main limitation	Visibility-dependent	Requires training data
Real-time suitability	High	Model-dependent

their reliability depends on marker visibility, viewing angle, image resolution, motion blur, and occlusions. Learning-based methods, on the other hand, are more flexible for detecting vehicles based on visual appearance, but they require annotated training data and sufficient computational resources. This comparison highlights the trade-off between identity reliability and continuous visual detection in small-scale autonomous racing perception systems.

2.2.4 Multi-Object Tracking and Data Association

MOT aims to maintain temporally consistent object identities across a sequence of frames. In tracking-by-detection systems, objects are first detected independently in each frame, and the resulting detections must then be associated with existing tracks over time [14], [30]. This association step is essential because incorrect matches can lead to identity switches, duplicated tracks, or fragmented trajectories, directly reducing the reliability of the estimated vehicle states.

A common tracking-by-detection workflow is illustrated in Fig. 2.3. First, an image frame is processed by an object detector, producing a set of detections. Each detection usually provides a position measurement, such as the center of a bounding box, together with additional information such as a confidence score. Since detections are computed independently at each frame, they do not automatically define a continuous trajectory. To obtain temporally consistent tracks, the system must predict the expected state of each existing object, associate current detections with predicted tracks, update the corresponding states, and manage the creation or deletion of tracks over time.

The prediction stage shown in Fig. 2.3 is commonly performed using a recursive state estimator, such as a Kalman Filter, which propagates each track according to a motion model **barshalom2004**, [13]. This prediction provides an expected object position in the next frame and helps restrict the set of plausible detection-to-track assignments. Prediction is particularly important when visual detections are noisy, intermittent, or temporarily unavailable, since it allows tracks to be maintained for short periods even in the absence of direct measurements.

After prediction, the data association stage assigns current detections to existing tracks. Simple nearest-neighbor strategies associate each detection with the closest predicted track, but these local decisions may become unreliable when multiple objects are close to each other. For this reason, many tracking-by-detection frameworks formulate data association as a global assignment problem. Simple Online and Realtime Tracking (SORT), for example, combines Kalman filtering for motion prediction with the Hungarian algorithm for assigning detections to tracks in real time [14], [31].

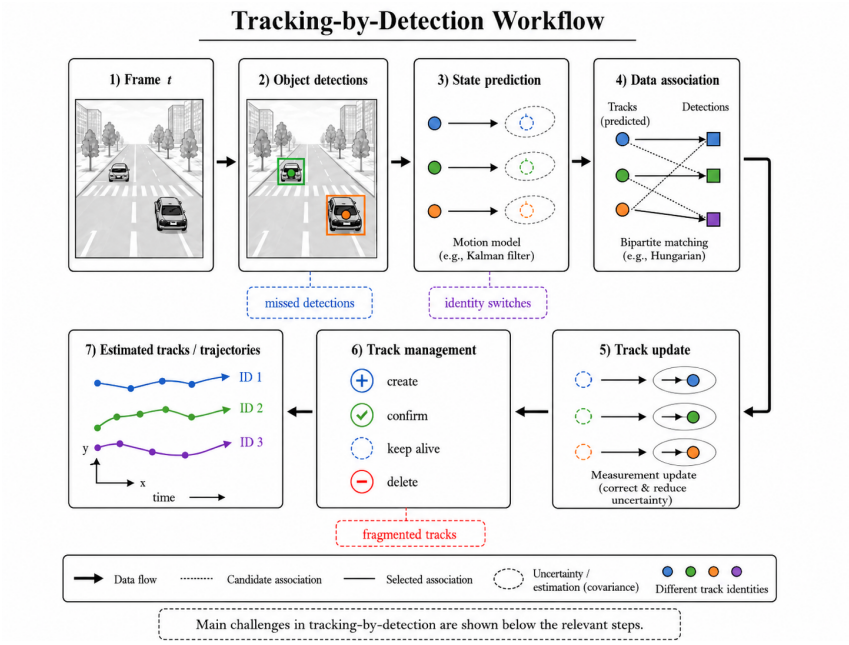

Figure 2.3: General tracking-by-detection workflow. The diagram also illustrates common challenges such as missed detections, identity switches, and fragmented tracks. Figure generated with the support of ChatGPT and manually reviewed by the author.

The Hungarian algorithm selects the set of assignments that minimizes a global cost, reducing ambiguities that may arise from purely local matching decisions.

The association cost can be based on different criteria, depending on the available detection information. In bounding-box-based tracking, the intersection-over-union between predicted and detected boxes is often used as a measure of spatial compatibility [14]. In other systems, distance-based costs or gating strategies are used to reject assignments that are physically implausible. Gating is important because it prevents detections that are too far from a predicted track from being incorrectly assigned, thereby reducing the probability of identity switches.

Once detections are associated with tracks, the track update stage corrects the predicted state using the new measurement. This update reduces uncertainty and improves temporal consistency. Tracks that are not matched with a detection may be kept alive for a limited number of frames using prediction only, while unmatched detections may be used to create new tentative tracks. This track management process, also represented in Fig. 2.3, is necessary to handle missed detections, newly appearing objects, and objects that leave the scene.

More advanced approaches, such as DeepSORT, extend the SORT framework by incorporating appearance descriptors into the association process [30]. Appearance information can improve identity preservation when multiple objects move close to each other or when their predicted positions become ambiguous. However, appearance-based association increases computational complexity and may require additional training data or feature extraction steps, which can be a limitation in real-time small-scale robotic systems.

Recent tracking-by-detection methods continue to address the same core problem of associating

detections over time. ByteTrack improves tracking performance by also using low-confidence detections during association, while Observation-Centric Simple Online and Realtime Tracking (OC-SORT) revisits the observation and motion modelling assumptions of SORT to improve robustness in challenging sequences [32], [33]. Although these methods were developed mainly for general MOT benchmarks, they are relevant for this dissertation because they illustrate the importance of detection quality, association strategy, and motion prediction.

Evaluation of MOT systems is also a specific research topic. Metrics such as the Classification of Events, Activities and Relationships Multi-Object Tracking (CLEAR MOT) measures and benchmarks such as MOT16 were developed to quantify tracking accuracy, precision, identity switches, and fragmentation in annotated video sequences [34]. These metrics are useful when frame-by-frame ground truth is available. In this dissertation, the experimental evaluation relies instead on logged indicators and visual inspection, because a complete annotated benchmark was not available.

In autonomous RC racing environments, data association is especially challenging because vehicles may move quickly, interact closely, and produce intermittent or noisy visual detections. Marker-based identification can reduce association ambiguity when fiducial markers are visible, since markers provide explicit identity information. However, marker detections may fail due to motion blur, occlusions, viewing angle, marker size, or camera configuration [17], [20]. Therefore, even when identity cues are available, robust association and state prediction mechanisms remain essential for maintaining consistent multi-vehicle trajectories.

2.2.5 Vision-Based Motion Estimation

Velocity estimation is a fundamental component of perception systems for autonomous vehicles, since it provides information about both the magnitude and direction of motion. While image-based detection methods can provide object positions in pixel coordinates, these measurements must often be transformed into a metric reference frame through camera calibration or planar projection methods before they can be used for motion analysis [9], [11], [12].

In vision-based tracking systems, velocity can be estimated from consecutive position measurements. However, direct numerical differentiation of visual measurements is highly sensitive to noise, detection jitter, missed detections, and variations in frame rate. Small errors in measured position can therefore produce large fluctuations in the estimated velocity, especially in real-time systems operating with fast-moving objects.

For this reason, velocity estimation is commonly integrated into the state estimation process. Recursive filtering techniques, such as the Kalman Filter, estimate velocity as part of the system state by combining noisy position measurements with a dynamic motion model [13]. This approach provides smoother and more temporally consistent velocity estimates than direct frame-to-frame differentiation, while also allowing short-term prediction when measurements are temporarily unavailable.

Different motion models can be used depending on the dynamics of the tracked objects. Constant-velocity models are widely used in visual tracking due to their simplicity and computational efficiency [14]. In more dynamic scenarios, where objects may accelerate, brake, or change direction rapidly, constant-acceleration models can provide a richer representation of motion by including acceleration components in the estimated state. The choice of motion model therefore involves a trade-off between model complexity, robustness, and real-time performance.

In autonomous racing applications, accurate motion estimation is particularly important because vehicles move quickly and interact in close proximity. Reliable velocity and acceleration estimates support trajectory prediction, interaction analysis, performance evaluation, and future integration with planning and control modules [2], [3]. For small-scale RC racing platforms, these requirements are further affected by camera frame rate, image resolution, calibration accuracy, and the latency of the vision-based perception pipeline.

2.2.6 Real-Time Perception Constraints

Real-time operation is a defining requirement of autonomous racing systems, since perception, planning, and control modules must operate within tight timing constraints [2], [3]. In these scenarios, delayed or inconsistent perception outputs can reduce the relevance of the estimated vehicle state, as the true vehicle position and velocity may have already changed by the time the information reaches downstream modules. This is particularly critical in racing environments, where vehicles move quickly and control decisions depend on up-to-date state estimates.

Perception latency is influenced by several stages of the processing pipeline, including image acquisition, object detection, data association, state estimation, and communication with external modules. In vision-based systems, these constraints are affected by camera frame rate, image resolution, exposure time, detector complexity, and available processing resources [9], [14], [27]. A higher image resolution may improve spatial detail, but it can also increase processing time. Conversely, reducing image resolution may improve speed but can degrade detection accuracy, especially for small or fast-moving objects.

In off-board camera-based RC racing setups, real-time performance also depends on the reliability and latency of the image acquisition architecture. When the camera stream is transmitted from an external device to a processing computer, communication delays and frame drops can introduce temporal inconsistencies in the perception pipeline. These issues are relevant for overhead-camera systems, where the camera provides a global view of the track but the processing pipeline must still deliver state estimates at a rate suitable for tracking and control.

Achieving real-time performance therefore requires a trade-off between accuracy, robustness, and computational complexity. Classical image processing methods are generally lightweight, but may be less robust to visual changes. Learning-based detectors can provide more reliable detections under variable appearance conditions, but require more computational resources [27], [28]. Similarly, more complex data association methods or appearance-based tracking approaches may improve identity consistency, but can increase processing latency [14], [30], [32], [33]. For

small-scale autonomous racing platforms, the perception system must balance these factors while maintaining sufficiently frequent and reliable state updates.

2.3 Research Gaps and Positioning

The reviewed literature shows substantial progress in vision-based perception, fiducial marker detection, MOT, and recursive state estimation. Marker-based systems provide explicit identity information in controlled robotic environments [16], [17], [18], [19], while tracking-by-detection approaches such as SORT combine object detections, data association, and Kalman filtering into efficient real-time pipelines [14], [30]. However, these components are not always studied together in the specific context of small-scale autonomous RC racing with an off-board overhead camera.

A first gap concerns the use of external camera-based perception for multi-vehicle racing scenarios. While overhead cameras provide a global view of the track and simplify observation in a shared reference frame, they also introduce challenges related to calibration, spatial resolution, acquisition latency, and visibility constraints. These issues are particularly relevant in small-scale racing environments, where vehicles move quickly and occupy a relatively small region of the image.

A second gap is related to the combination of detection robustness and identity preservation. Fiducial markers such as AprilTags can provide reliable identity cues when visible, but their performance is affected by marker size, viewing angle, motion blur, occlusions, and camera configuration [20]. Learning-based detectors such as YOLO can provide more continuous object detections under visual variability, but they do not inherently solve the identity assignment problem [24], [27], [28]. This creates a need for perception pipelines that can combine continuous visual detection with explicit identity information when available.

A third gap concerns state estimation from visual measurements. In racing scenarios, position measurements obtained from images may be noisy, intermittent, or affected by detection jitter. Directly differentiating these measurements to estimate velocity can lead to unstable results. Recursive filtering methods, such as Kalman filtering, provide a well-established solution for estimating temporally consistent motion states [13], but their practical integration with hybrid visual detection and identity assignment in small-scale multi-vehicle racing setups remains a relevant challenge.

Finally, real-time operation remains a key constraint. Perception systems for autonomous racing must balance detection accuracy, identity consistency, estimation robustness, and computational cost. More complex approaches may improve robustness, but can also increase latency and reduce suitability for real-time control-oriented experiments. These gaps motivate the development of modular perception pipelines that integrate visual detection, identity support, data association, and state estimation while remaining suitable for off-board, small-scale autonomous racing environments. Therefore, to the best of the authors' knowledge, no prior work presents a modular off-board perception pipeline tailored to small-scale RC racing environments. In particular, the reviewed literature does not address an experimentally validated system that jointly integrates YOLO-based

detection, AprilTag-based identity support, multi-stage data association, and Kalman Filter state estimation.

Chapter 3

System Architecture and Methodology

3.1 Overview of the Proposed System

The proposed system consists of an off-board vision-based perception pipeline designed for real-time multi-vehicle tracking in a small-scale autonomous RC racing environment. The main objective of the system is to monitor all vehicles present on the track and estimate their dynamic state over time, including position, velocity, and acceleration. Unlike onboard perception approaches, where each vehicle must carry its own sensing and processing hardware, the proposed architecture relies on a fixed external camera that observes the racing area from above. This allows all vehicles to be monitored within a common visual reference frame and enables centralized processing on an external computer.

The system follows a tracking-by-detection approach. At each iteration, an image frame is acquired from the external camera and processed by the detection modules. Vehicle detections are obtained using a YOLO-based object detector, which provides bounding boxes and confidence scores for the cars visible in the frame. The center of each bounding box is used as the visual position measurement associated with the detected vehicle. In parallel, AprilTags are used as an auxiliary source of identity information. When a tag is successfully detected, its decoded identifier can be used to support or correct the identity assigned to a vehicle track.

The proposed architecture is therefore hybrid, since YOLO and AprilTags provide complementary information. YOLO is used as the main source of continuous vehicle detection, because it can detect the cars based on their visual appearance even when fiducial markers are not visible. AprilTags, on the other hand, provide explicit identity information when the marker is visible and correctly decoded. This combination is useful in the considered racing environment, where motion blur, viewing angle, partial occlusions, and limited image resolution can make marker detections intermittent.

After detection, the system associates the current measurements with existing vehicle tracks. This data association stage is necessary because detections are obtained independently in each frame and do not automatically preserve identity over time. Existing tracks are predicted using a motion model, and incoming detections are matched to these predicted states according to their spatial

compatibility. The tracking module also manages the creation of new tracks, the confirmation of reliable tracks, and the removal of tracks that have not been observed for a predefined number of frames.

For each active track, a Kalman Filter is used to estimate the vehicle state over time. The filter combines the visual position measurements with a dynamic motion model, allowing the system to obtain smoother and more temporally consistent estimates. Since velocity and acceleration are not directly measured by the camera, they are estimated as part of the state vector. The output of the pipeline is a structured set of active vehicle tracks containing identity, position, velocity, acceleration, tracking status, and additional metadata when available.

3.2 System Requirements

The design of the proposed perception pipeline was guided by a set of requirements derived from the autonomous RC racing scenario. Since the system is intended to operate in a dynamic multi-vehicle environment, it must be able to acquire visual information, detect vehicles, maintain their identities over time, and estimate their motion states in real time. At the same time, the system must remain modular enough to support future improvements and integration with higher-level planning and control modules.

The functional requirements define the main capabilities that the system must provide. First, the system must acquire image frames from an external camera observing the racing area. These frames are used as the primary source of information for vehicle detection and state estimation. Second, the system must detect multiple vehicles in each frame, even when the vehicles move at different speeds or appear in different regions of the track. Third, the system must support vehicle identity assignment, using explicit AprilTag information when available and tracking-based identity maintenance when marker detections are temporarily unavailable. Finally, the system must estimate the dynamic state of each active vehicle, including position, velocity, and acceleration.

In addition to these functional requirements, the system must satisfy several non-functional requirements. Real-time operation is essential, since delayed state estimates may become less useful for trajectory analysis or future control applications. Robustness is also required because visual measurements can be affected by motion blur, occlusions, missed detections, and changes in viewing conditions. The architecture must also be modular, allowing individual components such as the camera, data association logic, or state estimator to be modified or improved independently.

These requirements directly influence the architecture described in the following sections. The need for continuous vehicle localization motivates the use of YOLO-based detection, while the requirement for identity support motivates the use of AprilTags as an auxiliary source of information. The need for track continuity and smooth motion estimates motivates the use of data association, track management, and Kalman-based state estimation.

3.3 Off-board Acquisition Architecture

The proposed system follows the off-board perception principle introduced in Chapter 1, but applies it through a specific acquisition architecture designed for small-scale autonomous RC racing. In this architecture, the racing area is observed by a fixed external camera positioned above the track. The camera captures the visual information required for vehicle detection and tracking, while the perception pipeline is executed on an external processing computer.

The use of a fixed overhead camera allows the system to monitor multiple vehicles within a common visual reference frame. This is particularly useful in multi-vehicle tracking, since all vehicles are observed from the same viewpoint and their positions can be estimated relative to the same image coordinate system. Since the vehicles do not need to carry onboard cameras or perception hardware, the architecture also reduces the complexity of the vehicle platform and allows the perception system to be developed independently from the vehicle hardware.

3.3.1 Camera Setup and Image Capture

The camera is positioned above the racing area in order to capture the track from an overhead viewpoint. This placement was selected to maximize the visibility of the vehicles and to provide a common image reference frame for all detections. Since the vehicles move on a planar surface, the overhead viewpoint also simplifies the interpretation of image coordinates and supports the later conversion between pixel measurements and metric quantities.

The camera field of view must include the entire track where vehicles are expected to move. For this reason, the camera position, height, lens field of view, and mounting stability are important aspects of the acquisition setup. If the camera does not cover the full racing area, vehicles may leave the observable region and tracking continuity may be affected. Similarly, if the vehicles occupy only a small number of pixels in the image, vehicle detection and fiducial marker recognition may become less reliable.The final camera position was selected empirically, by testing different heights and orientations until the track area was adequately covered while keeping the camera as close to the track as possible, thus ensuring the best possible image detail possible.

Fig. 3.3 shows the overhead view of the experimental racing area used by the perception system. This viewpoint provides a global representation of the track and allows multiple vehicles to be observed within the same image frame. As a result, vehicle detections, track association, and state estimation can be performed in a common visual reference frame.

3.3.2 Raspberry Pi and Network Transmission

The camera is connected to a Raspberry Pi, which acts as the interface between the camera module and the processing computer. The Raspberry Pi receives the image data from the camera and makes the video stream available through a wired Ethernet connection. This separates the image acquisition hardware from the main perception processing hardware.

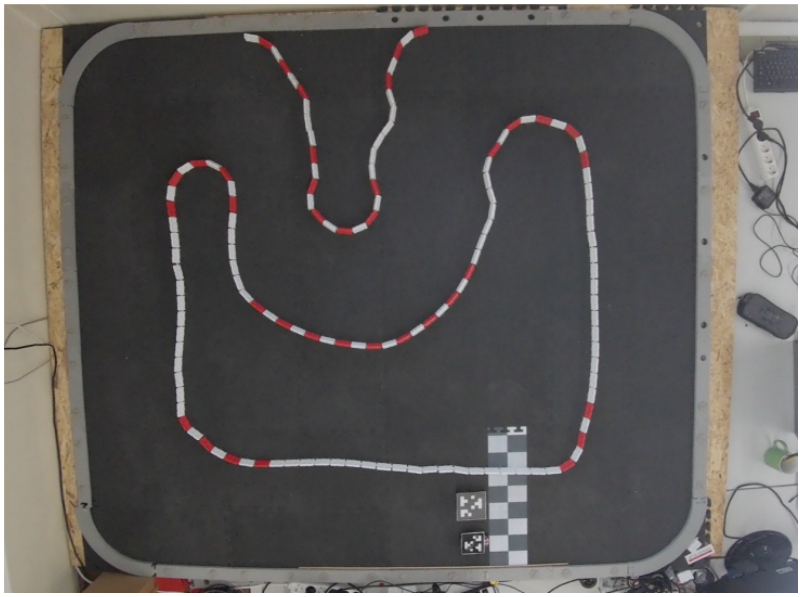

Figure 3.1: Overhead view of the experimental RC racing track captured from the fixed external camera. This viewpoint provides a global view of the racing area and is used as the input image stream for vehicle detection, tracking, and state estimation.

The video stream is transmitted from the Raspberry Pi to the processing computer through an Ethernet switch. A wired network connection is used to provide a stable communication link between the acquisition device and the processing computer. This is important because unstable transmission, frame drops, or excessive buffering can affect the temporal consistency of the perception pipeline.

The adopted acquisition architecture is shown in Fig. 3.3. The image is first captured by the external camera and passed to the Raspberry Pi. The Raspberry Pi then transmits the video stream through the Ethernet switch to the processing computer, where the received frames are used as input to the perception pipeline.

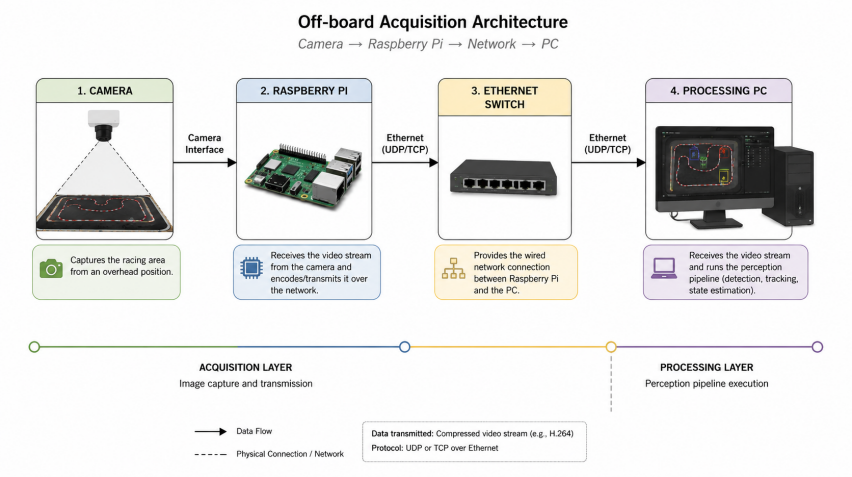

Figure 3.2: Off-board acquisition architecture used in the proposed system. The camera captures the racing area from an overhead position and sends the image stream to a Raspberry Pi. The stream is then transmitted through an Ethernet switch to the processing computer, where the perception pipeline is executed. Figure generated with the support of ChatGPT and manually reviewed by the author.

