Scientific Research Publishing

Open Journal of Applied Sciences, 2024, 14, 3569-3588
https://www.scirp.org/journal/ojapps
ISSN Online: 2165-3925
ISSN Print: 2165-3917

Research on Vehicle Tracking Method Based on YOLOv8 and Adaptive Kalman Filtering: Integrating SVM Dynamic Selection and Error Feedback Mechanism

Liping Zheng[1*], Hao Gou[2], Kaiwen Xiao[1], Moran Qiu[1]

[1]School of Computer Science, Sichuan University Jinjiang College, Meishan, China
[2]School of Automotive and Transportation, Xihua University, Yibin, China
Email: *3233751594@qq.com

How to cite this paper: Zheng, L.P., Gou, H., Xiao, K.W. and Qiu, M.R. (2024) Research on Vehicle Tracking Method Based on YOLOv8 and Adaptive Kalman Filtering: Integrating SVM Dynamic Selection and Error Feedback Mechanism. *Open Journal of Applied Sciences*, **14**, 3569-3588.
https://doi.org/10.4236/ojapps.2024.1412234

Received: November 26, 2024
Accepted: December 16, 2024
Published: December 19, 2024

Abstract

Vehicle tracking plays a crucial role in intelligent transportation, autonomous driving, and video surveillance. However, challenges such as occlusion, multi-target interference, and nonlinear motion in dynamic scenarios make tracking accuracy and stability a focus of ongoing research. This paper proposes an integrated method combining YOLOv8 object detection with adaptive Kalman filtering. The approach employs a support vector machine (SVM) to dynamically select the optimal filter (including standard Kalman filter, extended Kalman filter, and unscented Kalman filter), enhancing the system's adaptability to different motion patterns. Additionally, an error feedback mechanism is incorporated to dynamically adjust filter parameters, further improving responsiveness to sudden events. Experimental results on the KITTI and UA-DETRAC datasets demonstrate that the proposed method significantly improves detection accuracy (mAP@0.5 increased by approximately 3%), tracking accuracy (MOTA improved by 5%), and system robustness, providing an efficient solution for vehicle tracking in complex environments.

Keywords

Multi-Target Tracking, YOLOv8-Based Detection, Adaptive Filtering, Support Vector Machine, Error Feedback Mechanism

1. Introduction

Vehicle tracking technology finds extensive applications in intelligent transportation systems, autonomous driving, and video surveillance [1]. Its core task is to

precisely localize and continuously track target vehicles, supporting key functions such as path planning, traffic flow analysis, and behavior prediction [2].

In autonomous driving, accurate vehicle tracking enables vehicles to identify dynamic obstacles in complex environments, facilitating safe and efficient decision-making. In traffic surveillance, it aids in detecting traffic violations, monitoring congestion, and improving road efficiency.

However, the highly dynamic and unpredictable nature of real-world environments presents numerous challenges for vehicle tracking:

1) Occlusion Problem

Occlusion is a common issue in vehicle tracking within multi-vehicle scenarios [3]. For instance, when a vehicle is partially or fully obscured by other vehicles, pedestrians, or static objects, the continuity of detection and tracking may be severely affected.

2) Dynamic Scenarios and Complex Backgrounds

In densely populated dynamic traffic scenes, vehicle motion can exhibit nonlinear behaviors such as acceleration, deceleration, and turning. Additionally, complex backgrounds—including multi-target interference, strong lighting, or shadows—can significantly reduce the accuracy of detection and tracking [4].

3) Error Accumulation

In traditional tracking frameworks, discrepancies may arise between detection results and trajectory predictions by filters. When tracking persists over time or the dynamics of the scene intensify, these errors can accumulate, ultimately leading to tracking failures.

To overcome these challenges, it is crucial to develop vehicle tracking methods that offer high accuracy, strong robustness, and dynamic adaptability. This paper proposes an integrated approach combining object detection with trajectory prediction. The method utilizes the deep learning-based object detection model YOLOv8 [5] and an adaptive filter selector to achieve efficient and robust vehicle tracking.