3.4 Perception Pipeline Methodology

The perception pipeline is responsible for transforming the image frames received from the acquisition architecture into measurements that can be used for multi-object tracking and state estimation. In the proposed system, this stage combines visual vehicle detection with auxiliary identity information. The objective is not only to detect the vehicles present in each frame, but also to generate reliable position measurements and identity cues that can be passed to the tracking module.

The pipeline follows a tracking-by-detection approach. Each frame is processed independently by the detection modules, and the resulting detections are then converted into structured measurements. These measurements include the image position of each detected vehicle, the corresponding bounding box, the detection confidence, and, when available, an AprilTag identifier. Since detections obtained in individual frames do not automatically preserve identity over time, the output of this stage is later processed by the data association and track management modules.

3.4.1 Pipeline Role and Data Flow

The perception pipeline acts as the interface between raw image acquisition and vehicle tracking. The input of this stage is an image frame received from the external camera stream. The output is a set of candidate vehicle measurements that can be associated with existing tracks in the following stage of the system.

The data flow can be described as a sequence of operations. First, the frame is acquired and prepared for processing. Then, the YOLO detector is applied to locate vehicles in the image. In parallel, the AprilTag detector searches for visible fiducial markers and decodes their identifiers

when possible. The information obtained from both detection sources is then organized into a common representation before being passed to the tracking module.

This structure separates the detection problem from the tracking problem. YOLO is responsible for locating vehicles based on visual appearance, while AprilTags provide identity information when a marker is visible. The tracking module, described in the following section, is responsible for maintaining consistent vehicle identities over time. This separation is important because detection, identity support, and temporal tracking have different roles within the complete perception system.

3.4.2 Frame Acquisition and Pre-processing

Each iteration of the pipeline begins with the acquisition of a frame from the video stream received by the processing computer. The frame corresponds to the current visual observation of the racing area and is used as the input for the detection modules. Since the system operates in real time, frames are processed sequentially as they are received.

Before being used by the detection modules, the frame can be validated to ensure that it was correctly acquired. Invalid, empty, or corrupted frames should not be processed, since they could generate false detections or incorrect track updates. When necessary, the frame can also be resized or converted to the image format required by the detection algorithms.

Frame timing is also relevant for the later tracking and state estimation stages. The elapsed time between consecutive frames influences the prediction step of the Kalman Filter, since the motion model depends on the sampling interval. For this reason, the pipeline must preserve the temporal order of the frames and provide consistent frame timing information to the tracking system.

3.4.3 AprilTag-Based Identity Support

AprilTags are used as an auxiliary source of identity information within the perception pipeline. Each marker encodes a unique identifier that can be decoded when the tag is visible in the image. When an AprilTag is successfully detected, its identifier can be associated with the corresponding vehicle and used to reinforce the identity maintained by the tracking system.

The role of AprilTags in the proposed system is therefore complementary to the YOLO-based detection and tracking pipeline. YOLO provides the main visual detections used for vehicle localization, while the tracking module maintains vehicle identities over time through data association and state prediction. The system is able to operate using YOLO detections alone, but AprilTags can increase robustness by providing explicit identity cues when marker detections are available.

This auxiliary role is important because AprilTag detections may be intermittent in dynamic racing scenarios. Their reliability depends on factors such as marker size in the image, camera resolution, viewing angle, motion blur, illumination conditions, and partial occlusions. For this reason, AprilTags are not treated as the only source of identity or localization. Instead, they are used to support identity confirmation and reduce ambiguity in situations where the marker is visible and correctly decoded.

3.4.4 Hybrid Detection and Measurement Strategy

The proposed perception pipeline combines YOLO-based vehicle detection with AprilTag-based identity support. In this strategy, YOLO detections provide the primary measurements used by the system. For each detected vehicle, the bounding box returned by YOLO is used to obtain the visual position measurement, typically through the center of the bounding box. These measurements are then passed to the tracking module, which maintains vehicle identities over time using data association, track management, and state prediction.

AprilTags are integrated as an additional source of information rather than as a replacement for the tracking mechanism. When a marker is detected, its identifier can be used to support the identity assigned to a track. This can help confirm associations, reduce ambiguity between nearby vehicles, and improve robustness in multi-vehicle situations. However, when AprilTags are not detected, the system can continue operating using YOLO detections and the tracking logic.

This hybrid strategy reflects the different strengths of the two methods. YOLO provides continuous appearance-based vehicle detection and supports the operation of the tracking pipeline across frames. AprilTags provide explicit identity information when available, but their detections can be affected by visibility conditions. By combining both sources, the system preserves the ability to track vehicles using YOLO-based detections while benefiting from additional identity cues whenever fiducial markers are visible.

The measurement generated for each detected vehicle therefore contains the information required by the tracking module. At minimum, this includes the position measurement obtained from the YOLO bounding box center. When available, the measurement can also include the bounding box, detector confidence, and decoded AprilTag identifier. This structured representation allows the following tracking stage to use visual position measurements as its main input, while also taking advantage of auxiliary identity information when it is present.

3.4.5 Pipeline Output to the Tracking Module

The output of the perception pipeline is a set of candidate measurements for the current frame. These measurements are not yet final vehicle tracks, because identity must still be maintained over time through data association. Instead, they represent the information extracted from the current frame that can be used to update existing tracks or initialize new ones.

Each candidate measurement can contain the image position of the detected vehicle, the bounding box returned by the YOLO detector, the confidence score of the detection, and an optional AprilTag identifier. The position measurement is the main input used by the state estimator, while the bounding box and identity information can support association decisions. The confidence score can also be used to filter weak detections or prioritize more reliable observations.

By organizing the detection outputs into a structured representation, the perception pipeline provides a clear interface with the multi-object tracking module. The following section describes how these measurements are associated with existing tracks, how new tracks are created, and how track continuity is maintained over time.

3.5 Multi-Object Tracking Methodology

The tracking module is responsible for maintaining consistent vehicle tracks over time. Since detections are produced independently in each frame, a detection in the current frame must be associated with the correct vehicle track from previous frames. This process is necessary to avoid duplicated tracks, fragmented trajectories, and identity switches.

3.5.1 Track Representation

Each vehicle is represented by an individual track. A track stores the current state estimate associated with the vehicle, the latest visual measurement, the most recent bounding box, the number of successful detections, the number of missed frames, and the current tracking status. When available, an AprilTag identifier can also be associated with the track.

The tracking status indicates the reliability and maturity of the track. Newly created tracks can initially be considered tentative. If a track is detected consistently over a predefined number of frames, it becomes confirmed. When a confirmed track is temporarily not matched with a detection, it can be kept alive for a limited number of frames using prediction. If the track remains unmatched for too long, it is removed.

3.5.2 Data Association

Data association is responsible for assigning the detections obtained in the current frame to the active tracks already maintained by the system. Since detections are produced independently at each frame, this step is necessary to preserve track continuity and avoid duplicated tracks, fragmented trajectories, or identity switches. In the proposed system, data association combines motion prediction, spatial compatibility, bounding box overlap, detection confidence, and auxiliary AprilTag identity information.

Before association is performed, all active tracks are predicted forward using their Kalman Filter motion model. This prediction estimates the expected position of each vehicle in the current frame, based on its previous state and on the elapsed time between frames. The predicted position is then used as the reference for comparing existing tracks with the detections obtained from the current image.

As illustrated in Fig. 3.3, the implemented association strategy is organized into multiple stages. AprilTag detections are processed first because they can provide explicit identity information when available. The remaining tracks are then associated with YOLO detections in two visual matching stages, giving priority to high-confidence detections before considering lower-confidence detections. This structure allows the system to use AprilTags as auxiliary identity cues while maintaining YOLO detections as the primary source of visual measurements.

The association process starts with the AprilTag-based stage. If a detected tag identifier is already associated with an existing track, that track is updated directly using the tag measurement. If no track currently has that tag identifier, the system searches for a nearby visual track whose

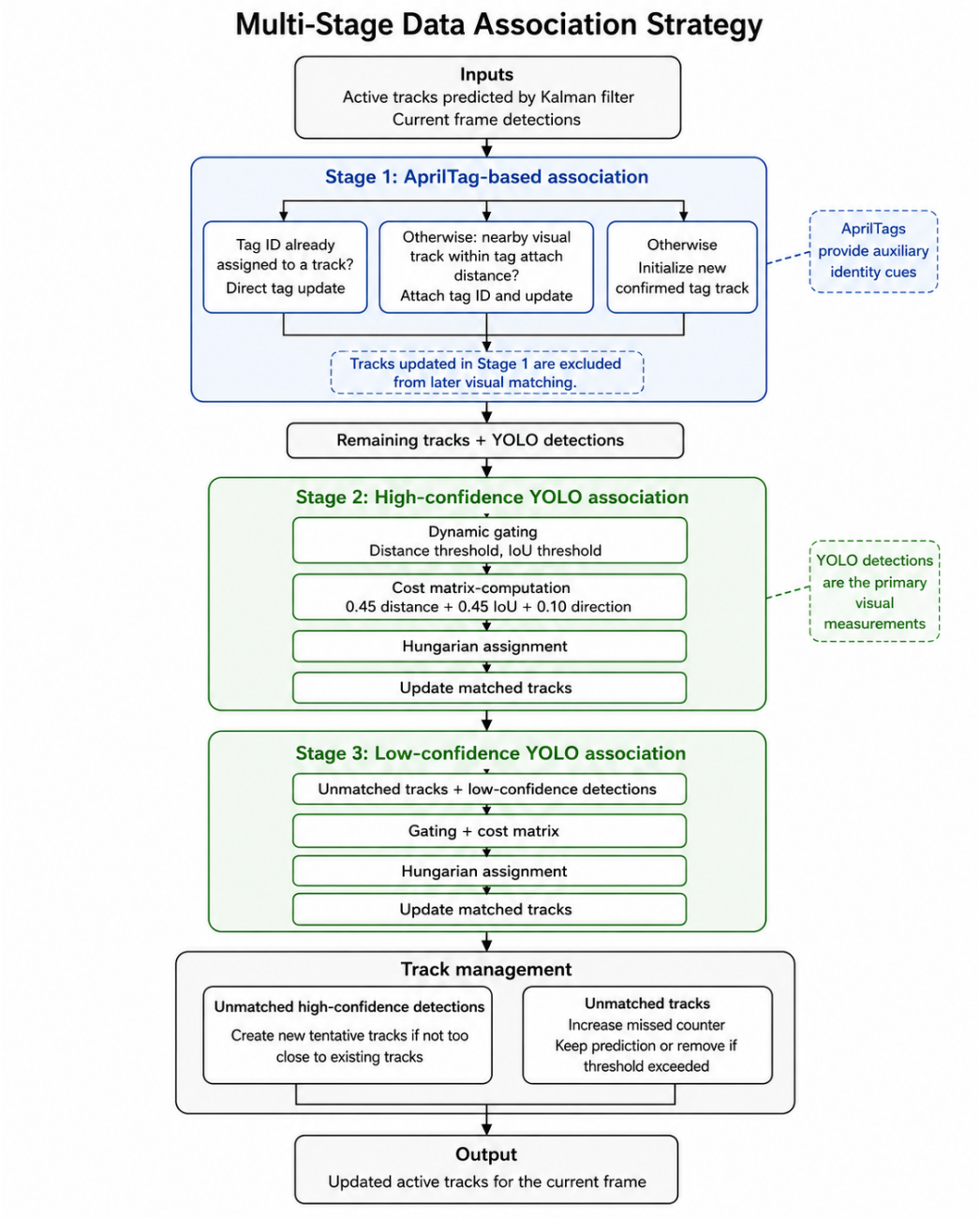

Figure 3.3: Multi-stage data association strategy used in the proposed tracking pipeline. AprilTag detections are processed first as auxiliary identity cues, followed by high-confidence and low-confidence YOLO association stages. The visual association stages use gating, cost matrix computation, and Hungarian assignment to update matched tracks, while unmatched detections and tracks are handled by the track management logic.

predicted position is within a predefined attachment distance. In this case, the tag identifier is assigned to that track and the track is updated using the tag position. If no compatible track is found, a new confirmed track is initialized from the AprilTag measurement.

After the AprilTag-based updates, the remaining tracks are associated with YOLO detections. Tracks that were already updated by AprilTags in the current frame are excluded from the visual matching stage, preventing the same track from being updated twice with conflicting measurements. YOLO detections are divided into high-confidence and low-confidence detections. High-confidence detections are matched first, while lower-confidence detections are only considered in a second association stage for tracks that remained unmatched after the first stage.

For the visual association stage, the system builds a cost matrix between candidate tracks and YOLO detections. Each cost represents the compatibility between one predicted track and one current detection. Associations that are not physically plausible are rejected before the assignment step. First, the distance between the predicted track center and the detected bounding box center must be below a dynamic gating threshold. This threshold increases with the estimated vehicle speed and with the number of consecutive missed frames, allowing more flexibility for faster vehicles or tracks that have not been observed recently. In the implementation, this gate is defined as

$$d_{\max} = d_0 + k_v, |\mathbf{v}|\Delta t + k_m, n_{\text{missed}}, \tag{3.1}$$

where d_0 is the base distance threshold, $|\mathbf{v}|$ is the estimated vehicle speed, Δt is the elapsed time between frames, and n_{missed} is the number of consecutive frames in which the track was not matched.

In addition to the center distance, the predicted bounding box of the track is compared with the detected bounding box using the intersection-over-union (IoU). Associations with insufficient overlap are rejected. This prevents detections that are spatially close but inconsistent with the expected bounding box from being incorrectly assigned to a track.

For valid candidate associations, the final matching cost combines three terms: normalized position distance, bounding box overlap, and motion direction consistency. The position term penalizes detections that are farther from the predicted track center. The IoU term penalizes low overlap between the predicted and detected bounding boxes. The direction term compares the estimated direction of motion from the Kalman Filter with the observed displacement direction between the last measurement and the current detection. The cost used in the implementation can be expressed as

$$C = 0.45, d_{\text{pos}} + 0.45, d_{\text{IoU}} + 0.10, d_{\text{dir}}, \tag{3.2}$$

where d_{pos} is the normalized center distance, $d_{\text{IoU}} = 1 - \text{IoU}$, and d_{dir} represents the disagreement between the estimated motion direction and the observed displacement direction.

Once the cost matrix is computed, the assignment problem is solved using the Hungarian algorithm. This produces a global assignment between tracks and detections, instead of selecting

matches independently using only local nearest-neighbor decisions. This is important in multi-vehicle situations, where two vehicles may be close to each other and a local association rule could produce inconsistent assignments.

The visual association is performed in two stages. First, high-confidence YOLO detections are matched to the available tracks. Then, the tracks that remain unmatched are matched against lower-confidence detections. This strategy gives priority to more reliable detections, while still allowing the system to recover tracks when only weaker visual detections are available.

Detections that remain unmatched after the association stage may initialize new tracks, but only if their confidence is sufficiently high. Before creating a new track, the system also checks whether the detection is too close to an existing predicted track or has significant overlap with it. This additional check reduces the creation of duplicated tracks for vehicles that are already being tracked.

Tracks that are not matched with any detection in the current frame are not immediately deleted. Instead, their missed-frame counter is increased and the Kalman Filter prediction is used to keep estimating their state for a limited number of frames. Tentative tracks are removed more quickly if they are not confirmed by repeated detections, while confirmed tracks are allowed to survive longer temporary detection losses. This behaviour improves robustness to short occlusions, missed YOLO detections, and intermittent AprilTag visibility.

3.5.3 Track Creation, Confirmation, and Removal

Track management defines how vehicle tracks are created, confirmed, maintained, and removed over time. This stage is necessary because detections may be intermittent, noisy, or occasionally incorrect. Without track management, isolated false detections could generate spurious tracks, while short detection failures could cause valid vehicle trajectories to be prematurely interrupted.

When a detection cannot be associated with any existing track, it may initialize a new tentative track. This allows the system to start tracking vehicles that enter the scene or vehicles that were not previously detected. However, a tentative track is not immediately considered reliable. To reduce the effect of isolated false positives, newly created tracks must be detected consistently for a predefined number of frames before being promoted to confirmed tracks.

Confirmed tracks represent vehicles that have been observed with sufficient consistency and are therefore considered reliable by the system. When a confirmed track is matched with a detection in the current frame, its state is updated using the new measurement and its missed-frame counter is reset. This reinforces the continuity of the track and keeps the estimated state aligned with the visual observations.

If an active track is not matched with any detection in a given frame, it is not immediately removed. Instead, the Kalman Filter prediction is used to propagate the track state for a limited number of frames. During this period, the track remains active but is considered temporarily unmatched. This behaviour allows the system to handle short-term missed detections, brief occlusions, motion blur, or temporary loss of AprilTag visibility without breaking the trajectory.

The removal criterion depends on the maturity of the track. Tentative tracks are removed more quickly when they fail to receive repeated detections, since they may correspond to false positives. Confirmed tracks are allowed to survive for a larger number of missed frames, because they are more likely to correspond to real vehicles that were temporarily not detected. If the number of consecutive missed frames exceeds the allowed threshold, the track is removed from the active set.

This strategy provides a balance between stability and responsiveness. Requiring repeated detections before confirming a track reduces false track creation, while keeping unmatched confirmed tracks alive for a short period improves robustness to intermittent detections. At the same time, removing tracks that remain unmatched for too long prevents the system from maintaining outdated or incorrect vehicle states.

3.6 State Estimation

State estimation is used to obtain smooth and temporally consistent vehicle states from noisy visual measurements. In the proposed system, the perception pipeline directly provides position measurements in the image plane, obtained from the center of YOLO bounding boxes or, when available, from AprilTag detections. However, velocity and acceleration are not directly measured and must be estimated over time.

Directly differentiating position measurements would make the estimates highly sensitive to image noise, missed detections, and frame-rate variations. For this reason, each active track is associated with an independent Kalman Filter. The filter combines visual measurements with a motion model, allowing the system to estimate position, velocity, and acceleration while maintaining short-term predictions when detections are temporarily unavailable.

3.6.1 Kalman Filter State Model

The vehicle state is represented using a constant-acceleration model in two dimensions. The state vector is defined as

$$\mathbf{x} = \begin{bmatrix} x & y & v_x & v_y & a_x & a_y \end{bmatrix}^T, \tag{3.3}$$

where x and y are the vehicle position components, v_x and v_y are the velocity components, and a_x and a_y are the acceleration components. In this model, the vehicle state contains not only the measured position but also the unobserved velocity and acceleration terms that are inferred over time by the filter.

The position components are obtained from visual measurements in the image plane. In the case of YOLO detections, the measured position corresponds to the center of the detected bounding box. When AprilTag detections are available, the marker position can also provide a position measurement associated with the vehicle. In both cases, the measurement is expressed in image coordinates before any metric conversion is applied.

The use of a constant-acceleration model does not imply that the vehicle acceleration is truly constant during the full experiment. Instead, it provides a local approximation of the vehicle motion between consecutive frames. Since the time interval between frames is small, this model is suitable for predicting short-term motion while allowing the filter to update the state whenever new measurements are available.

3.6.2 Prediction Step

The prediction step estimates the vehicle state at the current frame before the new visual measurement is applied. This is done using the previous state estimate and the elapsed time Δt between consecutive frames. The value of Δt is important because the amount of predicted displacement depends on how much time has passed since the last update.

The predicted state is computed as

$$\mathbf{x}_k = \mathbf{F}\mathbf{x}_{k-1} + \mathbf{w}_k, \tag{3.4}$$

where $\mathbf{x}_k$ is the predicted state at frame k, $\mathbf{x}_{k-1}$ is the previous state, $\mathbf{F}$ is the state transition matrix, and $\mathbf{w}_k$ represents process noise. The process noise accounts for the fact that the motion model is only an approximation of the real vehicle dynamics.

For a constant-acceleration model, the state transition matrix is given by

$$\mathbf{F} = \begin{bmatrix} 1 & 0 & \Delta t & 0 & \frac{1}{2}\Delta t^2 & 0 \\ 0 & 1 & 0 & \Delta t & 0 & \frac{1}{2}\Delta t^2 \\ 0 & 0 & 1 & 0 & \Delta t & 0 \\ 0 & 0 & 0 & 1 & 0 & \Delta t \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{3.5}$$

The first two rows update the predicted position using the previous position, velocity, and acceleration. For example, the horizontal position is propagated according to the kinematic relation

$$x_k = x_{k-1} + v_{x,k-1}\Delta t + \frac{1}{2}a_{x,k-1}\Delta t^2. \tag{3.6}$$

The third and fourth rows update the velocity components using the previous acceleration, while the last two rows assume that the acceleration remains constant between consecutive frames. This formulation allows the filter to estimate position, velocity, and acceleration in a unified state representation.

The prediction step is especially important when a vehicle is temporarily not detected. In that case, the system can still estimate the expected position of the vehicle based on its previous motion. This predicted position is also used during data association, where it helps determine which current detection is most compatible with each existing track.

3.6.3 Measurement Update

The vision system directly measures only the vehicle position in the image plane. Therefore, the measurement vector is defined as

$$\mathbf{z}_k = \begin{bmatrix} x_m & y_m \end{bmatrix}^T. \tag{3.7}$$

The measurement model relates the observed position to the estimated state through

$$\mathbf{z}_k = \mathbf{H}\mathbf{x}_k + \mathbf{v}_k, \tag{3.8}$$

where $\mathbf{H}$ is the measurement matrix and $\mathbf{v}_k$ represents measurement noise. Since only the position components of the state are directly observed, the measurement matrix is defined as

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}. \tag{3.9}$$

This matrix selects the x and y components from the full state vector and maps them to the measurement space. The velocity and acceleration components are not directly measured, but they are updated indirectly through the correction of the full state estimate. As new position measurements are received over time, the filter adjusts the velocity and acceleration estimates so that the predicted motion remains consistent with the observed vehicle trajectory.