To address the limitations of existing methods, this paper introduces two key innovations:

1) SVM-Based Adaptive Filter Selection Mechanism

Vehicle motion characteristics (e.g., constant velocity, acceleration, nonlinear movement) vary across different scenarios, making it challenging for a single filter to effectively handle diverse conditions. To address this, an adaptive filter selection mechanism based on Support Vector Machine (SVM) is designed. This mechanism dynamically selects the optimal filter—standard Kalman Filter (KF), Extended Kalman Filter (EKF), or Unscented Kalman Filter (UKF)—based on the current motion characteristics of the vehicle. This approach significantly enhances tracking accuracy and robustness in complex and dynamic environments.

2) Integration of Detection and Tracking Error Feedback Mechanism

To minimize discrepancies between detection and prediction, an error feedback mechanism is proposed. This mechanism dynamically feeds the error between YOLOv8 detection results and filter predictions back to the filter parameter ad-

justment module. By adaptively tuning the process noise covariance and measurement noise covariance, the filter can better handle sudden events (e.g., abrupt acceleration or sharp turns), thereby improving the consistency and stability of tracking.

2. Related Work

Vehicle tracking technology is a key research area in intelligent transportation, autonomous driving, and security fields. Its main objective is to accurately locate the target vehicle and predict its motion trajectory. Currently, vehicle tracking methods are mainly divided into two categories: deep learning-based object detection methods [6] and classical filtering-based trajectory prediction methods [7]. This section reviews these two approaches and discusses their advantages and limitations based on the latest research.

2.1. Deep Learning-Based Object Detection Methods

With the development of deep learning, vehicle tracking methods based on object detection have gradually become mainstream. Object detectors generate candidate bounding boxes by detecting each video frame, and vehicle trajectories are formed using subsequent association strategies.

1) YOLO Series Models

The YOLO (You Only Look Once) model, proposed by Redmon *et al.* [8], pioneered single-stage object detection. The current popular version, YOLOv8, optimizes the network structure and feature fusion, demonstrating excellent accuracy and speed in vehicle detection tasks [9] [10].

The advantage of the YOLOv8 model is its ability to achieve real-time vehicle detection with a high frame rate, performing well even in occlusion and low-light conditions. However, its detection results are highly dependent on the diversity of the training data, and there is limited coordination with subsequent tracking modules.

2) Faster R-CNN

Faster R-CNN, introduced by Ren *et al.* [11], significantly improves detection accuracy by incorporating a Region Proposal Network (RPN). Zhang Ying *et al.* [12] designed a vehicle detection solution based on the Faster R-CNN model for unmanned aerial vehicle (UAV) platforms. Ouyang Bo *et al.* [13] proposed a lightweight tracking model, FA-SORT, which achieved a tracking speed of 29.93 FPS when tested on the UAVDT dataset.

The advantage of Faster R-CNN lies in its high accuracy, making it particularly suitable for complex backgrounds and dense target scenarios. However, its computational complexity is relatively high, making it difficult to meet real-time processing requirements.

2.2. Kalman Filtering and Its Variants

Classic filtering methods, represented by the Kalman filter, estimate and update vehicle trajectories by combining detection results with predicted states. These

methods are typically used in conjunction with object detectors for multi-target tracking tasks.

1) Kalman Filter (KF)

The Kalman Filter (KF) is an efficient linear state estimation method widely used for vehicle tracking in scenarios involving constant or linear motion. Gordon *et al.* [14] applied KF in real-time traffic flow monitoring, significantly improving the accuracy of vehicle trajectory predictions. However, KF performs poorly when dealing with nonlinear or complex motion.

2) Extended Kalman Filter (EKF)

The Extended Kalman Filter (EKF) extends the KF to nonlinear systems by performing a first-order linearization, making it suitable for a wider range of applications. Reid *et al.* [15] applied EKF to multi-target tracking, achieving high accuracy in vehicle scenarios involving rapid acceleration and sharp turns. However, its performance in highly nonlinear scenarios is still limited by linearization errors.

3) Unscented Kalman Filter (UKF)

The Unscented Kalman Filter (UKF) improves trajectory prediction accuracy in nonlinear systems by using sigma-point sampling, eliminating the need for linearization. The UKF algorithm, proposed by van der Merwe *et al.* [16], demonstrates excellent performance in dynamic target tracking. Farag *et al.* [17] combined UKF with deep detectors to perform multi-target tracking in complex traffic environments.