The update step is applied when a detection has been associated with a track. The predicted state is corrected using the difference between the predicted position and the measured position. This correction reduces the uncertainty of the estimate and keeps the track aligned with the visual observations. When no measurement is available for a given track, the update step is skipped and the system relies temporarily on the prediction step.

3.6.4 Uncertainty and Noise Modelling

The Kalman Filter also maintains an uncertainty estimate associated with each state. This uncertainty represents how confident the filter is in the estimated position, velocity, and acceleration. During prediction, uncertainty generally increases because the state is propagated using a motion model that may not perfectly describe the real vehicle movement. During measurement update, uncertainty decreases because the filter receives new visual information.

The process noise describes the uncertainty associated with the motion model. In practice, this accounts for changes in vehicle motion that are not fully captured by the constant-acceleration assumption, such as sudden steering changes, acceleration changes, or small modelling errors. The measurement noise describes the uncertainty associated with the visual observations. This includes errors from bounding box localization, image resolution, motion blur, detection instability, and marker localization errors.

The balance between process noise and measurement noise determines how much the filter trusts the motion model compared with the incoming measurements. If measurement noise is

considered high, the filter gives more weight to the predicted motion and produces smoother estimates. If measurement noise is considered low, the filter follows the visual measurements more closely. This balance is important for obtaining stable velocity and acceleration estimates without making the system too slow to react to real vehicle motion.

3.6.5 Metric Position, Velocity, and Acceleration

The detections produced by the vision system are initially obtained in image coordinates. Therefore, the state estimated by the Kalman Filter is first expressed in pixel-based units. The position components are represented in pixels, the velocity components in pixels per second, and the acceleration components in pixels per second squared.

To interpret vehicle motion in physical units, the estimated state can be converted into a metric reference frame using the calibration procedure defined for the track. If s represents the scale factor between image coordinates and metric coordinates, position can be converted from pixels to meters by applying this factor to the image displacement. The same scale factor can also be applied to velocity and acceleration estimates, converting them from pixels per second and pixels per second squared into meters per second and meters per second squared.

This metric conversion is important for trajectory analysis and for future integration with planning or control modules. Pixel coordinates are useful for image processing and visual tracking, but physical units are more meaningful when analysing vehicle motion. For example, velocity expressed in meters per second can be compared across different camera resolutions or experimental setups, while pixel-based velocity depends directly on the image scale.

In the proposed system, the Kalman Filter provides a temporally consistent estimate of the vehicle state, while the calibration procedure allows that estimate to be interpreted in metric terms. Together, these two components transform noisy visual detections into structured motion information that can be used for monitoring, analysis, and future autonomous racing modules.

3.7 Output Data Representation

The output of the perception pipeline is a structured set of active vehicle tracks. For each active track, the system provides the track identifier, optional AprilTag identifier, most recent visual measurement, estimated position, estimated velocity, estimated acceleration, tracking status, and additional information related to missed detections or track confidence.

These outputs can be used for monitoring, debugging, trajectory analysis, and future integration with higher-level modules. Since the system produces structured state estimates rather than only visual detections, it can support applications that require temporally consistent vehicle information.

3.8 Chapter Summary

This chapter presented the proposed system architecture and methodology for off-board multi-vehicle perception in a small-scale autonomous RC racing environment. The architecture separates

image acquisition from perception processing, using a fixed overhead camera connected to a Raspberry Pi and a processing computer that executes the perception pipeline. The methodology combines YOLO-based detection, AprilTag-based identity support, data association, track management, and Kalman-based state estimation. The next chapter describes the implementation details of the proposed system.

Chapter 4

Implementation

This chapter describes the implementation of the perception system introduced in Chapter 3. The implementation includes the software structure, camera stream integration, detection modules, tracking logic, Kalman-based state estimation, visualization, and logging components. The system was designed to be modular, allowing individual components to be tested and modified independently.

4.1 Software Architecture

The implementation follows a modular structure, where each main function of the perception pipeline is handled by a specific component. This organization improves readability and makes it easier to debug or replace individual modules without changing the full system. The main components of the implementation are the camera input module, the detection modules, the tracking module, the state estimation module, the visualization module, and the logging and output module.

The implemented pipeline is organized as a sequence of software modules, as shown in Fig. 4.1. Each module is responsible for a specific stage of the perception process, from frame acquisition to detection, tracking, state estimation, and output generation. This modular organization makes the system easier to test, debug, and adapt, since individual components can be modified without changing the full pipeline.

The camera input module receives frames from the video stream. The detection modules process each frame using YOLO and AprilTag detection. YOLO provides vehicle bounding boxes and confidence scores, while AprilTags provide auxiliary identity information when visible. The tracking module associates detections with existing tracks. Each active track is linked to a Kalman Filter, which estimates position, velocity, and acceleration over time. Finally, the visualization and logging components are used to inspect system behaviour and persist data for later analysis.

This sequential structure is suitable for real-time operation because each frame is processed independently while the tracking module preserves temporal consistency across frames.

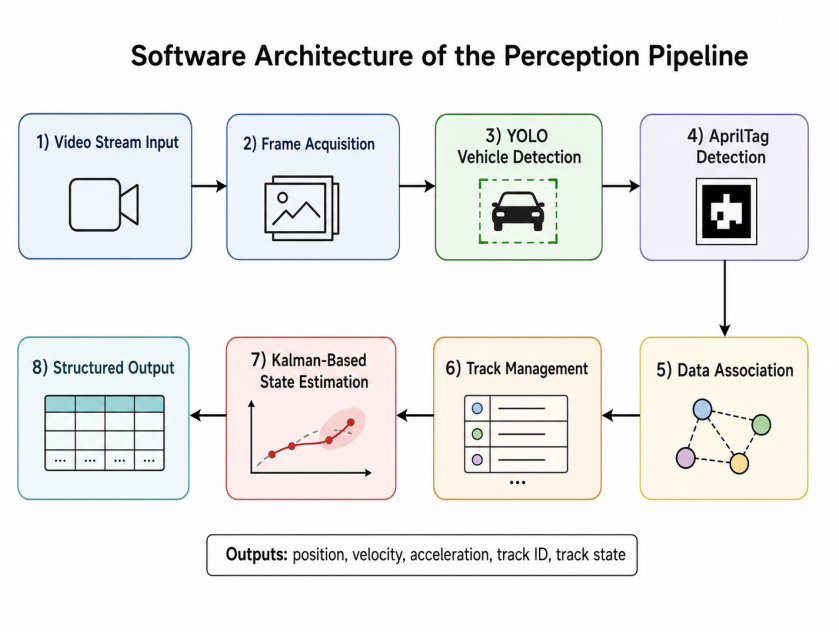

Figure 4.1: Software architecture of the implemented perception pipeline. The system receives video frames, applies YOLO and AprilTag detection, associates detections with active tracks, performs track management, estimates vehicle states using a Kalman-based approach, and generates structured outputs containing position, velocity, acceleration, track identity, and track state. Figure generated with the support of ChatGPT and manually reviewed by the author.

4.2 Development Environment

4.2.1 Runtime, Hardware, and Dependencies

To ensure reproducibility and enable fair performance interpretation, the following environment was used:

- Operating system: Windows 11 Pro, 64-bit;
- CPU: Intel Core i7-12700K; RAM: 32 GB;
- GPU: NVIDIA GeForce RTX 3060, 12 GB;
- Python 3.10.13;
- OpenCV 4.8;
- Ultralytics 8.x, for YOLO integration;
- NumPy 1.26.x;
- SciPy 1.11.x, for Hungarian assignment;
- `pupil_apriltags` 1.0.x.

Dependencies are frozen through a `requirements.txt` file. The YOLO model artifact, stored as `best.pt`, and the repository version used for experiments are referenced in the results chapter.

4.2.2 Libraries

OpenCV is used for video acquisition, frame manipulation, visualization, and image display. YOLO is integrated through the Ultralytics framework to load a custom-trained model. AprilTag detection uses a library capable of decoding marker identifiers from grayscale images. NumPy supports numerical operations and matrices, while the Hungarian algorithm is provided through SciPy.

Table 4.1 summarizes the main tools and libraries used.

Table 4.1: Main tools and libraries used in the implementation.

Tool / Library	**Role in the implementation**
Python	Main programming language for the perception pipeline.
OpenCV	Video acquisition, frame handling, visualization, and image processing.
Ultralytics YOLO	Custom trained object detection model integration.
AprilTag detector	Fiducial detection and tag ID decoding.
NumPy	Numerical operations, state vectors, matrices, and geometry.
SciPy	Global assignment between tracks and detections using the Hungarian algorithm.
CSV/JSON logging	Persistent storage of detections, tracks, and state estimates.

The perception pipeline runs on an external processing computer receiving the camera stream over Ethernet. This allows the more computationally demanding operations, such as YOLO inference and multi-object tracking, to be executed outside the acquisition device.

4.3 Camera Stream Integration

The camera is connected to a Raspberry Pi that streams video through a wired Ethernet connection to the processing computer. On the processing side, OpenCV's `VideoCapture` interface is used to open the stream. The implementation configures the expected image width, height, frame rate, and video format when supported by the selected backend.

The acquisition component reads and validates frames sequentially. If a frame is invalid, the system avoids updating the tracking pipeline with unreliable information. A watchdog monitors

consecutive acquisition failures and reinitializes the capture when the number of failed frames exceeds a configurable threshold. Resources are also closed cleanly when the execution is interrupted or when an acquisition error occurs.

The elapsed time between frames, represented by Δt, is computed from a monotonic clock and passed to the Kalman prediction step. Using the measured value of Δt instead of assuming a fixed frame rate improves robustness to small timing variations in the stream or in the processing loop.

A simplified version of the acquisition logic is:

Step 1: open the network video stream with the configured backend and video format;

Step 2: warm up the capture and read the current frame;

Step 3: validate the frame and detect acquisition failures;

Step 4: compute Δt since the previous frame;

Step 5: pass the frame and timing information to the perception pipeline.

4.4 Detection Modules

The detection stage uses two complementary modules: YOLO-based vehicle detection and AprilTag detection. YOLO provides the primary vehicle localization measurements, while AprilTags provide auxiliary identity information when visible. Both modules process the same input frame, but their outputs are used for different purposes in the tracking pipeline.

4.4.1 YOLO Model Integration

The YOLO detector was trained on a custom dataset collected from the experimental track. The dataset contains frames where the vehicle appears in different positions, orientations, lighting conditions, and motion situations. The model was trained for a single class, corresponding to the RC vehicle.

During execution, each incoming frame is passed to the YOLO model. The detector returns a set of bounding boxes, each associated with a confidence score. The confidence score is used to evaluate the reliability of each detection and to separate detections into different confidence levels for the association stage.

For each valid detection, the bounding box and its center are extracted. The center of the box is used as the image-plane position measurement for tracking. Detections with invalid bounding box dimensions are discarded before entering the tracking module.

The detections returned by the YOLO model are divided into two groups according to their confidence value:

Group 1: high-confidence detections, with confidence greater than or equal to the main detection threshold;

Group 2: low-confidence detections, with confidence below the main threshold but still above a lower recovery threshold.

This separation supports the two-stage association strategy described in Chapter 3. High-confidence detections are used first because they are more reliable. Lower-confidence detections are considered later, only for tracks that were not matched during the first association stage.

A region-of-interest filter is also applied before association. This prevents detections outside the valid racing area from being used by the tracker. Instead of relying only on whether the bounding box center lies inside the region, the implementation considers the overlap between the detected box and the valid region. This is more robust near the borders, where a valid vehicle detection may partially overlap the track boundary.

4.4.2 AprilTag Detection Implementation

AprilTag detection is implemented as an auxiliary identity support module. Before running the marker detector, the input frame is converted to grayscale. The detector searches the image for visible tags and decodes their identifiers when possible.

For each detected AprilTag, the implementation extracts the marker identifier and its image position. The marker position is computed from the detected tag geometry, using the marker center as the position measurement. The decoded identifier provides explicit information about the physical vehicle associated with the marker, as long as the marker is visible and correctly detected.

The AprilTag output contains:

- the decoded tag identifier;
- the marker center in image coordinates;
- the marker corners, when available;
- the detection source.

AprilTags are not treated as the only source of tracking information. Their role is to reinforce identity assignment when they are available. Since marker detections can fail due to motion blur, occlusions, viewing angle, marker size, or resolution limitations, the system remains capable of operating using YOLO detections and tracking logic alone.

4.4.3 Detection Output Representation

The outputs of the YOLO and AprilTag modules are converted into a common representation before entering the tracking stage. This representation allows the tracking module to process detections in a consistent way, even though the two detectors provide different types of information.

Each detection can include the following fields:

- detection source;
- bounding box, when available;
- center position;
- confidence score, for YOLO detections;
- tag identifier, for AprilTag detections;
- additional metadata used for debugging or logging.

This common structure separates vehicle localization from identity support. YOLO detections are used as the primary source of position measurements, while AprilTag detections are used to attach or confirm identity information when possible.

4.5 Tracking and State Estimation Implementation

The tracking implementation is responsible for maintaining vehicle trajectories across frames. Since detections are generated independently for each frame, the system must determine whether a detection corresponds to an existing vehicle track, a new vehicle, or a false detection. The implementation combines track data structures, data association, track management, and Kalman filtering.

4.5.1 Track Data Structures

Each vehicle is represented internally by a track object. A track stores all the information required to maintain the identity and state of one vehicle over time. The values stored by each track include visual information, tracking status, missed detection counters, optional marker identity, and the state estimator associated with that vehicle.

Each track stores:

- a unique track identifier;
- an optional AprilTag identifier;
- the latest bounding box;
- the latest position measurement;

- the current Kalman Filter state, represented by $[x, y, v_x, v_y, a_x, a_y]$;
- the number of successful detections;
- the number of consecutive missed frames;
- the current track status;
- the source of the latest update.

The track status is used to distinguish between tentative and confirmed tracks. Tentative tracks correspond to newly initialized tracks that have not yet been observed consistently. Confirmed tracks correspond to vehicles that have been detected reliably over multiple frames. This distinction reduces the probability of creating permanent tracks from isolated false detections.

4.5.2 Data Association Implementation

The implementation of data association follows the multi-stage strategy described in Section 3.5.2. The process starts by predicting all active tracks forward using their Kalman Filter motion model. This predicted state provides an expected position for each vehicle in the current frame.

AprilTag detections are processed first because they can provide explicit identity information. If a detected tag identifier is already associated with an active track, that track can be updated using the tag measurement. If the tag identifier is not yet associated with any active track, the implementation searches for a nearby visual track to which the tag can be attached. If no compatible track is found, a new confirmed track can be initialized from the tag measurement.

After the AprilTag-based stage, the remaining tracks are matched with YOLO detections. The implementation gives priority to high-confidence YOLO detections. Lower-confidence detections are considered only in a later stage for tracks that remained unmatched. This two-stage visual matching strategy allows the system to prioritize reliable detections while still preserving tracks when only weaker detections are available.

For the YOLO-based association stage, the implementation builds a cost matrix between candidate tracks and detections. Each entry in the matrix represents the compatibility between a predicted track and a current detection. Candidate matches can be rejected before assignment if they violate gating conditions. The gating threshold depends on the predicted position, estimated velocity, elapsed time, and number of missed frames. This makes the gate more flexible for faster vehicles or tracks that have not been observed recently.

For valid candidate pairs, the matching cost combines position distance, bounding box overlap, and motion direction consistency. The position distance measures how far the detection is from the predicted track center. The bounding box overlap is evaluated using the intersection-over-union between the predicted and detected boxes. The direction term compares the estimated direction of motion with the observed displacement direction. These terms are combined into a single cost value, which is then used by the assignment solver.

The assignment problem is solved globally using the Hungarian algorithm. This avoids making independent local decisions for each track and reduces ambiguity in cases where multiple vehicles are close to one another. After assignment, matched tracks are updated with their associated detections, unmatched detections may initialize new tracks, and unmatched tracks are handled by the track management logic.

Algorithm 1 Multi-stage data association

```
Predict all active tracks
Process AprilTag detections
for each detected AprilTag do
    if the tag ID is already assigned to a track then
        Update the corresponding track
    else if a nearby compatible visual track exists then
        Attach the tag ID to that track and update it
    else
        Initialize a new confirmed track from the tag detection
    end if
end for
Exclude tracks already updated by AprilTags
Match remaining tracks with high-confidence YOLO detections
Match still-unmatched tracks with low-confidence YOLO detections
Initialize tentative tracks from reliable unmatched YOLO detections
Increase missed-frame counters for unmatched tracks
Remove tracks that exceed the allowed missed-frame limit
```

4.5.3 Kalman Filter Implementation and Noise Modelling

Each active track runs an independent Kalman Filter using a constant-acceleration model in two dimensions. The state vector is represented as $[x, y, v_x, v_y, a_x, a_y]$. The transition matrix is updated using the measured value of Δt, allowing the prediction step to reflect the actual timing of the video stream.

The filter supports two main operations. First, it predicts the expected vehicle state before association. This prediction provides an estimated position that can be used by the data association stage. Second, when a detection is assigned to a track, the filter performs a measurement update using the detected position.

The implementation allows different measurement noise values to be used depending on the detection source. YOLO measurements and AprilTag measurements may have different levels of uncertainty, since bounding box centers and marker centers are affected by different types of noise. This distinction allows the filter to weigh measurements according to their expected reliability.

During prolonged missed detections, uncertainty increases because the track is being propagated using prediction only. This reflects the fact that the estimated state becomes less reliable when no visual correction is available. Once a detection is matched again, the measurement update reduces uncertainty and realigns the estimated state with the visual observation.

4.5.4 Track Management Implementation

Track management controls the life cycle of each track. When a detection cannot be associated with an existing track, the implementation may initialize a new tentative track. This track must receive repeated detections before it is considered confirmed. The confirmation step prevents isolated false detections from being treated as reliable vehicles.

When a confirmed track is matched with a detection, its measurement is updated, its Kalman Filter is corrected, and its missed-frame counter is reset. When a track is not matched in a given frame, its missed-frame counter is increased. The track can remain active for a limited number of missed frames, during which its state is propagated using Kalman prediction.

Tentative tracks are removed more quickly than confirmed tracks because they are less reliable. Confirmed tracks are allowed to survive longer detection gaps, since they are more likely to correspond to real vehicles. If the number of missed frames exceeds the configured threshold, the track is removed from the active set. This strategy balances robustness to short detection failures with the need to remove outdated tracks.

4.6 Coordinate Conversion and Calibration Implementation

The perception pipeline initially operates in image coordinates. Vehicle detections, bounding box centers, AprilTag positions, and Kalman Filter states are represented in pixels. To obtain physically meaningful quantities, these values must be converted into metric coordinates using the calibration procedure described in Chapter 3.

The implementation defines reference points in the image and their corresponding positions in the physical track plane. These correspondences are used to obtain the transformation between image coordinates and metric coordinates. When enough non-collinear reference points are available, a planar homography can be used to map image points to the track plane.

A simpler scale-only conversion can also be used as an approximation when the camera is close to a top-down view and the region of interest is limited. In this case, a scale factor $s = d_m/d_{px}$ converts pixel displacements into metric distances. However, this approximation is less accurate near the image borders, where perspective effects become more visible.

Velocity and acceleration are converted consistently with the position mapping. Pixel velocities can be converted into meters per second, and pixel accelerations can be converted into meters per second squared. This makes the output of the system easier to interpret and more suitable for future integration with trajectory planning or control modules.

4.7 Visualization and Debugging

A real-time visualization module was implemented to support development, debugging, and qualitative evaluation of the perception pipeline. The visualization overlays information directly

on the video frame, allowing the behaviour of the detector and tracker to be inspected while the system is running.

The displayed information can include detected bounding boxes, track identifiers, AprilTag identifiers, estimated positions, and state information such as velocity or acceleration. Visual feedback is particularly useful during development because it allows incorrect associations, duplicated tracks, missed detections, and unstable estimates to be identified quickly.

The visualization module also helps verify that the camera stream is being received correctly and that the detection modules are operating as expected. Since the system processes frames in real time, visual debugging provides immediate feedback about the effect of parameter changes, such as confidence thresholds, gating distances, or track removal limits.

Although visualization is useful for debugging, it can also increase processing load. For this reason, the implementation separates the tracking logic from the display logic, allowing visualization to be reduced or disabled when performance measurements are required.

4.8 Output Generation and Logging

The implementation includes logging functionality to store relevant information from the perception pipeline. Logging is important because it allows the behaviour of the system to be analysed after each experiment, instead of relying only on real-time visualization.

For each frame or active track, the logged data can include:

- frame index or timestamp;
- track identifier;
- optional AprilTag identifier;
- detection source;
- bounding box information;
- detection confidence;
- estimated position;
- estimated velocity;
- estimated acceleration;
- number of missed frames;
- track status.

These logs support later analysis of detection behaviour, tracking continuity, AprilTag availability, and Kalman Filter performance. They also provide the data required to generate plots and

tables in the results chapter. For example, the logs can be used to compare visual detections with state estimates, evaluate the number of AprilTag-supported updates, or inspect periods where tracks were maintained through prediction.

The output of the system is therefore both visual and structured. The visual output supports real-time inspection, while the structured logs support quantitative analysis and reproducibility.

An optional UDP output can also be enabled when downstream communication is required. In that case, the system sends a lightweight JSON message containing the current timestamp, frame index, and active track states. This output is stateless and can tolerate occasional packet loss, which makes it suitable for real-time monitoring or future integration with other modules.

4.9 Real-Time Execution Loop

The complete implementation runs as a continuous real-time loop. Each iteration processes one frame from the video stream and updates the active vehicle tracks. The main loop is responsible for coordinating acquisition, detection, association, estimation, visualization, and logging.

The execution sequence can be summarized as follows:

Algorithm 2 Real-time perception loop

while the video stream is active **do**
    Acquire frame from the camera stream
    Validate frame and compute elapsed time Δt
    Run YOLO vehicle detection
    Run AprilTag detection
    Predict all active tracks using the Kalman Filter
    Process AprilTag-based identity updates
    Associate high-confidence YOLO detections with remaining tracks
    Associate low-confidence YOLO detections with unmatched tracks
    Initialize new tentative tracks from reliable unmatched detections
    Update matched tracks and propagate unmatched tracks
    Remove tracks that exceed the missed-frame limit
    Convert states to metric units when required
    Display and log the active track states
end while

This loop structure ensures that all modules are executed in a consistent order. The Kalman prediction step is performed before association so that each track has an expected position in the current frame. Detection and association are then used to correct the predicted states. Finally, the visualization and logging components record the output of the current iteration.

Because the loop runs continuously, the processing time of each component affects the overall update rate. The implementation therefore requires a balance between detection accuracy, tracking robustness, visualization detail, and real-time performance.

4.10 Chapter Summary

This chapter described the implementation of the proposed off-board perception pipeline. The system was implemented in Python using OpenCV for video acquisition and visualization, YOLO for vehicle detection, AprilTag detection for auxiliary identity support, and Kalman filtering for state estimation. The implementation follows a modular structure that separates camera input, detection, tracking, state estimation, visualization, and logging.

The chapter also described how the system receives the camera stream, processes each frame, associates detections with tracks, estimates vehicle states, and stores outputs for later analysis. The next chapter presents the experimental setup and evaluation methodology used to assess the behaviour of the implemented system.

Chapter 5

Experimental Setup and Evaluation Methodology

This chapter describes the experimental setup and the evaluation methodology used to assess the implemented off-board perception pipeline. While Chapter 4 focused on the software implementation, this chapter defines the conditions under which the system was tested, the data collected during the experiments, and the indicators used to evaluate detection, tracking, state estimation, and real-time performance.

The goal of the evaluation is to verify whether the implemented system can detect and track RC vehicles from an external camera perspective, maintain consistent vehicle tracks over time, estimate motion-related quantities, and operate at a frame rate suitable for real-time monitoring. Since a complete frame-by-frame annotated ground truth was not available for all sequences, the evaluation combines log-based quantitative indicators with qualitative inspection of the visual output.

5.1 Experimental Setup

The experiments were conducted on a small-scale RC racing track observed by a fixed external camera. The setup was designed to reproduce the conditions required for off-board perception in autonomous racing, where the environment is monitored from a global viewpoint instead of relying on cameras mounted on the vehicles.

The experimental setup includes the physical track, the RC vehicles, the camera acquisition system, the network transmission setup, and the processing computer running the perception pipeline. The camera was positioned above the track in order to capture a global view of the racing area. This viewpoint allows the system to observe the vehicle motion across the track and provides the image stream used for detection, tracking, and state estimation.

5.1.1 Track Environment

The test environment consists of a small-scale racing area, measuring approximately 1.95 m $\times$ 2.25 m, where RC vehicles can move along a predefined track. The track was used to evaluate the

behaviour of the perception pipeline under different motion conditions, including straight motion, curved motion, and changes in vehicle speed.

The camera position was selected empirically, assuring track coverage while maximizing image detail for vehicle detection and AprilTag recognition. A wider view allows more of the track to be observed, but it reduces the number of image pixels occupied by each vehicle. This trade-off is relevant because both YOLO detections and AprilTag decoding depend on the visual size and clarity of the vehicle in the image.

Figure 5.1 shows the experimental track used for data collection and system evaluation.

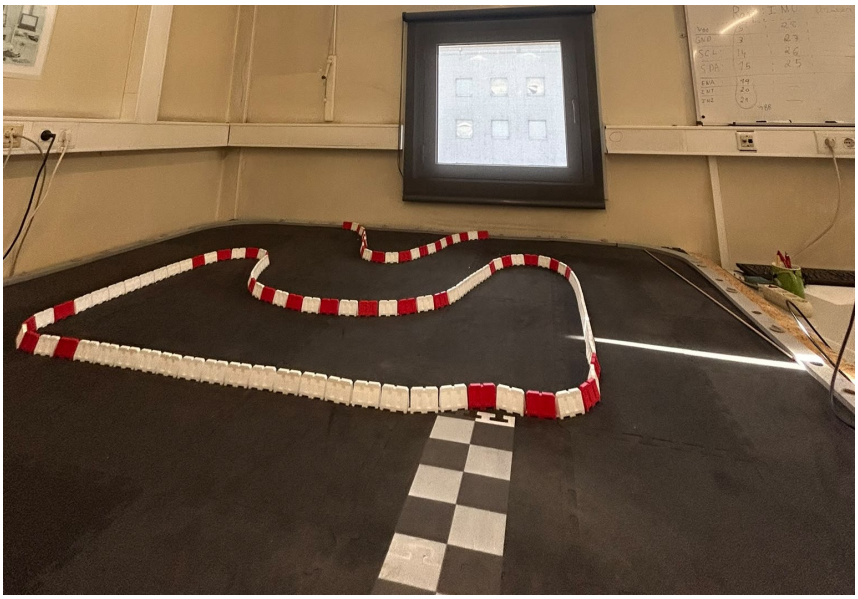

Figure 5.1: Experimental RC racing track used for data collection and evaluation. The fixed external camera provides an off-board view of the racing area, which is used as input to the perception pipeline.

5.1.2 Camera and Acquisition Configuration

The image stream was acquired using a fixed camera connected to a Raspberry Pi. The Raspberry Pi transmitted the video stream to the processing computer through a wired Ethernet connection. This configuration separates image acquisition from the more computationally demanding perception tasks, which are executed on the processing computer.

The processing computer receives the stream using OpenCV and processes the incoming frames sequentially. For each frame, the system validates the acquired image and computes the elapsed time between consecutive frames. This time interval is used by the Kalman Filter prediction step, since the state transition model depends on the sampling time.

The acquisition setup can be summarized as follows:

- fixed external camera positioned above the track;
- Raspberry Pi used for image acquisition and video streaming;

- wired Ethernet connection between the acquisition device and the processing computer;
- OpenCV-based frame reception on the processing computer;
- timestamping and elapsed-time computation for each processed frame.

Figure 5.2 illustrates the acquisition setup used during the experiments.

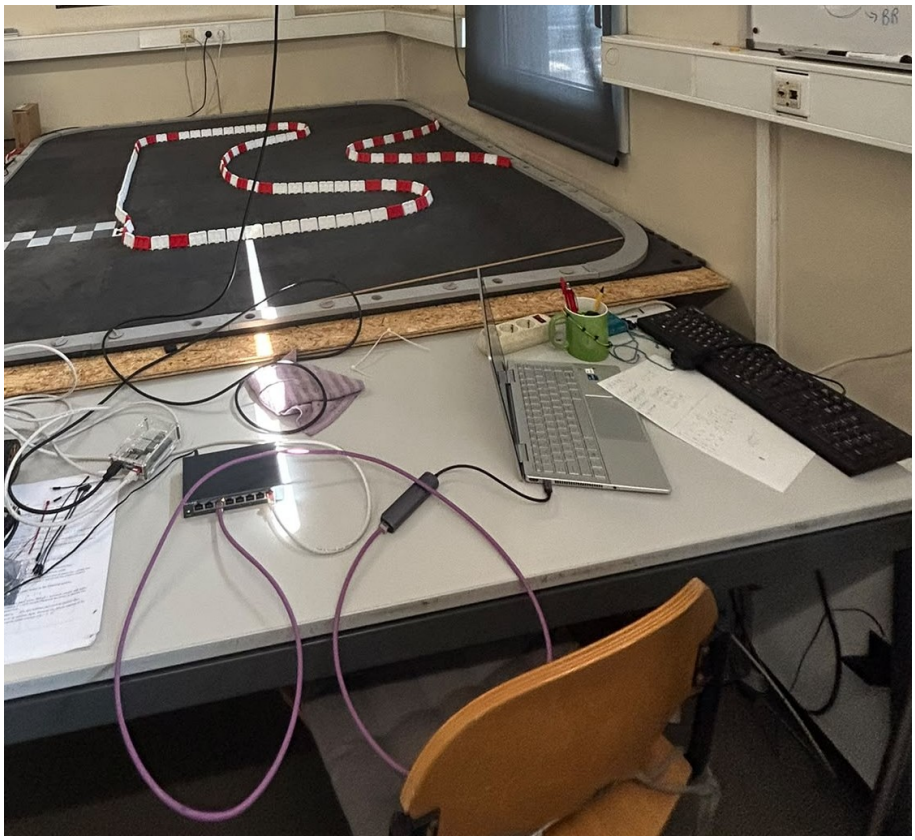

Figure 5.2: Experimental acquisition setup used to transmit the image stream from the external camera to the processing computer. The Raspberry Pi is responsible for camera acquisition and network streaming, while the perception pipeline runs on the external processing computer.

5.1.3 Vehicles, Markers, and Detection Model

The experiments were performed using RC vehicles observed from the overhead camera perspective. The perception pipeline uses YOLO as the primary source of vehicle localization. The YOLO model was trained on images collected from the experimental setup, allowing it to detect the vehicle appearance under the camera viewpoint used in the tests.

AprilTags were used as auxiliary identity markers when visible. Each tag encodes an identifier that can be used to support track identity assignment. However, the marker detections are not guaranteed to be available in every frame, since they can be affected by motion blur, viewing angle, image resolution, and partial occlusions. For this reason, the evaluation considers AprilTags as identity support rather than as the only tracking source.

Figure 5.3 shows an example of the vehicle and marker configuration used during the experiments.

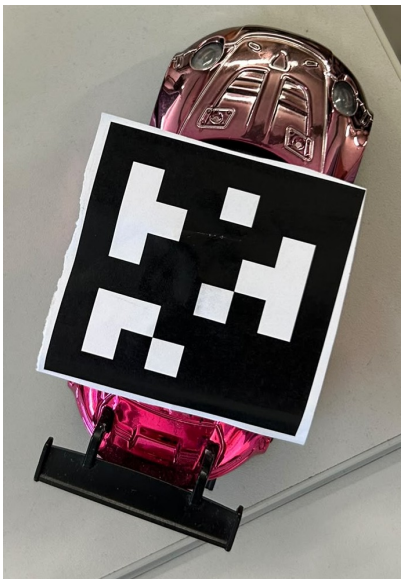

Figure 5.3: Example of the RC vehicle and AprilTag marker used in the experiments. The vehicle is detected using YOLO, while the AprilTag can provide auxiliary identity information when visible and correctly decoded.

5.1.4 Processing Hardware and Software Configuration

The perception pipeline was executed on an external processing computer. This computer receives the video stream, runs the detection modules, performs data association, updates the tracking state, estimates position, velocity, and acceleration, and stores the resulting logs.

The main software components used during evaluation were the same as those described in Chapter 4: Python, OpenCV, Ultralytics YOLO, an AprilTag detection library, NumPy, and SciPy. The trained YOLO model was loaded from the model artifact used in the implementation, and the same tracking and Kalman Filter parameters were used throughout the evaluated sequences unless otherwise stated.

Table 5.1 summarizes the main experimental setup elements.

5.2 Data Collection Procedure

Data collection was performed by running the perception pipeline on video sequences acquired from the experimental track. The system processed frames sequentially and stored relevant information in log files for later analysis. The collected data supports both qualitative inspection and quantitative evaluation based on the behaviour of detections, tracks, and state estimates over time.

Table 5.1: Summary of the experimental setup.

Component	Description
Track environment	Small-scale RC racing track observed from an external viewpoint.
Camera setup	Fixed camera positioned above the racing area.
Acquisition device	Raspberry Pi used to acquire and transmit the video stream.
Network connection	Wired Ethernet connection between the Raspberry Pi and the processing computer.
Processing computer	External computer running detection, tracking, state estimation, visualization, and logging.
Vehicle detection	Custom YOLO model trained for RC vehicle detection.
Identity support	AprilTag markers used as auxiliary identity cues when visible.
Logged data	Frame information, detections, active tracks, state estimates, and processing indicators.

5.2.1 Video Sequences

The experiments were organized into video sequences representing different operating conditions. Each sequence was processed by the same perception pipeline, allowing the behaviour of the system to be compared across scenarios.

The sequences were selected to evaluate different aspects of the system: basic vehicle detection, tracking continuity, behaviour during faster motion, AprilTag availability, and robustness during temporary missed detections. When available, sequences with more than one vehicle can also be used to evaluate the data association logic and identity preservation.

Table 5.2 presents the structure used to organize the experimental sequences. The exact sequence duration and number of processed frames should be filled according to the final recorded data.

5.2.2 Experimental Scenarios

The experimental scenarios were defined to test the main functions of the perception system. In the simplest case, a single vehicle moves through the track while the system detects it, creates a track, and estimates its motion state. This scenario is useful for verifying whether the full pipeline operates correctly under controlled conditions.

Additional scenarios introduce more difficult conditions. Faster vehicle motion can increase motion blur and make both YOLO detections and AprilTag decoding less stable. Temporary missed detections are useful for evaluating whether the tracking module can maintain a vehicle state

Table 5.2: Experimental sequences considered for evaluation.

Sequence	**Scenario**	**Purpose**
S1	Single vehicle, regular motion	Validate basic detection, tracking, and state estimation.
S2	Single vehicle, faster motion	Evaluate robustness to higher speed and motion blur.
S3	Intermittent AprilTag visibility	Analyse the contribution and limitations of tag-based identity support.
S4	Temporary missed detections	Evaluate track continuity during short detection gaps.
S5	Multi-vehicle motion	Evaluate data association and identity preservation with multiple active tracks.

through Kalman prediction. Sequences with intermittent AprilTag visibility allow the auxiliary identity mechanism to be analysed without assuming that markers are always available.

The experimental scenarios were therefore selected to evaluate the following behaviours:

- consistency of YOLO-based vehicle detections;
- availability and reliability of AprilTag detections;
- creation and confirmation of vehicle tracks;
- continuity of tracks during short detection gaps;
- stability of estimated position, velocity, and acceleration;
- processing time and real-time behaviour of the full pipeline.

5.2.3 Logged Data

During execution, the system stores structured logs containing information about detections, tracks, state estimates, and timing. These logs are used in the results chapter to analyse system behaviour over time.

The logged data can include:

- frame index;
- timestamp;
- processed frame rate or frame processing time;
- YOLO detection confidence;
- bounding box coordinates;

- detection source;
- track identifier;
- AprilTag identifier, when available;
- track status;
- number of missed frames;
- estimated position;
- estimated velocity;
- estimated acceleration.

The use of structured logs makes it possible to evaluate the system after the experiment, instead of relying only on the visual output shown during execution. This is particularly important for analysing state estimation, tracking continuity, and performance over time.

5.3 Evaluation Methodology

The evaluation methodology is divided into four parts: detection evaluation, tracking evaluation, state estimation evaluation, and real-time performance evaluation. Each part focuses on a different component of the implemented perception pipeline.

The evaluation combines quantitative indicators extracted from logs with qualitative visual inspection. This mixed approach was adopted because a complete manually annotated ground truth was not available for all recorded sequences. As a result, the evaluation focuses on whether the system behaves consistently under the tested scenarios and on identifying the main limitations observed during execution.

5.3.1 Detection Evaluation

Detection evaluation focuses on the behaviour of the YOLO and AprilTag detection modules. Since these two modules have different roles in the system, they are analysed separately.

YOLO is evaluated as the primary source of vehicle localization. The evaluation considers whether the vehicle is detected consistently across frames, whether bounding boxes remain stable, and whether detection confidence is sufficient for the tracking module. Missed detections and visible false detections are also considered, especially in situations involving motion blur, partial visibility, or fast vehicle motion.

AprilTag detection is evaluated as an auxiliary identity source. The evaluation focuses on whether the marker is decoded when visible, whether the decoded identifier is consistent, and how often tag detections are available during the tested sequences. Since AprilTags are not required

to be visible in every frame, their evaluation is focused on identity support rather than continuous localization.

The detection evaluation therefore considers:

- number of YOLO detections over time;
- confidence values of YOLO detections;
- visible missed detections;
- visible false detections;
- number of decoded AprilTag detections;
- consistency of decoded AprilTag identifiers;
- effect of motion blur, viewing angle, and marker visibility.

5.3.2 Tracking Evaluation

Tracking evaluation focuses on whether the system maintains consistent vehicle tracks across frames. Since detections are independent at each frame, the tracking module must associate current detections with existing tracks and preserve identity over time.

The main aspects evaluated are track continuity, track creation, track removal, identity preservation, and behaviour during missed detections. A successful tracking sequence should maintain the same track identifier for the same physical vehicle, avoid creating duplicate tracks, and recover from short detection gaps without immediately deleting the track.

The tracking evaluation considers:

- number of created tracks;
- number of confirmed tracks;
- number of removed tracks;
- duration of active tracks;
- number and duration of missed-frame intervals;
- visible identity switches;
- visible duplicated tracks;
- recovery after temporary detection losses.

For sequences with multiple vehicles, the evaluation also considers whether the data association logic preserves identities when vehicles move close to each other. If only single-vehicle sequences are used, the analysis focuses mainly on track stability, duplicated tracks, and continuity through missed detections.

5.3.3 State Estimation Evaluation

State estimation evaluation focuses on the behaviour of the Kalman Filter associated with each active track. The purpose is to assess whether the filter produces temporally consistent estimates of position, velocity, and acceleration from noisy visual measurements.

The position estimates are evaluated by comparing the raw position measurements with the filtered trajectory over time. The expected behaviour is that the filtered estimate follows the overall motion of the vehicle while reducing frame-to-frame fluctuations. During short missed-detection periods, the filter is expected to propagate the vehicle state using prediction.

Velocity and acceleration estimates are evaluated by analysing their temporal behaviour. Since these quantities are not directly measured by the camera, they are inferred from the evolution of position over time. Velocity is expected to be smoother and more interpretable than acceleration, while acceleration may remain more sensitive to noise and detection instability.

The state estimation evaluation considers:

- raw position measurements compared with filtered position estimates;
- estimated trajectory in the image plane or metric plane;
- estimated velocity over time;
- estimated acceleration over time;
- behaviour of the filter during temporary missed detections;
- sensitivity of the estimates to frame timing and detection noise.

5.3.4 Real-Time Performance Evaluation

Real-time performance evaluation focuses on whether the implemented pipeline can process frames at a rate suitable for live monitoring. The total processing time depends on frame acquisition, YOLO inference, AprilTag detection, data association, Kalman update, visualization, and logging.

The evaluation uses timing information collected during execution to estimate the average processing time per frame and the resulting frame rate. When possible, the time spent in individual pipeline stages is also analysed to identify the most computationally demanding components.

Table 5.3 presents the processing-time indicators considered in the evaluation.

5.4 Evaluation Metrics and Indicators

This section defines the main indicators used to analyse the results. Since a complete annotated benchmark dataset was not available, the evaluation does not rely on benchmark multi-object tracking metrics such as MOTA, MOTP, or IDF1. Instead, it uses indicators that can be obtained from the system logs and from visual inspection of the output.

Table 5.3: Processing-time indicators considered in the real-time performance evaluation.

Indicator	**Description**
Average FPS	Average number of processed frames per second.
Frame processing time	Time required to process one frame through the full pipeline.
Detection time	Time spent on YOLO and AprilTag detection.
Tracking time	Time spent on association, track management, and Kalman update.
Visualization and logging time	Time spent displaying annotated frames and storing outputs.

5.4.1 Log-Based Indicators

The log-based indicators are extracted directly from the data generated during the experiments. These indicators provide quantitative information about detection activity, track behaviour, state estimation, and processing time.

The main log-based indicators are:

- total number of processed frames;
- average frame rate;
- average processing time per frame;
- number of YOLO detections;
- number of AprilTag detections;
- number of active tracks per frame;
- number of created, confirmed, and removed tracks;
- duration of each track;
- number of missed frames per track;
- estimated position, velocity, and acceleration over time.

These indicators are used to support the analysis in the results chapter. For example, the number of missed frames helps evaluate tracking robustness, while velocity and acceleration profiles help assess the behaviour of the state estimation module.

5.4.2 Qualitative Assessment

Qualitative assessment is used to interpret behaviours that are not fully captured by the log-based indicators. This includes visual inspection of the annotated video output, where bounding boxes, track identifiers, AprilTag identifiers, and estimated states can be observed directly.

The qualitative assessment focuses on:

- whether detections visually correspond to the vehicle;
- whether the same vehicle keeps the same track identifier;
- whether duplicate tracks appear;
- whether tracks are maintained during short detection gaps;
- whether AprilTag detections support identity assignment when visible;
- whether the visual output is stable enough for real-time monitoring.

This type of assessment is especially useful for identifying identity switches, duplicated tracks, or incorrect associations. These issues may not be fully described by simple counts if a complete ground truth annotation is not available.

Chapter 6

Results and Discussion

This chapter presents and discusses the results obtained with the implemented off-board perception pipeline. The evaluation includes three experimental runs. The first run validates the behaviour of the system using YOLO-based detections, tracking, and Kalman-based state estimation without AprilTag support. The second run validates the hybrid YOLO–AprilTag behaviour using a larger AprilTag marker, which was detected and used as auxiliary identity information. The third run evaluates the system in a two-vehicle scenario, using vehicles equipped with AprilTags of different sizes.

The three runs are analysed together because they show complementary aspects of the system. The vision-based run demonstrates that the pipeline can maintain a stable vehicle track using YOLO detections and motion prediction alone. The hybrid run shows that, when the AprilTag is sufficiently visible in the image, the same tracking architecture can use the decoded marker as an additional identity cue while preserving a stable internal track identifier. Finally, the two-vehicle run shows that the system can maintain two simultaneous vehicle tracks, while also highlighting the practical effect of marker size on AprilTag availability.