2.3. Integrated Approaches Combining Deep Learning and Classical Filtering

In recent years, researchers have explored the deep integration of object detection and filtering techniques to optimize vehicle tracking performance. Bui T *et al.* [18] proposed a tracking framework that combines YOLOv5 with an optimized Kalman filter (KF) algorithm in DeepSORT, achieving a 15.5% improvement in mAP and a 14.2% increase in MOTA. Gao J *et al.* [19] reduced the impact of observation noise on detection accuracy during nonlinear motion by incorporating a lightweight channel block attention mechanism (LCBAM) and noise-adaptive Extended Kalman Filter (NSA-EKF). These methods demonstrate the complementarity of deep detectors and classical filtering techniques in vehicle tracking tasks, making them a current research hotspot.

Deep learning-based object detection methods offer efficient target recognition capabilities, while classical filtering methods provide irreplaceable advantages in state estimation and trajectory prediction. Integrating the strengths of both approaches into a unified tracking framework is an effective solution for vehicle tracking in complex dynamic environments.

3. Methodology

3.1. YOLOv8 Detection Module

In this study, YOLOv8 is responsible for the initial vehicle detection, providing

high-quality target information for the subsequent tracking module. This information is then used to enhance the overall tracking performance through Kalman filtering and error feedback mechanisms. The following section outlines the configuration and structural optimizations of the YOLOv8 model used in this research.

1) Overall Framework Configuration

In this study, YOLOv8 still uses CSPDarknet as the backbone network, incorporating a bottleneck structure (Bottleneck CSP) to reduce redundant computations and lower model complexity, thereby improving detection speed while maintaining accuracy. Additionally, multi-scale feature fusion modules (FPN and PAN) are introduced in the structure to enhance the model's performance in detecting vehicles of varying sizes. Furthermore, the application of depthwise separable convolutions significantly reduces the computational cost of YOLOv8, enabling real-time processing. Finally, to improve detection accuracy, a reinforced data augmentation strategy is employed during YOLOv8 training, including random scaling, translation, flipping, and color adjustment, which enhances the model's robustness and generalization ability. The overall framework of YOLOv8 is shown in Figure 1.

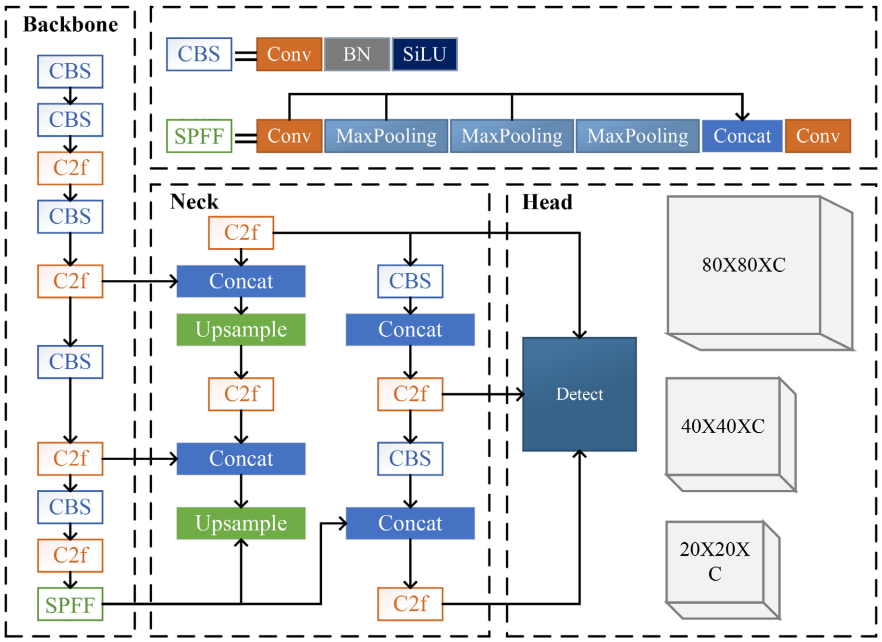

Figure 1. YOLOv8 structural frame diagram.