6.1 Overview of the Evaluated Sequences

Two main experimental sequences were selected for the results analysis. Both sequences correspond to single-vehicle runs on the experimental RC racing track, observed from the fixed off-board camera. The same perception pipeline was used in both cases, with YOLO providing the main visual detections and the Kalman Filter estimating the vehicle state over time.

6.1.1 Vision-Based Run

The first evaluated sequence corresponds to a successful vision-based run. In this sequence, the system maintained a stable internal track identifier throughout the full run. The vehicle was detected and tracked using YOLO-based measurements, data association, track management, and Kalman prediction. The AprilTag marker used in this run had a physical size of approximately 5 cm and was not detected during the sequence.

This run is useful because it shows that the proposed system can operate even when the fiducial marker is unavailable. In this case, the tracking pipeline relied on visual detections and motion prediction to preserve the vehicle trajectory. The absence of AprilTag detections should therefore be interpreted as a limitation of marker visibility under the tested image resolution and camera geometry, rather than as a failure of the full tracking pipeline.

6.1.2 Hybrid Run

The second evaluated sequence corresponds to a successful hybrid run. In this sequence, a larger AprilTag marker of approximately 12 cm was used. The larger marker increased the apparent size of the tag in the image and allowed the AprilTag detector to decode the marker during most of the run. The internal track identifier remained stable during the sequence, while the decoded tag provided auxiliary identity information when visible.

This run is used to evaluate the intended hybrid behaviour of the system. YOLO remained the main source of vehicle localization, since the bounding box center provides the position measurement used by the tracking module. The AprilTag contributed explicit identity information when detected, supporting the association between the physical vehicle and the maintained track.

6.2 Detection Results

The detection results are analysed by considering the behaviour of YOLO and AprilTag detections separately. This distinction is important because the two detection sources have different roles in the system. YOLO provides the main vehicle localization measurement, while the AprilTag is used as an auxiliary identity cue.

6.2.1 YOLO Detection Behaviour

YOLO detections were available in both evaluated sequences and provided the main position measurements used by the tracking module. Figures 6.1 and 6.2 show the YOLO confidence values over time for the vision-based and hybrid runs, respectively.

In both runs, YOLO confidence values provide a useful indication of detection stability. Variations in confidence are expected because the apparent vehicle shape changes with orientation, speed, image blur, and position in the camera field of view. In the hybrid run, the confidence values remained high during the sequence, which confirms that YOLO continued to provide stable localization measurements even when AprilTag identity information was also available.

6.2.2 AprilTag Detection with the 5 cm Marker

The AprilTag marker used in the vision-based run had a physical size of approximately 5 cm. Figure 6.3 shows the AprilTag detection availability for this run. The detection signal remained inactive throughout the sequence, meaning that the marker was not decoded.

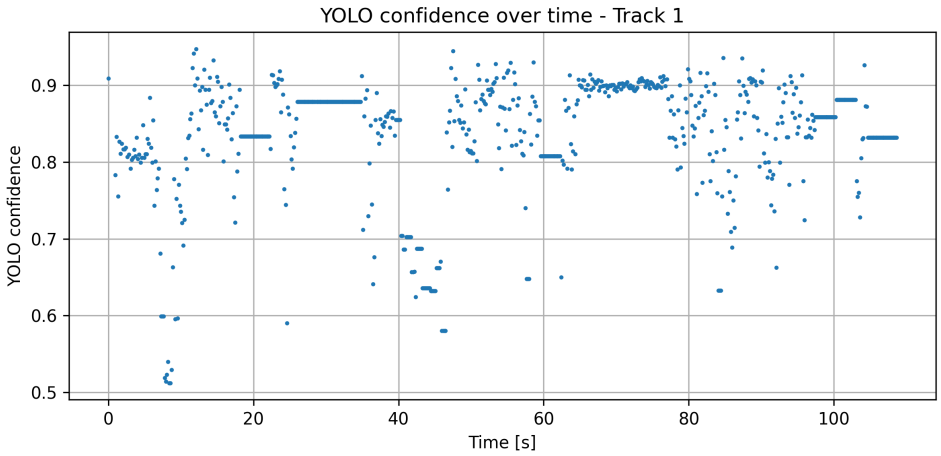

Figure 6.1: YOLO confidence values over time for the successful vision-based run. The detector provided the main visual measurements used to maintain the vehicle track.

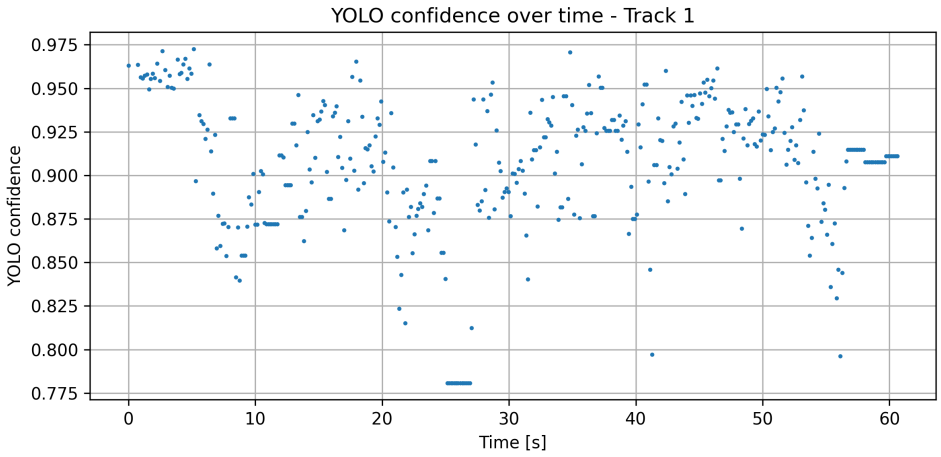

Figure 6.2: YOLO confidence values over time for the successful hybrid run. YOLO remained the primary localization source, while the AprilTag provided auxiliary identity information when detected.

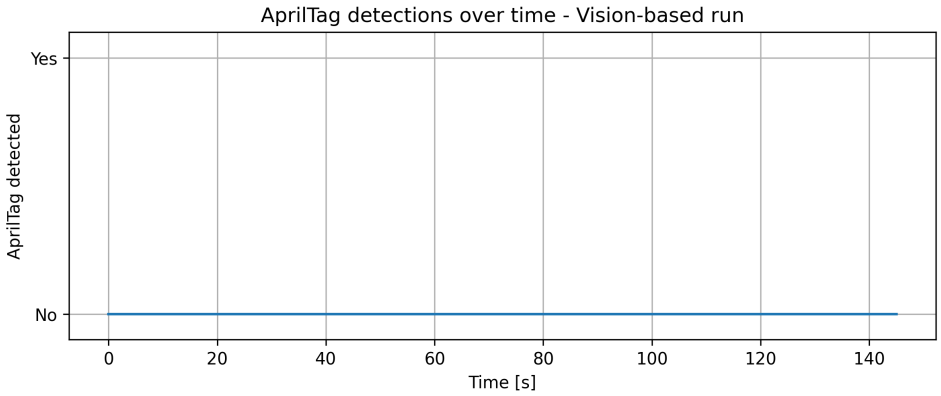

Figure 6.3: Decoded AprilTag identifier over time for the vision-based run. Since the 5 cm marker was not detected, no valid tag identifier was available.

The absence of AprilTag detections is consistent with the experimental setup. The stream resolution was 1280×720 pixels, and the camera was positioned high enough to observe the full track. Under these conditions, the 5 cm marker occupied a small number of pixels in the image, which made reliable decoding difficult.

The result does not reduce the relevance of the hybrid architecture. It shows that the fiducial marker component depends on the apparent marker size in the image. In this run, YOLO detections and the tracking module were sufficient to maintain a stable vehicle track, while the AprilTag component was limited by marker visibility.

6.2.3 Hybrid Detection with the 12 cm Marker

The hybrid run was performed with a larger AprilTag marker of approximately 12 cm. This marker occupied a larger region of the image and was detected during most of the sequence. Figure 6.4 shows the AprilTag detection availability over time, while Fig. 6.5 shows the decoded tag identifier.

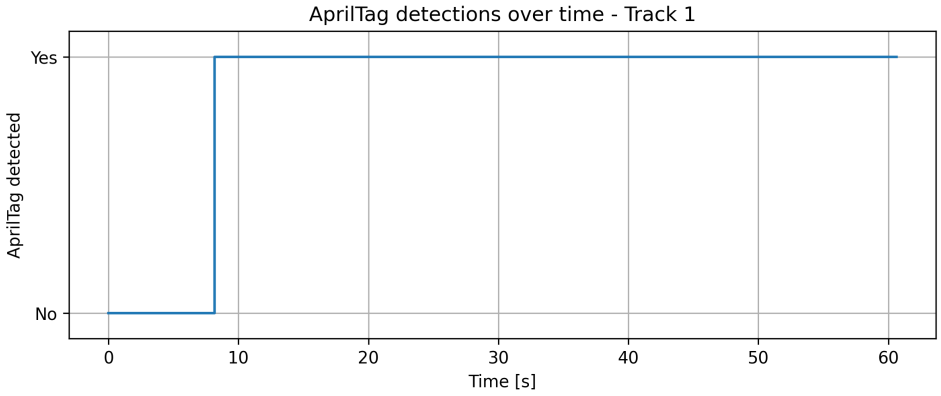

Figure 6.4: AprilTag detection availability over time for the successful hybrid run with the 12 cm marker. The marker was not detected during the initial seconds, but was decoded consistently after becoming visible to the detector.

The marker was not decoded during the first seconds of the sequence. After this initial interval, the detection signal remained active until the end of the run. This indicates that the larger marker occupied enough pixels to be reliably decoded under the experimental camera configuration.

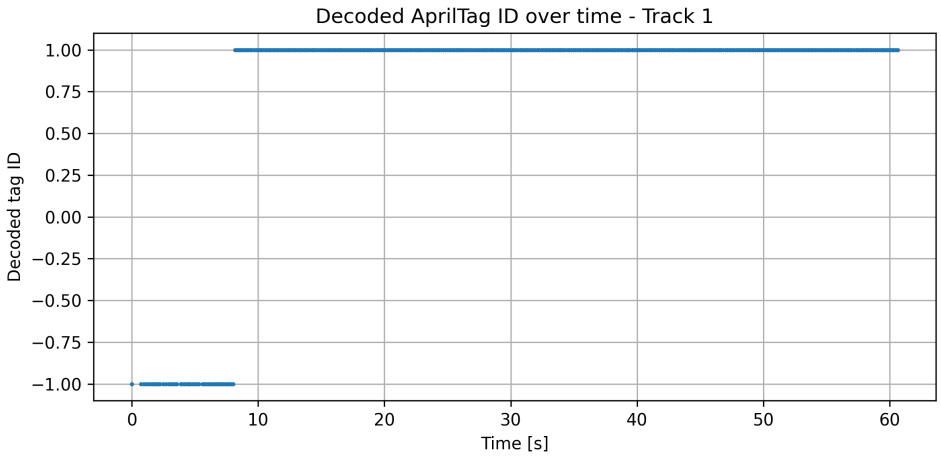

Figure 6.5: Decoded AprilTag identifier over time for the hybrid run. After the initial interval without tag detection, the decoded identifier remained stable at ID 1.

The stable decoded tag identifier confirms that the AprilTag provided consistent identity information once it became visible to the detector. In this run, YOLO continued to provide the main vehicle localization measurement, while the AprilTag supplied an explicit identity cue associated with the tracked vehicle. This validates the intended hybrid behaviour of the system.

Figure 6.6 summarizes the track update source distribution for the hybrid run. This result should be interpreted together with the AprilTag detection plot. The fact that vision-based updates are more frequent does not mean that the tag was rarely detected. Instead, it means that the tracking state was usually updated using the YOLO bounding box, while the decoded AprilTag provided identity information when available.

Together with the stable internal track identifier shown later in Fig. 6.8, these results show that the hybrid configuration was successful. The AprilTag was integrated as auxiliary identity information without disrupting the continuity of the internal track.

6.2.4 Effect of Marker Size and Image Resolution

The difference between the 5 cm and 12 cm marker runs is mainly explained by the apparent size of the AprilTag in the image. The 5 cm marker was not detected in the 1280×720 stream, while the 12 cm marker became detectable under the same general camera viewpoint. This shows that the limiting factor was not the hybrid design itself, but the number of pixels occupied by the marker.

The 12 cm marker should therefore be interpreted as a practical compensation for the experimental camera configuration. It was used to increase the apparent size of the fiducial marker and make the hybrid behaviour observable in the laboratory setup. With a higher-resolution camera stream, a smaller physical marker could occupy a similar number of pixels and become detectable.

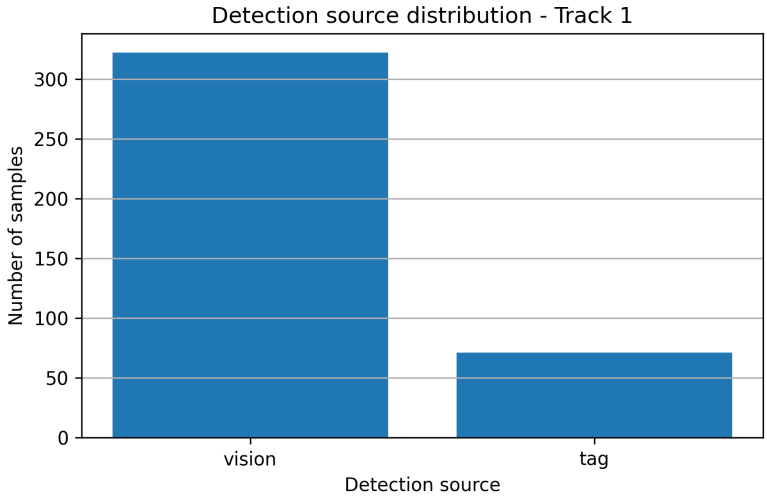

Figure 6.6: Track update source distribution for the successful hybrid run. Although the AprilTag was decoded during most of the sequence, most state updates remained vision-based because YOLO provided the primary localization measurement. The AprilTag was used mainly as auxiliary identity support.

To put this into context, Table 6.1 shows an approximate marker-size equivalence using a 12 cm AprilTag in the current 1280×720 stream as reference. The comparison assumes the same viewpoint, lens, distance, and field of view. Under these assumptions, increasing the image resolution allows a smaller physical marker to occupy a similar number of pixels in the image.

Table 6.1: Approximate marker-size equivalence using a 12 cm AprilTag at 1280×720 as reference, assuming the same viewpoint, lens, distance, and field of view.

Resolution	**Relative pixel scale**	**Marker size equivalent to 12 cm at 1280 px**
1280×720	1.00	12.0 cm
1920×1080	1.50	8.0 cm
2560×1440	2.00	6.0 cm
3072×1728	2.40	5.0 cm
3840×2160	3.00	4.0 cm
4056×3040	3.17	3.8 cm

This comparison shows that the 12 cm marker used in the hybrid run should be interpreted as a practical compensation for the limited resolution of the experimental stream. Under the same optical setup, a 5 cm marker at approximately 3072 pixels of horizontal resolution would occupy a similar number of pixels to a 12 cm marker in the current 1280-pixel stream. Therefore, the use of a larger tag in the laboratory does not mean that such a large marker would be required in a higher-resolution implementation.

6.3 Tracking Results

The tracking results are evaluated using the internal track identifier and the missed-frame counter. These indicators show whether the system maintained a stable vehicle identity and how it behaved during short detection gaps.

6.3.1 Track Continuity in the Vision-Based Run

Figure 6.7 shows the internal track identifier over time for the vision-based run. The track identifier remained constant throughout the sequence. This indicates that the system did not fragment the same physical vehicle into multiple internal tracks, even though no AprilTag identity cue was available.

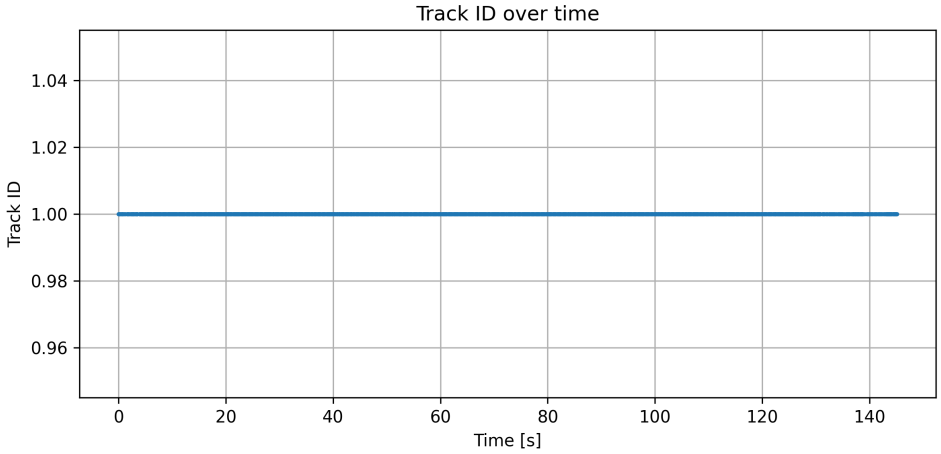

Figure 6.7: Internal track identifier over time for the successful vision-based run. The same track ID was maintained throughout the sequence, indicating stable tracking without AprilTag support.

This result is important because it shows that YOLO detections, data association, track management, and Kalman prediction were sufficient to maintain identity continuity in a single-vehicle scenario. The AprilTag was unavailable, but the track did not reset or split into multiple identifiers.

6.3.2 Track Continuity in the Hybrid Run

Figure 6.8 shows the internal track identifier over time for the hybrid run. As in the vision-based run, the track identifier remained stable during the sequence. This indicates that the addition of AprilTag identity information did not disrupt track continuity.

This result is particularly relevant because the AprilTag was decoded during most of the sequence, but the internal track identifier remained constant. This indicates that the tag was integrated as auxiliary identity information without creating a competing track or causing identity fragmentation. The AprilTag provides explicit identity information when visible, while the internal tracking identity remains controlled by the tracking module.

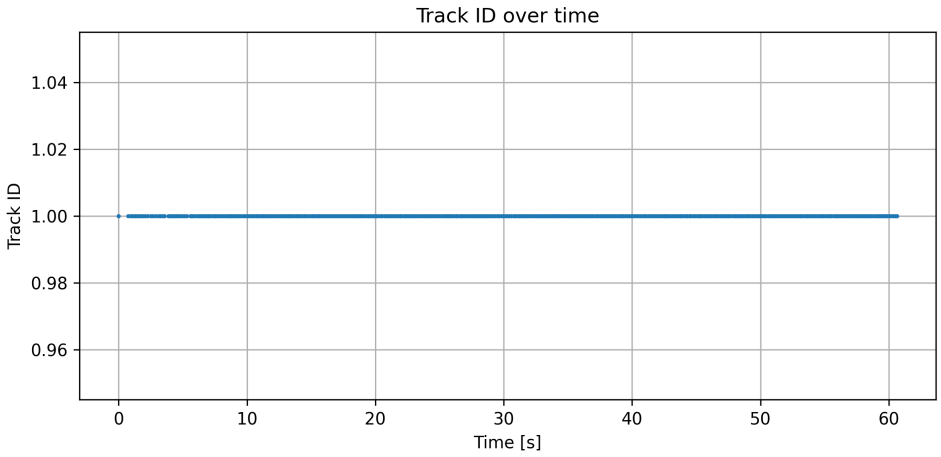

Figure 6.8: Internal track identifier over time for the successful hybrid run. The stable track ID shows that the system preserved the same internal identity while using AprilTag detections as auxiliary identity cues.

6.3.3 Missed Detections and Track Recovery

Figures 6.9 and 6.10 show the missed-frame counter over time for the two runs. This counter increases when an active track is not associated with a detection in the current frame.

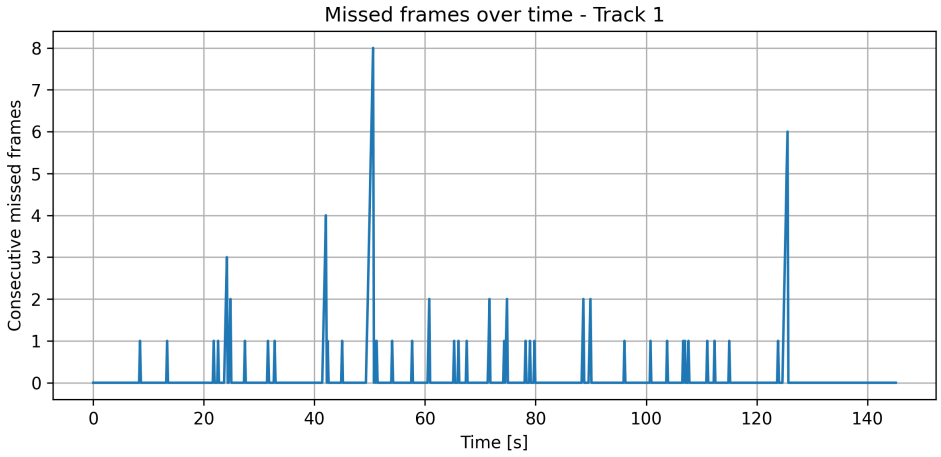

Figure 6.9: Consecutive missed frames over time for the vision-based run. The main peak corresponds to a short interval where sunlight or reflection degraded the image and the vehicle was temporarily not associated with a detection. The tracker recovered without changing the internal track identifier.

The two runs show different missed-frame behaviours. In the vision-based run, a short peak was observed and the tracker recovered without changing the internal track identifier. This peak was associated with temporary image degradation caused by sunlight or reflection on the scene. During this interval, the tracker relied on the Kalman prediction step to propagate the vehicle state.