2) Input and Output Formats

The input format for the YOLOv8 detection module is a standard RGB image, which is normalized and resized to a specific dimension (e.g., 640 × 640) before being fed into the model. This ensures consistent detection accuracy across images with different resolutions. During the input process, YOLOv8 also incorporates anchor box adaptive adjustment techniques, allowing the model to more flexibly

accommodate vehicles of various sizes and perspectives.

The output format of YOLOv8 includes the coordinates of the predicted bounding boxes, target categories, and confidence scores. For vehicle detection tasks, the model outputs the vehicle's bounding box positions (x, y, width, height) along with the probability distribution for the corresponding category (e.g., car, truck, motorcycle), where the confidence score is used to filter out low-confidence targets. After post-processing with Non-Maximum Suppression (NMS), overlapping redundant boxes are eliminated to ensure the accuracy of the output results. In this study, the detection boxes output by YOLOv8 are directly fed into the Kalman filter module for further trajectory prediction and state estimation.

3.2. Kalman Filtering and Its Improvements

In vehicle tracking tasks, the Kalman filter is a commonly used state estimation method that combines observation and prediction to estimate the object's position and motion state. While the standard Kalman filter (KF) performs excellently in linear systems with Gaussian noise, it has limitations in dynamic, nonlinear, and non-Gaussian environments. Therefore, this study introduces several variants of the Kalman filter, including the Extended Kalman Filter (EKF), Unscented Kalman Filter (UKF), and Interactive Multiple Model (IMM) filter. Additionally, an adaptive filter selection mechanism based on Support Vector Machine (SVM) is proposed to dynamically select the optimal filter in different scenarios, thus optimizing detection and tracking performance.

3.2.1. Mathematical Models and Characteristics of Kalman Filter Variants

1) Standard Kalman Filter (KF)

The standard Kalman filter assumes that both the system state model and the measurement model are linear and that the noise follows a Gaussian distribution. The state estimation process consists of two stages: prediction and update.

The state prediction equations are given by the following Equation (1) and (2).

$$\hat{x}_{k|k-1} = F_k \hat{x}_{k-1|k-1} + B_k u_k \tag{1}$$

$$P_{k|k-1} = F_k P_{k-1|k-1} F_k^{\mathrm{T}} + Q_k \tag{2}$$

The state update equations are given by the following Equation (3), (4) and (5).

$$K_k = P_{k|k-1} H_k^{\mathrm{T}} \left(H_k P_{k|k-1} H_k^{\mathrm{T}} + R_k \right)^{-1} \tag{3}$$

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k \left(z_k - H_k \hat{x}_{k|k-1} \right) \tag{4}$$

$$P_{k|k} = \left(I - K_k H_k \right) P_{k|k-1} \tag{5}$$

Wherein, F_k represents the state transition matrix, H_k is the measurement matrix, Q_k and R_k denote the process noise and measurement noise covariance matrices, respectively, and P_k is the estimation error covariance matrix.

2) Extended Kalman Filter (EKF)

The Extended Kalman Filter (EKF) extends the linear assumption of the standard Kalman Filter (KF) to accommodate mildly nonlinear systems. The EKF

achieves approximate linearization by performing a first-order Taylor expansion on the nonlinear state transition function and observation function.

The equations for the state prediction phase (nonlinear) are given by Equations (6) and (7) as follows.

$$\hat{x}_{k|k-1} = f\left(\hat{x}_{k-1|k-1}\right) + B_k u_k \tag{6}$$

$$P_{k|k-1} = F_k P_{k-1|k-1} F_k^{\mathrm{T}} + Q_k \tag{7}$$

The state update equations are given by the following Equation (8), (9) and (10).

$$K_k = P_{k|k-1} H_k^{\mathrm{T}} \left(H_k P_{k|k-1} H_k^{\mathrm{T}} + R_k\right)^{-1} \tag{8}$$

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k \left(z_k - h\left(\hat{x}_{k|k-1}\right)\right) \tag{9}$$

$$P_{k|k} = \left(I - K_k H_k\right) P_{k|k-1} \tag{10}$$

In this context, $f(\cdot)$ and $h(\cdot)$ represent the nonlinear functions for state transition and observation, respectively.