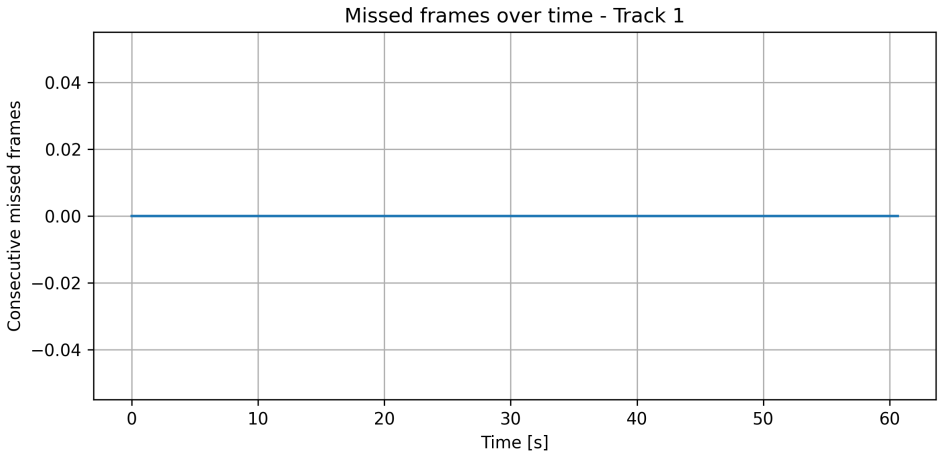

Figure 6.10: Consecutive missed frames over time for the hybrid run. The counter remained at zero throughout the sequence, indicating that no temporary detection loss occurred for the maintained track.

In the hybrid run, the missed-frame counter remained at zero throughout the sequence, indicating that the vehicle was consistently associated with detections.

6.4 Trajectory and State Estimation Results

The state estimation results are analysed using the estimated trajectories, position measurements, Kalman estimates, and motion quantities. The same types of plots are used for both runs, allowing the vision-based and hybrid behaviours to be compared.

6.4.1 Trajectory Estimation

Figures 6.11 and 6.12 show the estimated vehicle trajectory in metric coordinates for the two runs. The metric representation is useful because it allows the vehicle motion to be interpreted in physical units rather than only in pixels.

The trajectories show that the system produced consistent position estimates for the vehicle motion around the track. Small deviations between laps are expected because the vehicle does not follow exactly the same path every time and because the measurements are affected by visual detection noise. The trajectory plots are kept separate from the measurement-estimate comparison to make the spatial path easier to interpret. The agreement between measured and estimated positions is analysed in the following subsection.

6.4.2 Kalman Filter Behaviour

The behaviour of the Kalman Filter can be analysed by comparing raw position measurements with the filtered estimates. Figures 6.13 and 6.14 show this comparison for the vision-based run, while Figs. 6.15 and 6.16 show the same analysis for the hybrid run.

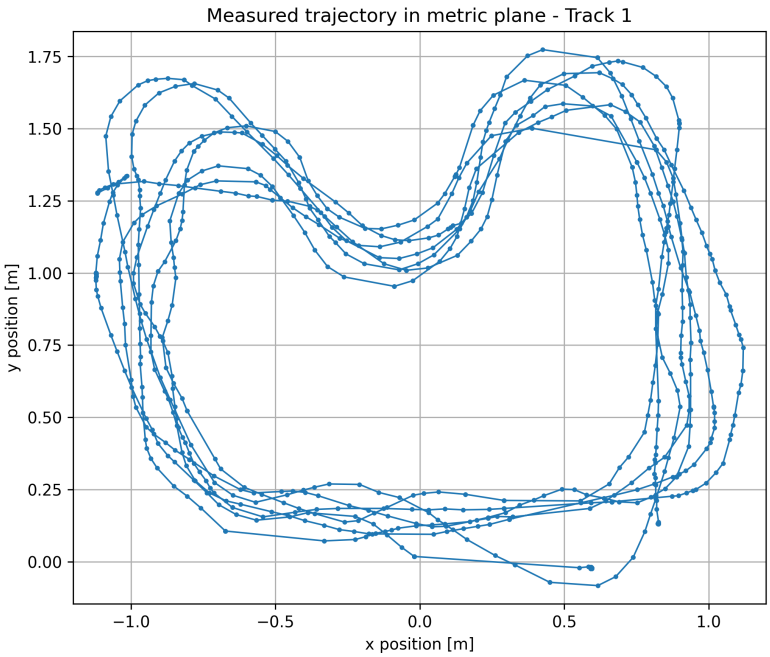

Figure 6.11: Vehicle trajectory in the metric plane for the successful vision-based run. The trajectory shows the path followed by the vehicle around the experimental track.

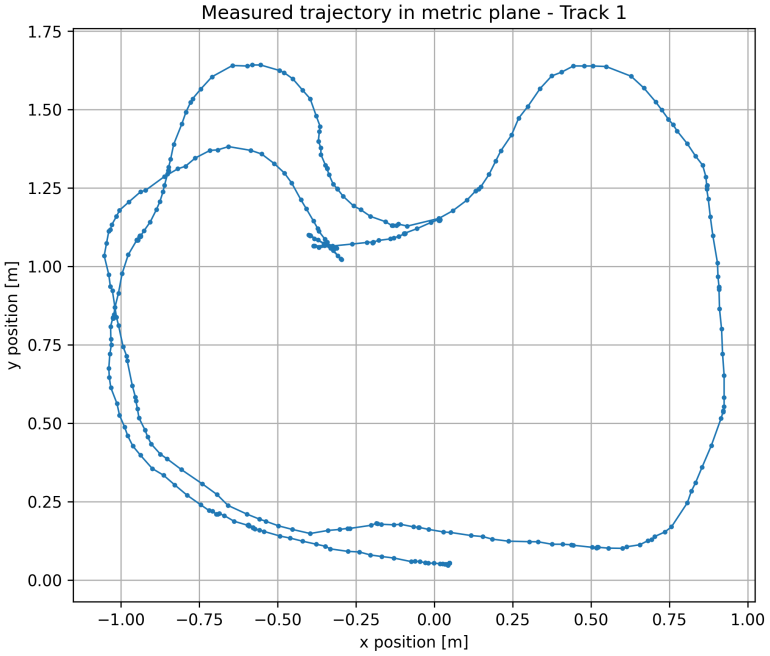

Figure 6.12: Vehicle trajectory in the metric plane for the successful hybrid run. The trajectory is obtained while the system combines YOLO-based localization with AprilTag identity support.

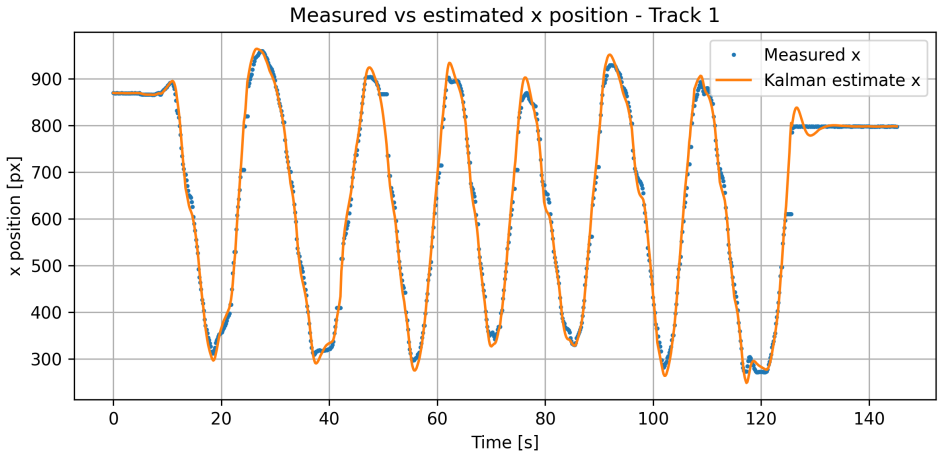

Figure 6.13: Measured and Kalman-estimated horizontal position over time for the vision-based run. The estimate follows the visual measurements while maintaining a continuous state representation.

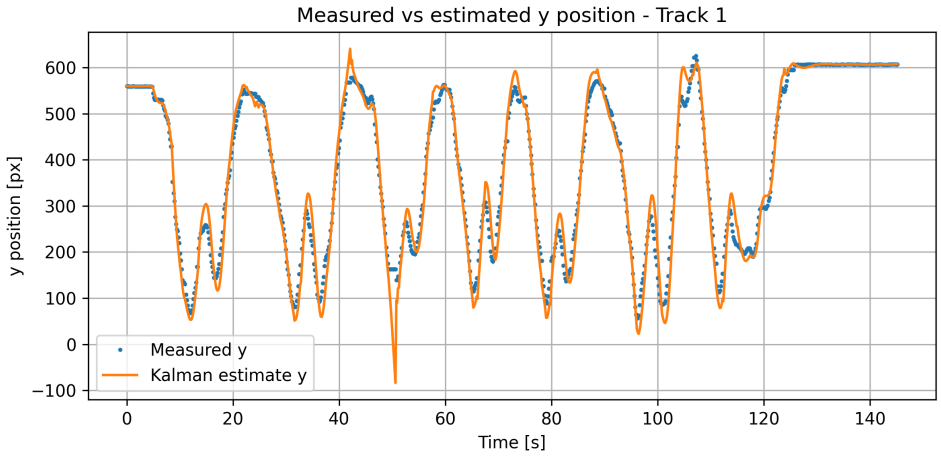

Figure 6.14: Measured and Kalman-estimated vertical position over time for the vision-based run. The filter follows the main motion pattern and supports continuity during short gaps.

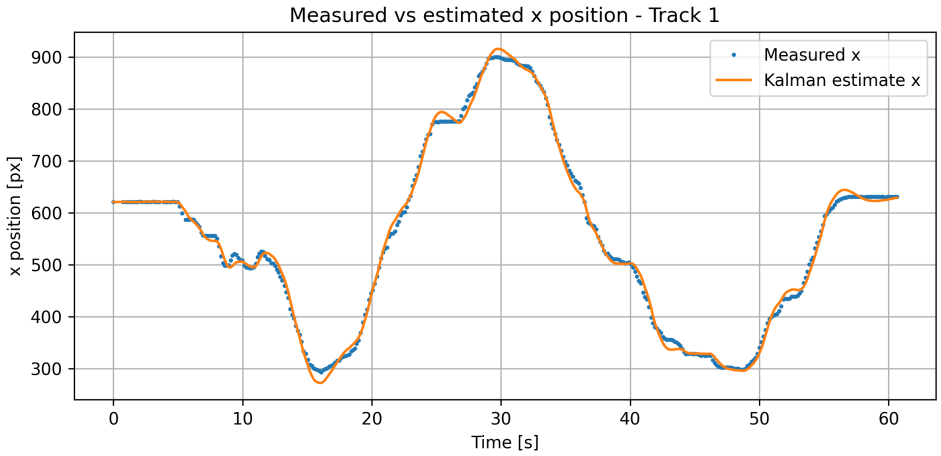

Figure 6.15: Measured and Kalman-estimated horizontal position over time for the hybrid run. The filtered estimate remains aligned with the visual measurements while the AprilTag provides auxiliary identity information.

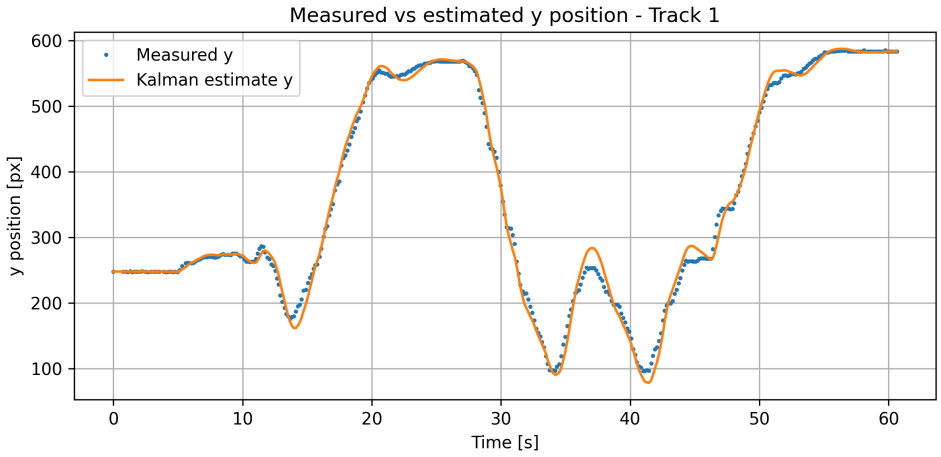

Figure 6.16: Measured and Kalman-estimated vertical position over time for the hybrid run. The Kalman Filter provides temporal consistency between measurements.

In both runs, the filtered estimates follow the raw measurements while maintaining a coherent state over time. In the vision-based run, the prediction step was relevant during the short missed-detection interval observed in Fig. 6.9. In the hybrid run, the missed-frame counter remained at zero, and the Kalman Filter mainly provided temporal smoothing and a structured state representation between consecutive measurements. This behaviour is consistent with the role of the Kalman Filter in the implemented tracker.

6.4.3 Velocity and Acceleration Estimates

The Kalman state also provides velocity and acceleration estimates. Figures 6.17 and 6.18 show the estimated speed magnitude for the two runs. The speed is more interpretable than the individual velocity components because it represents the magnitude of motion independently of direction.

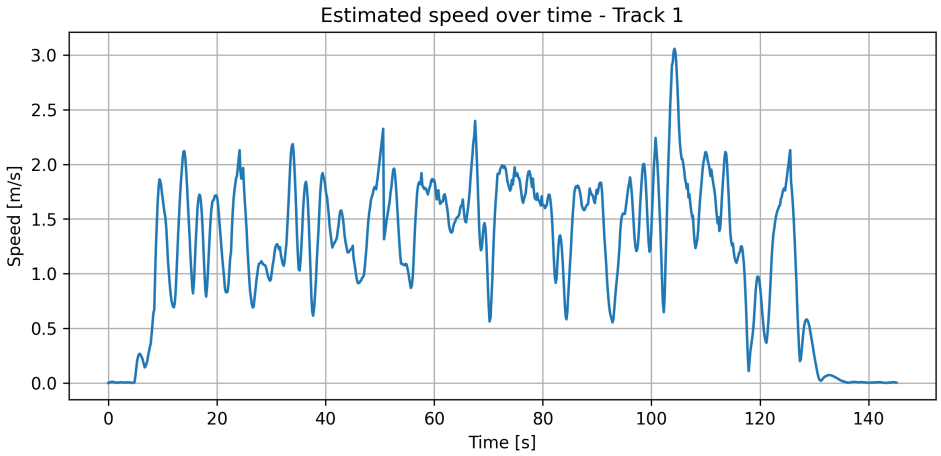

Figure 6.17: Estimated vehicle speed over time for the vision-based run. The speed profile reflects the vehicle motion around the track and periods of lower motion.

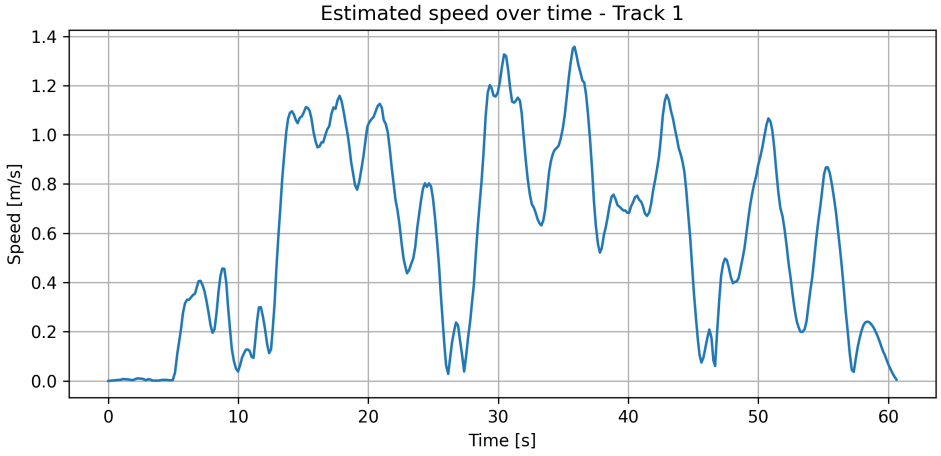

Figure 6.18: Estimated vehicle speed over time for the hybrid run. The speed estimate is obtained from the Kalman state while the track identity remains stable.

Acceleration estimates are shown in Figs. 6.19 and 6.20. These plots should be interpreted more carefully than position and velocity. Acceleration is inferred indirectly from image-based position measurements and is therefore more sensitive to small variations in detection position and frame timing.

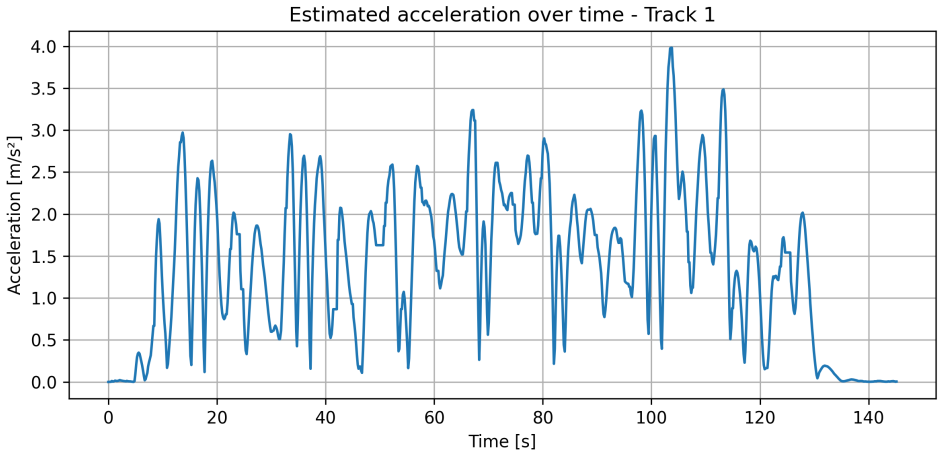

Figure 6.19: Estimated acceleration magnitude over time for the vision-based run. The acceleration estimate is more variable than position and velocity because it is inferred indirectly from visual measurements.

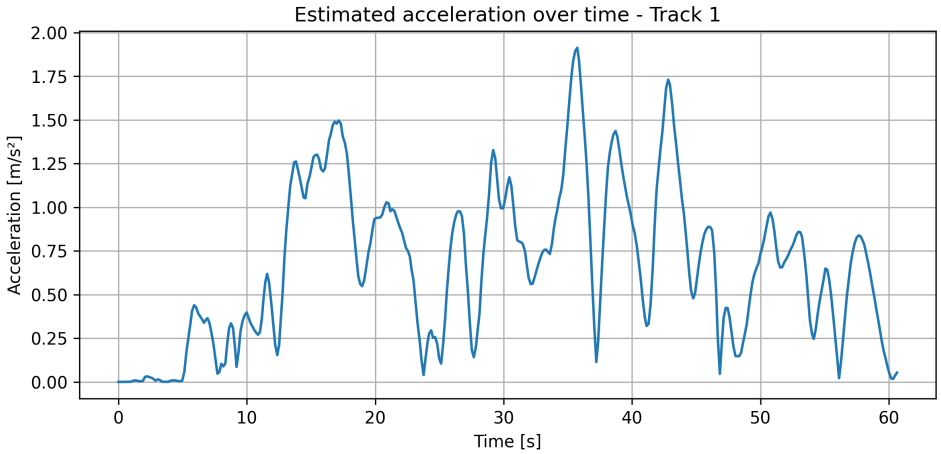

Figure 6.20: Estimated acceleration magnitude over time for the hybrid run. The acceleration profile captures changes in motion but remains sensitive to measurement noise and timing variations.

The acceleration results are useful as motion indicators, but they should not be interpreted with the same confidence as the position estimates. Small variations in detected position, camera stream timing, or association behaviour can produce larger changes in acceleration. This sensitivity is expected because acceleration is derived from changes in motion over time.

6.5 Two-Vehicle Run with 5 cm and 10 cm AprilTags

After evaluating the perception pipeline in single-vehicle sequences, a two-vehicle run was performed to analyse the behaviour of the system in a multi-vehicle scenario. This experiment is particularly relevant because the tracking problem becomes more difficult when more than one vehicle is visible in the image. In the single-vehicle runs, each detection normally corresponds to only one possible track. With two vehicles moving on the same circuit, the system must keep separate internal states, associate detections with the correct vehicle, and avoid creating duplicate or fragmented track identifiers.

In this run, the two vehicles were equipped with AprilTags of different sizes. The vehicle associated with Track 1 used a 10 cm AprilTag, while the vehicle associated with Track 2 used a 5 cm AprilTag. The main localization source remained the YOLO-based vehicle detector, while the AprilTags were used as auxiliary identity cues when visible and decoded. The results presented in this section focus on track continuity, detection stability, AprilTag availability, trajectory estimation, Kalman Filter behaviour, and motion-related estimates.

6.5.1 Track Continuity and Active Tracks

The first aspect analysed is whether the tracker was able to maintain two independent vehicle tracks. Figure 6.21 shows the evolution of the internal track identifiers during the two-vehicle run. Only Track 1 and Track 2 are present, which indicates that the tracker preserved two distinct internal vehicle identities. This is an important result because it shows that the association and track management logic was able to represent both vehicles without creating additional identifiers.