3) Unscented Kalman Filter (UKF)

The Unscented Kalman Filter (UKF) addresses the state estimation problem in nonlinear systems through a set of weighted sample points known as "sigma points." Without the need for linearization, the UKF achieves higher-precision state estimation under stronger nonlinear conditions.

Sigma Point Generation and Propagation: A set of weighted sigma points is generated, and the propagation results of each sigma point are used to update the mean and covariance, thereby calculating the estimated state distribution.

3.2.2. SVM-Based Adaptive Filter Selection Mechanism

In complex vehicle tracking scenarios, a single filter struggles to cope with all situations due to the presence of different motion patterns (such as uniform speed, acceleration, nonlinear trajectories, etc.). Therefore, an adaptive filter selection mechanism based on Support Vector Machines (SVMs) is proposed. This mechanism utilizes SVMs to classify motion patterns in the input feature space, thereby selecting the most suitable Kalman filter variant according to the characteristics of the scene. This dynamic selection can significantly enhance the adaptability of the filter and ensure the accuracy and robustness of the tracking process.

The design of the SVM-based adaptive filter selection mechanism can be divided into three main steps: feature extraction, SVM classifier training, and dynamic filter selection.

1) Feature Extraction

To achieve accurate filter selection, the system needs to extract features that can effectively distinguish between motion patterns. These features include the target's velocity (v), acceleration (a), and the rate of change in its motion trajectory (such as curvature changes). The distributions of these features under different motion patterns are typically distinct, serving as effective bases for classification. Therefore, we define the feature vector as shown in Equation (11):

$$\boldsymbol{X} = [v, a, \theta] \tag{11}$$

Wherein, v represents the instantaneous velocity of the target at the current moment; a denotes the acceleration of the target; and θ indicates the change angle of the movement direction (*i.e.*, steering angle or trajectory curvature).

2) SVM Classifier

Within the feature space, the feature vector $\boldsymbol{X}$ is input into the Support Vector Machine (SVM) classifier. The SVM utilizes the trained classification boundary to separate different motion patterns. Specifically, we train a multi-class SVM classifier that categorizes vehicle motion into three classes: uniform speed, acceleration, and complex nonlinear motion.

Assuming the training data is denoted as $D = \{(\boldsymbol{X}_i, y_i)\}_{i=1}^{N}$, where $\boldsymbol{X}_i$ represents the input features and y_i corresponds to the motion pattern labels (1 for uniform motion, 2 for accelerated motion, and 3 for nonlinear motion). The objective is to find an optimal classification boundary by solving the following optimization problem using SVM, as shown in Equations (12) and (13):

$$\min_{w,b} \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{N} \varepsilon_i \tag{12}$$

$$\text{subject to} \quad y_i\left(\boldsymbol{w} \cdot \varnothing(\boldsymbol{X}_i) + b\right) \geq 1 - \varepsilon_i,\ \varepsilon_i \geq 0,\ i = 1, \cdots, N \tag{13}$$

Wherein, W represents the classification weight vector, B is the bias, CC stands for the slack variable, C is the penalty parameter, and QQ denotes the kernel function used to map features into a high-dimensional space. By solving the aforementioned optimization problem, the SVM obtains the optimal separating hyperplane, enabling classification of newly input data points.

3) Filter Selection Strategy

Once the current motion pattern is determined, the system dynamically selects the corresponding Kalman filter variant based on the SVM classification result. The specific selection strategy is as follows:

- For uniform motion, the standard Kalman Filter (KF) is selected;
- For accelerated motion, the Extended Kalman Filter (EKF) is chosen;
- For nonlinear motion, the Unscented Kalman Filter (UKF) is utilized.

Therefore, given the input features X at the current time, the SVM outputs a label $y \in \{1, 2, 3\}$, and the corresponding filter is selected based on this label to enhance vehicle tracking performance.

Compared to a fixed filter, the SVM-based adaptive selection mechanism overcomes the limitations of a single filter in complex scenarios by dynamically switching filters, thereby improving tracking accuracy and stability.

Figure 2 illustrates the flowchart of the SVM-based adaptive filter selection mechanism.

Firstly, data preprocessing and feature extraction are conducted to compute the current speed, acceleration, and direction change angle from the target's historical trajectory, forming the feature vector X.

Subsequently, the feature vector X is input into an SVM classifier, which out-

puts the current motion mode label y (in a multi-class SVM, based on the trained classification boundaries, the SVM can effectively distinguish between uniform, accelerated, and complex nonlinear motions).

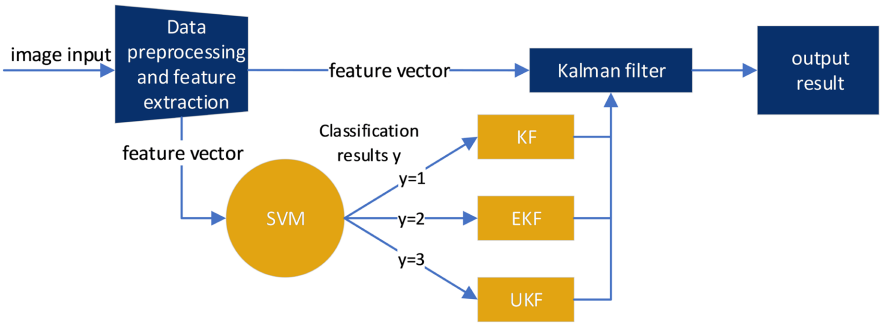

Figure 2. Flowchart of the adaptive filter selection mechanism.

Concurrently, based on the classification result y, the corresponding Kalman filter variant is selected. For instance, when $y = 1$, the system employs the standard Kalman Filter (KF); when $y = 2$, the Extended Kalman Filter (EKF) is used; and when $y = 3$, the Unscented Kalman Filter (UKF) is chosen.

Finally, with the selected filter, the YOLOv8 detection results are input as observations to the filter, and the updated target state is calculated to obtain more precise location information.

3.3. Error Feedback Mechanism

Apart from selecting the appropriate Kalman filter, dynamic changes or unexpected events in the environment (such as sudden acceleration, deceleration, occlusion, or unexpected direction changes of the target) can also lead to increased tracking errors. To enhance the system's responsiveness to these unexpected events, this study proposes an error feedback mechanism that dynamically adjusts the parameters of the Kalman filter by feeding back detection errors. This mechanism not only compensates for errors generated during detection and tracking but also adapts when the filter's response is poor, thereby improving the system's stability and accuracy in complex environments.

The design of the error feedback mechanism is based on dynamic feedback adjustment of detection errors and mainly consists of the following steps:

1) Error Calculation

In each frame, the detection error e_k is calculated based on the difference between the target position detected by YOLOv8 (detection result) and the predicted position by the Kalman filter (prediction result).

Assuming the detection result is z_k (observation) and the filter's prediction is $\hat{x}_{k|k-1}$, the error is defined as Equation (14):

$$e_k = z_k - \hat{x}_{k|k-1} \tag{14}$$

Wherein, e_k denotes the detection error vector at time k, encompassing in-

formation on the differences in position and velocity.

2) Error Weight Calculation

Different feedback intensities should be applied to detection errors in various scenarios. By introducing an error weight W_k, the errors are weighted accordingly. The error weight can be dynamically adjusted based on the magnitude of the detection error and the frequency of occurrence of unexpected events. For instance, a larger feedback weight is assigned to significantly increased errors.

The feedback weight can be defined by Equation (15):

$$W_k = \alpha \cdot \|e_k\| + \beta \tag{15}$$

Wherein, α and β are adjustable parameters, and $\|e_k\|$ represents the magnitude of the error. By adjusting the feedback weight, the filter's responsiveness to unexpected events can be enhanced.

3) Filter Gain Adjustment

The error is fed back to the process noise covariance Q_k and measurement noise covariance R_k of the Kalman filter. By dynamically adjusting Q_k and R_k, the filter can enhance its adaptability to different error scenarios, thereby improving tracking accuracy.