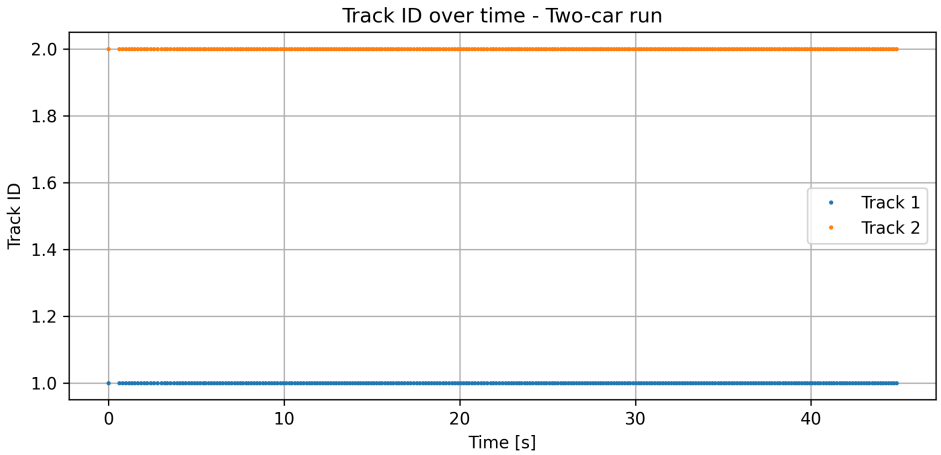

Figure 6.21: Track identifier evolution during the two-vehicle run with 5 cm and 10 cm AprilTags. The tracker maintained two internal vehicle identities, Track 1 and Track 2.

The number of active tracks over time is shown in Figure 6.22. The result remains constant at two active tracks, confirming that both vehicles were followed at the same time. Together with

the track identifier plot, this shows that the implemented pipeline was able to maintain a stable multi-vehicle representation under the tested conditions.

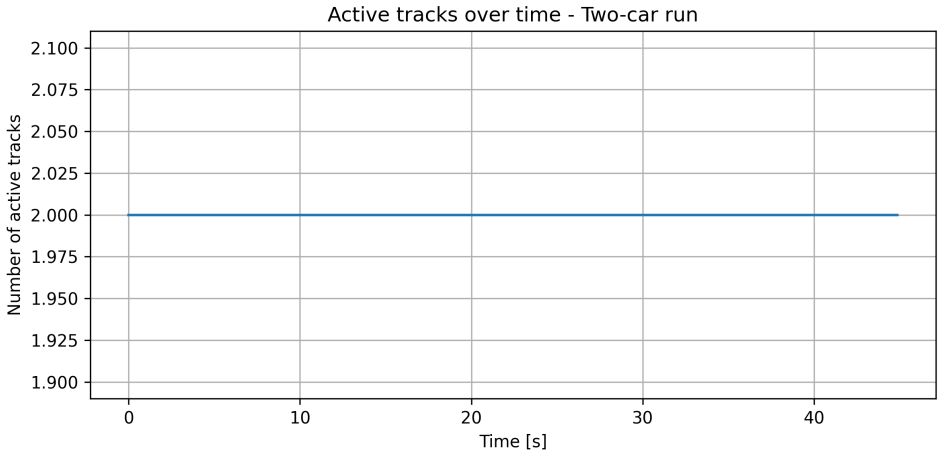

Figure 6.22: Number of active tracks over time during the two-vehicle run. The system maintained two simultaneous vehicle tracks.

6.5.2 Vehicle Detection and AprilTag Availability

The behaviour of the YOLO detector during the two-vehicle run is shown in Figure 6.23. The confidence values remain generally high for both tracks, with most detections above approximately 0.8. Some lower-confidence detections are visible, especially for Track 2, but these did not prevent the tracker from maintaining both vehicles. This indicates that the detector provided sufficiently stable measurements for the tracking module in this multi-vehicle run.

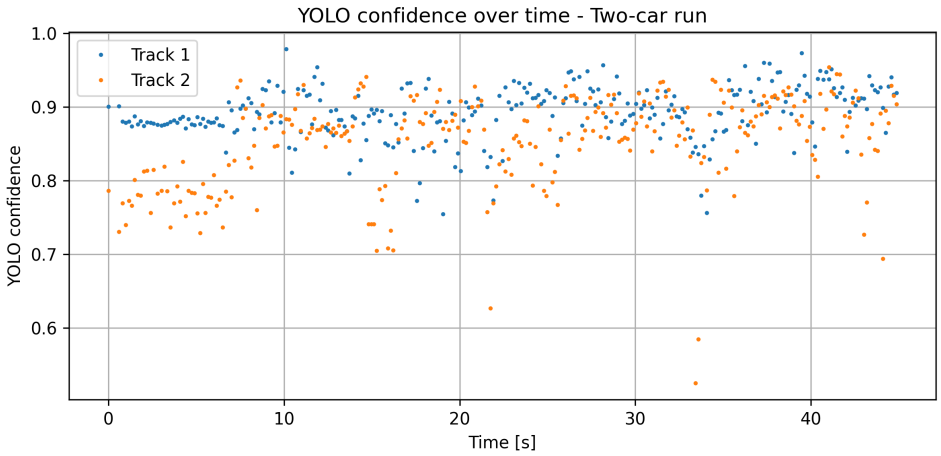

Figure 6.23: YOLO confidence values over time for both tracks in the two-vehicle run.

Figure 6.24 shows the AprilTag detection availability. A valid AprilTag detection was registered for Track 1, corresponding to the vehicle equipped with the 10 cm marker. The marker became

available during the run and remained detected afterwards. For Track 2, corresponding to the vehicle equipped with the 5 cm marker, no valid AprilTag detection was recorded.

This result is consistent with the previous observations regarding the influence of marker size. Under the same camera resolution and viewpoint, the 10 cm AprilTag provided enough image detail to be decoded, while the 5 cm marker was not detected reliably. In this run, the 10 cm AprilTag therefore acted as auxiliary identity information for Track 1, while Track 2 was maintained through YOLO-based localization and Kalman-based prediction. This reinforces the role of AprilTags in the implemented pipeline: they can support identity assignment when available, but they are not treated as the main localization source.

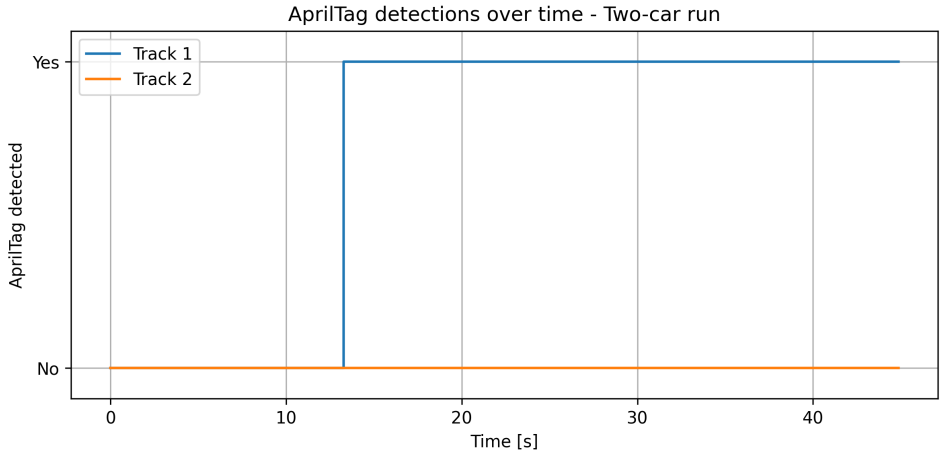

Figure 6.24: AprilTag detection availability during the two-vehicle run with 5 cm and 10 cm markers. The 10 cm marker associated with Track 1 was detected, while the 5 cm marker associated with Track 2 was not decoded.

The update source distribution is presented in Figure 6.25. The logged source is classified as vision-based for both tracks. This does not mean that the AprilTag information was irrelevant. Instead, it reflects the implementation logic: the YOLO bounding box provides the position measurement used to update the track state, while the decoded AprilTag is used as auxiliary identity support when it is available.

6.5.3 Trajectory Estimation

The measured trajectories of the two vehicles are shown in Figure 6.26. The result shows two separate paths around the experimental circuit, corresponding to the two internal tracks maintained by the system. The trajectories follow the same physical track but remain distinguishable, which is expected from two vehicles moving on the circuit at the same time.

This result is useful because it confirms that the logging and tracking structure preserved the motion history of each vehicle independently. Each position sample is associated with its corresponding internal track identifier, allowing the movement of both vehicles to be analysed separately.

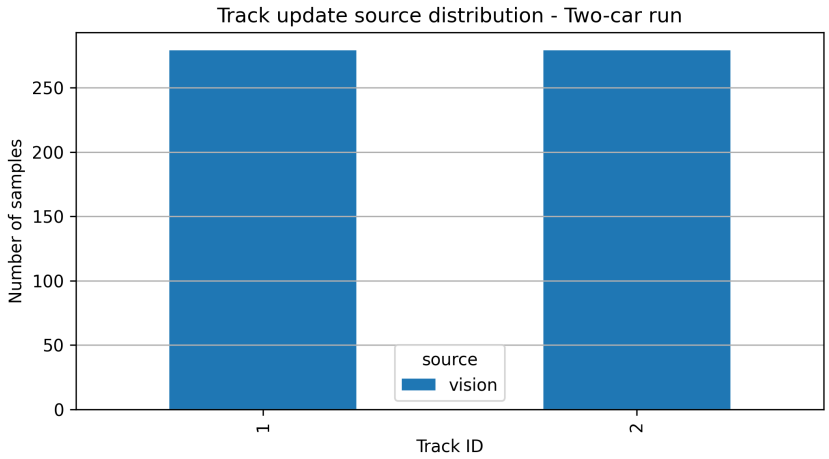

Figure 6.25: Track update source distribution for the two-vehicle run. The state updates were vision-based for both tracks, while the detected 10 cm AprilTag was used as auxiliary identity support for Track 1.

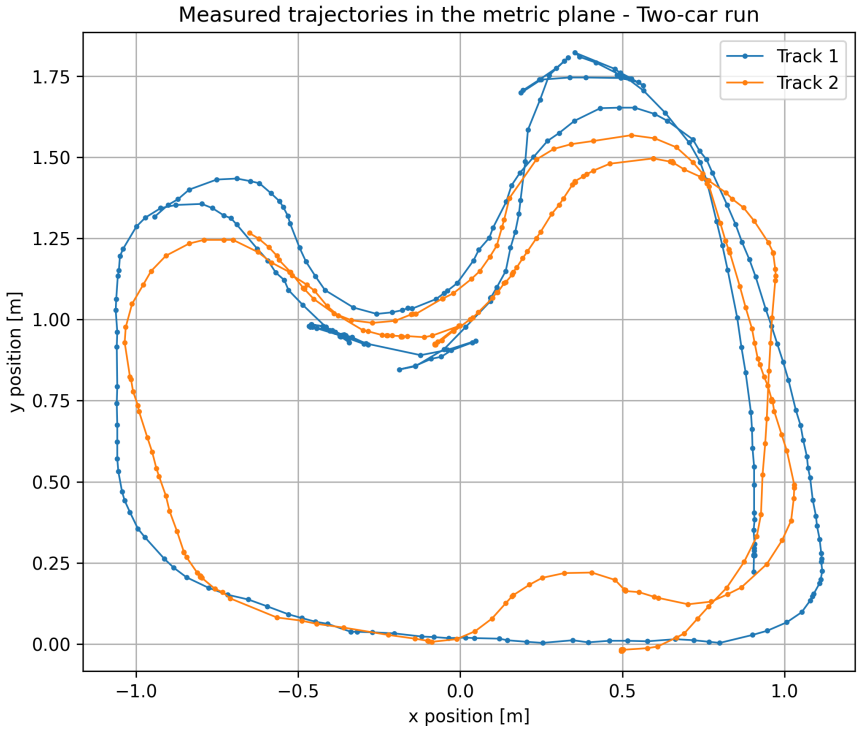

Figure 6.26: Measured trajectories of the two vehicles in the metric reference frame during the two-vehicle run.

6.5.4 Kalman Filter Behaviour

The measured and estimated image positions for Track 1 are shown in Figure 6.27. The Kalman estimate follows the main shape of the measured motion in both image coordinates. The estimate smooths part of the frame-to-frame variation, while still following the vehicle as it moves around the circuit. Some lag and overshoot can be observed during sharper direction changes, which is expected because the filter combines noisy image measurements with a motion model.

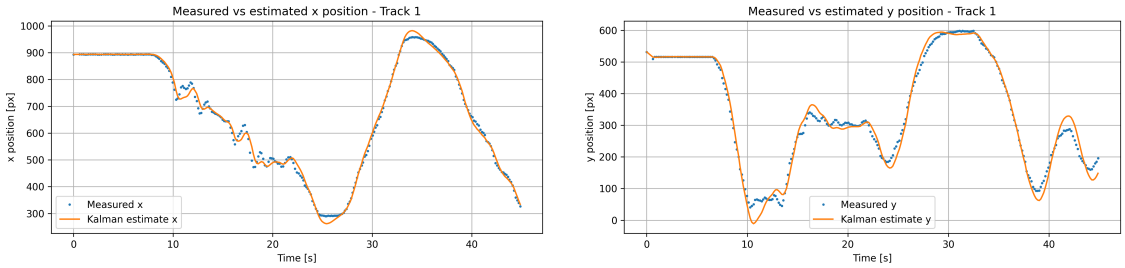

Figure 6.27: Measured and estimated image positions for Track 1 during the two-vehicle run: x coordinate on the left and y coordinate on the right.

The same comparison for Track 2 is presented in Figure 6.28. The estimated state also follows the measured position of the second vehicle. This is particularly relevant because Track 2 was maintained without a valid AprilTag detection. The result shows that the system was able to estimate the motion of the vehicle equipped with the 5 cm marker using only YOLO-based measurements and the Kalman Filter prediction-update cycle.

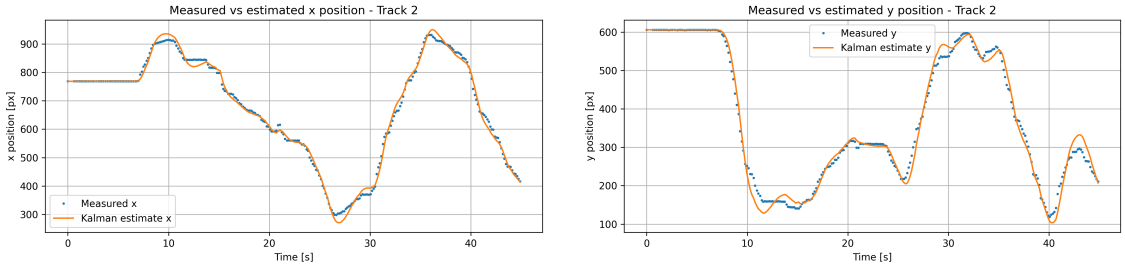

Figure 6.28: Measured and estimated image positions for Track 2 during the two-vehicle run: x coordinate on the left and y coordinate on the right.

Figure 6.29 presents the measurement-estimate position error for both tracks. The error remains within a limited range for most of the run, with larger peaks during periods of faster motion or sharper changes in direction. These peaks are consistent with the behaviour seen in the measured versus estimated position plots. They do not indicate loss of tracking, but rather the expected difference between a noisy visual measurement and the smoother state estimated by the filter.

Figure 6.29: Measurement-estimate position error for both tracks during the two-vehicle run.

6.5.5 Missed Detections and Motion Estimates

Figure 6.30 shows the number of consecutive missed frames for both tracks. Track 1 remained continuously associated with detections, while Track 2 presented only a short peak of two missed frames. This indicates that the tracker was able to maintain both vehicles with only a minor temporary loss of association.

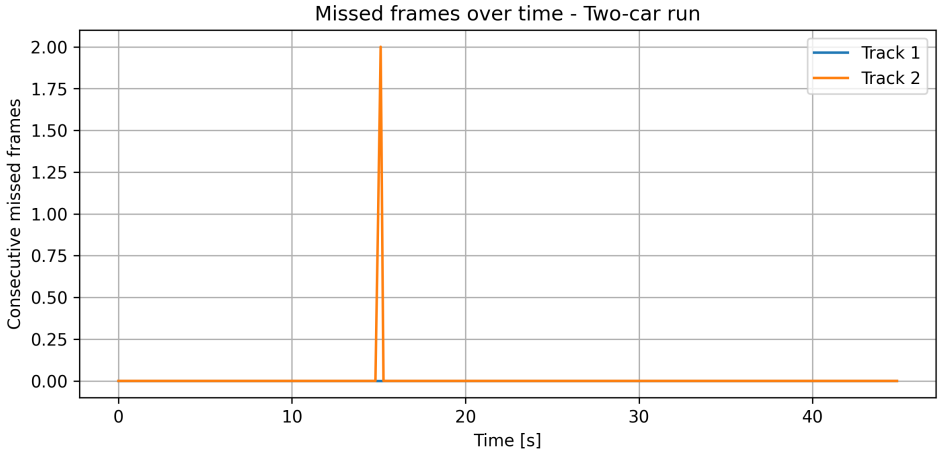

Figure 6.30: Consecutive missed frames over time for both tracks during the two-vehicle run.

The estimated speed and acceleration are shown in Figures 6.31 and 6.32. The speed profiles reflect the motion of the two vehicles around the circuit, including both slower and faster sections. The acceleration estimate is more variable than the speed estimate, which is consistent with the behaviour observed in the single-vehicle runs. Since acceleration is inferred indirectly from image-based position measurements, it is more sensitive to detection noise, frame-to-frame variation, and sudden changes in direction.

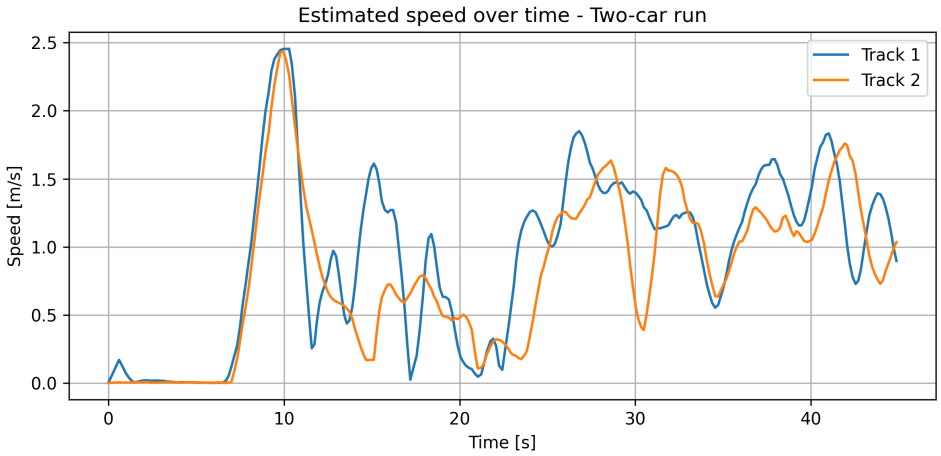

Figure 6.31: Estimated speed over time for both tracks during the two-vehicle run.

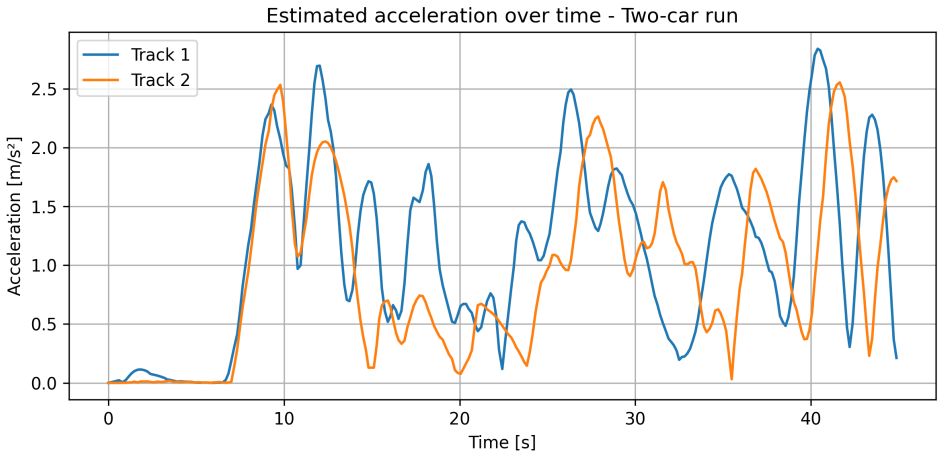

Figure 6.32: Estimated acceleration over time for both tracks during the two-vehicle run.

6.5.6 Discussion of the Two-Vehicle Run

The two-vehicle run shows that the implemented off-board perception pipeline can maintain two simultaneous vehicle tracks using a fixed external camera. The system preserved two internal identifiers, kept two active tracks throughout the run, and produced separate trajectory and state estimates for both vehicles.

The results also clarify the practical effect of AprilTag marker size. The 10 cm marker associated with Track 1 was decoded and provided auxiliary identity information during the run. In contrast, the 5 cm marker associated with Track 2 was not detected under the same acquisition conditions. However, the absence of a valid tag for Track 2 did not prevent the system from tracking the second vehicle, since the YOLO-based detections remained the main localization source.

Overall, this run extends the single-vehicle validation by showing that the same tracking and state estimation architecture can operate with two vehicles. It also reinforces one of the main findings of the marker-based experiments: AprilTags can support identity assignment when their apparent size is sufficient, but they should not be treated as the primary tracking source in the current acquisition setup.

6.6 Discussion

The results presented in this chapter show that the implemented pipeline can maintain stable vehicle tracking in both single-vehicle and two-vehicle configurations. In the vision-based single-vehicle run, YOLO detections and Kalman-based prediction were sufficient to preserve a continuous internal track without AprilTag support. In the hybrid single-vehicle run, the larger AprilTag became available after the initial seconds of the sequence and remained consistently decoded as ID 1. The internal track identifier remained constant, and the missed-frame counter stayed at zero. These results indicate that the hybrid component can provide identity support without disrupting the YOLO-based tracking pipeline.

The two-vehicle run extends this validation to a simple multi-vehicle scenario. In this run, the system maintained two simultaneous active tracks, Track 1 and Track 2, without creating additional internal identifiers. This is an important result because it shows that the association and track management logic can separate two vehicles moving on the same circuit. The measured trajectories and state estimation plots also show that the logging structure preserved the motion history of each vehicle independently.

The comparison between the runs clarifies the role of each component. YOLO remains the main localization source, since the vehicle bounding box provides the position measurement used by the tracker. The Kalman Filter supports temporal consistency and allows the system to propagate the vehicle state between detections. AprilTags are useful when visible because they provide an explicit identity cue, but their availability depends strongly on the apparent marker size in the image.

The source distribution should therefore be interpreted as the track update source, rather than as a direct measure of AprilTag availability. Most updates remained vision-based because YOLO provided the primary localization measurement. A frame can contain a valid AprilTag detection while the corresponding track update is still classified as vision-based, since the tag is mainly used to support identity assignment.

The marker-size results are consistent across the experiments. The 5 cm marker was not detected reliably in the current 1280×720 stream. In the single-vehicle hybrid run, the 12 cm marker was detected and allowed the hybrid behaviour to be evaluated. In the two-vehicle run, the 10 cm marker associated with Track 1 was detected, while the 5 cm marker associated with Track 2 was not decoded. These results suggest that the current acquisition setup requires a sufficiently large apparent marker size for reliable AprilTag decoding. This does not mean that smaller markers are unsuitable in general. With a higher-resolution image stream, improved focus, or a closer camera viewpoint, a smaller physical marker could occupy enough pixels to become detectable.

The state estimation results also support the design of the pipeline. The measured and filtered position plots show that the Kalman Filter follows the detected vehicle motion while maintaining a continuous state. In both the single-vehicle and two-vehicle runs, the estimated position follows the main motion pattern of the vehicles and smooths part of the frame-to-frame variation. The velocity estimates provide an interpretable description of vehicle motion. The acceleration estimates are more variable, which is expected because acceleration is inferred indirectly from noisy image-based position measurements and is more sensitive to small detection changes.

Overall, the results show that the proposed off-board perception pipeline is able to detect, track, and estimate the state of RC vehicles from an external camera view. The single-vehicle runs validate the basic and hybrid tracking behaviour, while the two-vehicle run shows that the same architecture can maintain two simultaneous vehicle tracks under the tested conditions.

6.7 Limitations

The results presented in this chapter have several limitations. First, the evaluation is based on a limited number of laboratory sequences. The results should therefore be interpreted as prototype validation under controlled experimental conditions rather than as a complete statistical benchmark.

Second, a complete frame-by-frame ground truth annotation was not available. For this reason, formal detection and multi-object tracking metrics such as precision, recall, MOTA, MOTP, and IDF1 were not computed. The analysis instead relies on logged system outputs, visual inspection, track continuity, missed-frame behaviour, trajectory plots, and state estimation results.

Third, the metric trajectory depends on the calibration assumptions used to convert image coordinates into physical units. If the camera is not perfectly top-down or if perspective effects are significant, a simple scale-based conversion can introduce spatial error. A more complete calibration using a planar homography would improve metric accuracy and make the estimated positions more reliable in physical units.

Fourth, AprilTag detection depends on several practical factors, including apparent marker size, focus, motion blur, illumination, viewing angle, and image resolution. The experiments show this clearly. The 5 cm marker was not decoded reliably in the current 1280×720 configuration, while the 10 cm and 12 cm markers were detected in the tested sequences. This confirms that the tag-based component is useful as identity support, but it is sensitive to the acquisition setup.

Fifth, the two-vehicle experiment validates the pipeline in a simple multi-vehicle scenario, but it does not cover all possible association challenges. More demanding cases, such as vehicles driving very close to each other, crossing paths, partial occlusions, or longer periods of missed detections, would be needed to fully evaluate the robustness of the data association logic.

Finally, the state estimation results are affected by the quality of the visual measurements. The Kalman Filter provides smoother and more continuous estimates, but it cannot fully remove the effect of noisy detections, abrupt changes in direction, or temporary measurement instability. This is especially visible in acceleration, which is more sensitive than position or speed because it is derived indirectly from image-based measurements.

6.8 Chapter Summary

This chapter presented the results obtained with the implemented off-board perception pipeline. The evaluation included two single-vehicle runs and one two-vehicle run.

The first run validated the vision-based pipeline using YOLO detections, tracking, and Kalman-based state estimation without AprilTag support. The system maintained a stable internal track and recovered from a short missed-detection interval through Kalman prediction.

The second run validated the hybrid behaviour using a larger AprilTag marker. In this sequence, the marker was decoded consistently after the initial seconds, the internal track identifier remained stable, and the missed-frame counter stayed at zero. This showed that AprilTag information can support identity assignment when the marker is visible and sufficiently large in the image.

The two-vehicle run evaluated the pipeline in a simple multi-vehicle scenario. The system maintained two simultaneous active tracks, preserved the internal identifiers Track 1 and Track 2, and produced separate trajectory and state estimates for both vehicles. In this run, the 10 cm AprilTag associated with Track 1 was detected, while the 5 cm marker associated with Track 2 was not decoded. Track 2 was still maintained through YOLO-based localization and Kalman-based state estimation.

The comparison between the marker sizes shows that AprilTag availability is strongly influenced by apparent marker size in the image. In the current setup, the 5 cm marker did not provide enough reliable visual detail for decoding, while the 10 cm and 12 cm markers were successfully detected in the evaluated runs. The results therefore support the adopted hybrid strategy: YOLO provides the main localization measurement, the Kalman Filter maintains temporal consistency, and AprilTags provide auxiliary identity information when they are successfully decoded.

Chapter 7

Conclusions and Future Work

This chapter concludes the dissertation by summarizing the work developed, the main findings obtained from the experimental evaluation, the limitations of the current prototype, and possible directions for future work. The goal is not to repeat the results presented in the previous chapter, but to reflect on what the implemented system was able to demonstrate and what still needs to be improved.

7.1 Conclusions

This dissertation addressed the development of an off-board vision-based perception pipeline for small-scale RC vehicle tracking. The proposed system was designed to detect vehicles from an external camera view, maintain consistent tracks over time, use AprilTags as auxiliary identity information when available, and estimate motion-related quantities using a Kalman Filter.

The work began with the definition of an off-board acquisition architecture, where a fixed external camera observes the racing area and the video stream is processed on an external computer. This configuration provides a global view of the track and allows the perception system to estimate vehicle motion from a shared reference frame. The final setup used an external Universal Serial Bus (USB) camera connected to a Raspberry Pi, with the video stream transmitted through a wired Ethernet connection to the processing computer.

The implemented perception pipeline combines several components. YOLO is used as the main source of vehicle localization, providing bounding boxes and confidence values for the detected vehicles. AprilTags are integrated as auxiliary identity cues, allowing the system to associate a decoded marker identifier with a vehicle track when the tag is visible and reliably detected. The tracking module associates detections over time, manages track creation and removal, and uses Kalman-based prediction to maintain continuity during short missed-detection intervals.

The experimental evaluation showed that the system was able to maintain stable single-vehicle tracking under the tested conditions. In the vision-based run, the system preserved the same internal track identity using YOLO detections and Kalman prediction, even without AprilTag support. This confirmed that the tracking pipeline does not depend on fiducial markers to operate. In the hybrid

run, the larger 12 cm AprilTag was detected and used as identity support while the internal track remained stable. This showed that the proposed architecture can combine visual detection with marker-based identity information when the marker occupies enough pixels in the image.

The dissertation also evaluated the system in a two-vehicle run. In this experiment, one vehicle was equipped with a 10 cm AprilTag and the other with a 5 cm AprilTag. The system maintained two simultaneous internal tracks and produced separate trajectory and state estimates for both vehicles. The 10 cm marker was decoded and provided identity support for Track 1, while the 5 cm marker associated with Track 2 was not detected. Even so, the second vehicle remained tracked through YOLO-based localization and Kalman-based state estimation. This result shows that the implemented pipeline can operate in a simple multi-vehicle scenario and that the tag component remains auxiliary rather than essential for vehicle localization.

The results highlighted the importance of apparent marker size for AprilTag detection. The 5 cm marker was not detected reliably in the 1280×720 stream, while the 10 cm and 12 cm markers were successfully decoded in the evaluated runs. This indicates that the limitation was mainly related to image resolution, camera geometry, and marker size in the image, rather than to the structure of the hybrid tracking approach itself.

The Kalman Filter provided a useful state representation for the vehicles. Position estimates followed the visual measurements while maintaining temporal continuity, and velocity estimates provided an interpretable description of vehicle motion. Acceleration estimates were more sensitive to noise, as expected, since they are inferred indirectly from image-based measurements and affected by small variations in position and frame timing.

Overall, the developed prototype demonstrates the feasibility of an off-board perception pipeline for RC racing experiments. It provides vehicle detection, track continuity, auxiliary identity support through AprilTags when available, and state estimation from visual measurements. The system is therefore a useful basis for future work involving more demanding multi-vehicle tracking, more robust identity management, and integration with planning or control modules.

7.2 Main Contributions

The main contributions of this dissertation are mainly practical and experimental. First, the work implements an off-board perception architecture for RC vehicle tracking using an external camera and a separate processing computer. This architecture provides the basis for monitoring the racing track from a global viewpoint and for processing the acquired video stream outside the vehicles.

Building on this architecture, the dissertation integrates YOLO-based vehicle detection as the main source of visual localization. AprilTags are then used as auxiliary identity information within the tracking pipeline, allowing detected markers to support vehicle identification when they are visible and correctly decoded. Together, these two detection sources provide complementary information for vehicle localization and identity support.

Another contribution is the development of a tracking module that combines data association, track management, and Kalman-based prediction. This module is responsible for maintaining

consistent vehicle tracks over time, handling missed detections, and reducing unnecessary track fragmentation. Based on the image measurements associated with each active track, the system also estimates vehicle position, velocity, and acceleration, providing a more complete representation of vehicle motion.

The dissertation further contributes with structured logs and visual outputs that support the experimental analysis of detection, tracking, and state estimation behaviour. These outputs are used to evaluate the system in vision-based, hybrid YOLO–AprilTag, and two-vehicle configurations. Finally, the experiments also analyse the practical effect of AprilTag marker size, using 5 cm, 10 cm, and 12 cm markers to assess how marker visibility affects identity support in the proposed setup.

These contributions are mostly practical and experimental. The dissertation does not propose a new detection model or a new filtering method. Instead, it combines established computer vision and estimation techniques into a working prototype adapted to a small-scale racing environment.

7.3 Limitations

Although the implemented system achieved the main objectives of the dissertation, some limitations remain.

First, the evaluation was performed using a limited number of experimental sequences. The results are useful for validating the prototype, but they do not represent a complete benchmark across all possible operating conditions. A larger dataset would be needed to evaluate the system more systematically.

Second, the experiments did not include a complete frame-by-frame ground truth annotation. As a result, standard object detection and multi-object tracking metrics, such as precision, recall, Multiple Object Tracking Accuracy (MOTA), Multiple Object Tracking Precision (MOTP), and Identification F1 Score (IDF1), were not computed. The evaluation relied instead on logged indicators, track continuity, missed-frame behaviour, trajectory plots, state estimation results, and visual inspection.

Third, the two-vehicle run validates the pipeline in a simple multi-vehicle scenario, but it does not cover all possible association challenges. More demanding cases, such as vehicles moving very close to each other, crossing paths, overlapping in the image, or remaining undetected for longer periods, still require further evaluation.

Fourth, AprilTag detection was sensitive to the apparent marker size in the image. The experiments showed that the 5 cm marker was not decoded reliably in the current 1280×720 configuration, while the 10 cm and 12 cm markers were detected in the tested sequences. This confirms that the tag-based component is useful as identity support, but it depends strongly on the acquisition setup.

Fifth, lighting conditions affected detection reliability. Reflections and direct sunlight on the track surface can reduce image quality and temporarily affect visual detections. Although the tracker was able to recover from short detection gaps in the evaluated sequences, more controlled illumination conditions would improve repeatability.

Finally, the metric conversion used in the current prototype is limited by the calibration assumptions. If the camera view is not perfectly top-down or if perspective effects are significant, a simple scale-based conversion may introduce spatial errors. A more complete calibration method would improve the accuracy of metric position estimates.

7.4 Future Work

Several improvements can be explored in future work.

A first direction is the evaluation of more demanding multi-vehicle scenarios. The current pipeline was validated with two vehicles under stable conditions, but additional experiments are needed to test identity preservation when vehicles move close to each other, cross similar regions of the track, or temporarily occlude one another. This would make it possible to evaluate the robustness of the association logic under more realistic racing conditions.

A second direction is the improvement of camera calibration. Future work could use a planar homography based on known reference points on the track. This would allow image coordinates to be converted into metric coordinates more accurately, especially if the camera is not perfectly aligned with the track plane.

A third improvement concerns AprilTag detection. The experiments showed that marker size, resolution, and camera placement have a strong effect on detectability. Future setups could use a higher-resolution stream, improved focus, better lighting control, or optimized marker placement on the vehicle. These changes would make it possible to use smaller physical markers while preserving sufficient apparent size in the image.

The detection model could also be improved. Additional training data collected under different lighting conditions, vehicle orientations, and speeds would make YOLO detections more robust. Data augmentation could also help the detector handle reflections, shadows, and motion blur more reliably.

Another possible improvement is the refinement of the tracking and association strategy. The current system uses visual detections, motion prediction, and AprilTag identity information, but future versions could include more advanced appearance features or probabilistic association methods. This would be especially useful in multi-vehicle scenarios where vehicles may be close together or visually similar.

The state estimation module could also be extended. The current Kalman Filter estimates position, velocity, and acceleration from visual measurements. Future work could compare different motion models, tune process and measurement noise parameters more systematically, or include additional sensors if available. This could improve velocity and acceleration estimates, especially during rapid manoeuvres.

Finally, the perception pipeline could be integrated with planning and control modules. In the current dissertation, the focus was on perception and state estimation. A natural next step would be to use the estimated vehicle state as input for closed-loop control, trajectory planning, or

autonomous racing strategies. This would allow the off-board perception system to become part of a complete autonomous RC racing platform.

7.5 Final Remarks

The work developed in this dissertation shows that an off-board vision-based perception pipeline can provide useful tracking and state estimation information for small-scale RC racing. The system maintained stable single-vehicle tracks, handled short missed-detection intervals, and was also able to maintain two simultaneous vehicle tracks in a simple multi-vehicle run.

The comparison between vision-based, hybrid, and two-vehicle runs clarified the role of each component in the proposed architecture. YOLO detections provide the main localization measurement, the Kalman Filter supports temporal continuity, and AprilTags can provide additional identity information when they are visible and occupy enough pixels in the image. The experiments with 5 cm, 10 cm, and 12 cm markers also showed that marker size is a practical constraint in the current acquisition setup.

The final prototype provides a solid foundation for future development. With improved calibration, higher-resolution acquisition, stronger multi-vehicle evaluation, and integration with control modules, the system can be extended toward a more complete off-board autonomous racing platform.

References

[1] V. M. Perdigão, C. G. da Costa, and L. T. Paiva, "Real–time multi–vehicle tracking of autonomous vehicles using vision–based perception," [Submitted] CONTROLO 2026, 2026.

[2] P. Engström, *High performance autonomous racing with rc cars*, Bachelor's thesis, Chalmers University of Technology, Gothenburg, Sweden, 2018. [Online]. Available: https://odr.chalmers.se/items/855fdb11-e0e2-46b3-aba7-760c26e637e5.

[3] M. O'Kelly, H. Zheng, D. Karthik, and R. Mangharam, "F1/10: An open-source autonomous cyber-physical platform," *arXiv preprint arXiv:1901.08567*, 2019. DOI: 10.48550/arXiv.1901.08567. [Online]. Available: https://arxiv.org/abs/1901.08567.

[4] U. Rosolia, X. Zhang, and F. Borrelli, "Autonomous racing using learning model predictive control," *arXiv preprint arXiv:1804.06155*, 2018. [Online]. Available: https://arxiv.org/abs/1804.06155.

[5] J. Kabzan et al., "AMZ driverless: The full autonomous racing system," *arXiv preprint arXiv:1905.05150*, 2019. [Online]. Available: https://arxiv.org/abs/1905.05150.

[6] A. Liniger, A. Domahidi, and M. Morari, "Optimization-based autonomous racing of 1:43 scale rc cars," *Optimal Control Applications and Methods*, vol. 36, no. 5, pp. 628–647, 2015. DOI: 10.1002/oca.2123.

[7] B. Goldfain et al., "AutoRally: An open platform for aggressive autonomous driving," *IEEE Control Systems Magazine*, vol. 39, no. 1, pp. 26–55, 2019. DOI: 10.1109/MCS.2018.2876958.

[8] G. Williams, B. Goldfain, P. Drews, J. M. Rehg, and E. A. Theodorou, "Autonomous racing with AutoRally vehicles and differential games," *arXiv preprint arXiv:1707.04540*, 2017. [Online]. Available: https://arxiv.org/abs/1707.04540.

[9] R. Szeliski, *Computer Vision: Algorithms and Applications*. London: Springer, 2010, ISBN: 978-1-84882-935-0. DOI: 10.1007/978-1-84882-935-0. [Online]. Available: https://link.springer.com/book/10.1007/978-1-84882-935-0.

[10] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the KITTI vision benchmark suite," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 3354–3361. [Online]. Available: https://www.cvlibs.net/datasets/kitti/.

[11] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 11, pp. 1330–1334, 2000. DOI: 10.1109/34.888718.

[12] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, 2004, ISBN: 9780521540513. DOI: 10.1017/CBO9780511811685.

[13] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of Basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960. DOI: 10.1115/1.3662552. [Online]. Available: https://asmedigitalcollection.asme.org/fluidsengineering/article/82/1/35/397706/A-New-Approach-to-Linear-Filtering-and-Prediction.

[14] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, "Simple online and realtime tracking," in *Proceedings of the IEEE International Conference on Image Processing*, 2016, pp. 3464–3468. DOI: 10.1109/ICIP.2016.7533003. [Online]. Available: https://arxiv.org/abs/1602.00763.

[15] D. Comaniciu, V. Ramesh, and P. Meer, "Kernel-based object tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 5, pp. 564–577, 2003. DOI: 10.1109/TPAMI.2003.1195991.

[16] M. Fiala, "ARTag, a fiducial marker system using digital techniques," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, 2005, pp. 590–596. DOI: 10.1109/CVPR.2005.74.

[17] E. Olson, "AprilTag: A robust and flexible visual fiducial system," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2011, pp. 3400–3407. DOI: 10.1109/ICRA.2011.5979561. [Online]. Available: https://april.eecs.umich.edu/media/pdfs/olson2011tags.pdf.

[18] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and M. J. Marín-Jiménez, "Automatic generation and detection of highly reliable fiducial markers under occlusion," *Pattern Recognition*, vol. 47, no. 6, pp. 2280–2292, 2014. DOI: 10.1016/j.patcog.2014.01.005.

[19] J. Wang and E. Olson, "AprilTag 2: Efficient and robust fiducial detection," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2016, pp. 4193–4198. DOI: 10.1109/IROS.2016.7759617.

[20] J. Kallwies, B. Forkel, and S. Scherer, "Determining and improving the localization accuracy of AprilTag detection," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2020. DOI: 10.1109/ICRA40945.2020.9196649.

[21] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015. DOI: 10.1038/nature14539.

[22] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems*, vol. 28, 2015. [Online]. Available: https://arxiv.org/abs/1506.01497.

[23] W. Liu et al., "SSD: Single shot multibox detector," in *Computer Vision – ECCV 2016*, Springer, 2016, pp. 21–37. DOI: 10.1007/978-3-319-46448-0_2.

[24] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 779–788. DOI: 10.1109/CVPR.2016.91. [Online]. Available: https://arxiv.org/abs/1506.02640.

[25] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal speed and accuracy of object detection," *arXiv preprint arXiv:2004.10934*, 2020. [Online]. Available: https://arxiv.org/abs/2004.10934.

[26] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *Computer Vision – ECCV 2020*, Springer, 2020, pp. 213–229. DOI: 10.1007/978-3-030-58452-8_13. [Online]. Available: https://arxiv.org/abs/2005.12872.

[27] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018. [Online]. Available: https://arxiv.org/abs/1804.02767.

[28] G. Jocher, A. Chaurasia, and J. Qiu, *Ultralytics YOLO*, GitHub repository, 2023. [Online]. Available: https://github.com/ultralytics/ultralytics.

[29] J. Terven and D. Cordova-Esparza, "A comprehensive review of YOLO architectures in computer vision: From YOLOv1 to YOLOv8 and YOLO-NAS," *arXiv preprint arXiv:2304.00501*, 2023. [Online]. Available: https://arxiv.org/abs/2304.00501.

[30] N. Wojke, A. Bewley, and D. Paulus, "Simple online and realtime tracking with a deep association metric," in *Proceedings of the IEEE International Conference on Image Processing*, 2017, pp. 3645–3649. DOI: 10.1109/ICIP.2017.8296962. [Online]. Available: https://arxiv.org/abs/1703.07402.

[31] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval Research Logistics Quarterly*, vol. 2, no. 1–2, pp. 83–97, 1955. DOI: 10.1002/nav.3800020109.

[32] Y. Zhang et al., "ByteTrack: Multi-object tracking by associating every detection box," in *Computer Vision – ECCV 2022*, Springer, 2022, pp. 1–21. DOI: 10.1007/978-3-031-20047-2_1. [Online]. Available: https://arxiv.org/abs/2110.06864.

[33] J. Cao, J. Pang, X. Weng, R. Khirodkar, and K. Kitani, "Observation-centric SORT: Rethinking SORT for robust multi-object tracking," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 9686–9696. [Online]. Available: https://arxiv.org/abs/2203.14360.

[34] A. Milan, L. Leal-Taixé, I. Reid, S. Roth, and K. Schindler, "MOT16: A benchmark for multi-object tracking," *arXiv preprint arXiv:1603.00831*, 2016. [Online]. Available: https://arxiv.org/abs/1603.00831.