The adjustment model for the feedback mechanism can be expressed as Equations (16) and (17):

$$Q_k = Q_k + W_k \cdot (e_k e_k^T) \tag{16}$$

$$R_k = R_k + W_k \cdot (e_k e_k^T) \tag{17}$$

The error feedback adjusts the process noise Q_k and the measurement noise covariance matrix R_k, enabling the filter to adaptively adjust its gain matrix based on the current error.

Figure 3 illustrates the complete overall process of adaptive tracking.

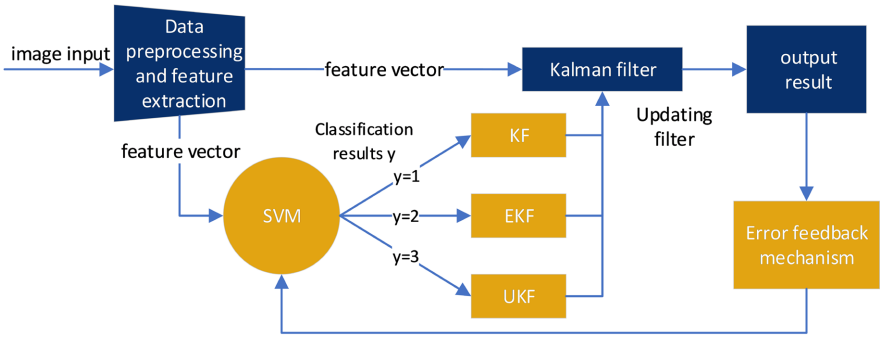

Figure 3. Overall flowchart of adaptive tracking.

4. Experimental Setup

4.1. Datasets and Evaluation Metrics

To comprehensively evaluate the performance of the vehicle detection and tracking system proposed in this study, two widely used standard datasets, KITTI [20] and UA-DETRAC [21], are selected. These datasets cover a variety of scenarios

(such as urban roads, highways, and adverse weather conditions) and different dynamic conditions (such as occlusion and low-light environments). Additionally, to quantify the model's performance, mean Average Precision (mAP), Multiple Object Tracking Accuracy (MOTA), and Frames Per Second (FPS) for real-time performance are adopted as the core evaluation metrics.

4.1.1. Dataset Selection

1) KITTI Dataset

The KITTI dataset is one of the standard datasets for autonomous driving research. It primarily consists of images from common traffic scenarios such as urban roads and highways, including 7481 training images and 7518 test images.

2) UA-DETRAC Dataset

The UA-DETRAC dataset focuses on vehicle detection and tracking tasks. It covers high-density traffic areas such as urban roads and intersections. It provides 10 hours of video data, comprising 1210 video clips and 140,000 annotated frames.

Table 1 summarizes the main characteristics of the KITTI and UA-DETRAC datasets:

Table 1. Table type styles (Table caption is indispensable).

Dataset	Task Type	Data Volume	Scenario Coverage	Challenges
KITTI	Detection & Tracking	7481 Training Images	Urban, Highways	Occlusion, Multi-target, Illumination Changes
UA-DETRAC	Detection & Tracking	1210 Video Clips	Urban, Traffic Intersections	Complex Background, Dense Targets, Low-light Scenarios

4.1.2. Evaluation Metrics

This paper quantitatively evaluates the system performance from both detection and tracking perspectives, mainly using the following metrics:

1) Mean Average Precision (mAP@0.5)

mAP is a core metric for object detection, used to measure the overall performance of a detection model across all categories. mAP@0.5 indicates the mean average precision calculated with an Intersection over Union (IoU) threshold of 0.5. Its calculation is shown in Equation (18):

$$\text{mAP}@0.5 = \frac{1}{N}\sum_{i=1}^{N} AP_i \tag{18}$$

where, N is the number of categories, and AP_i is the average precision for the i-th category.

2) Multiple Object Tracking Accuracy (MOTA)

MOTA is used to quantify the overall performance of a tracking task, taking into account object losses, mis-tracking, and ID switches. Its calculation is shown in Equation (19):

$$\text{MOTA} = 1 - \frac{\sum_t \left(FN_t + FP_t + IDS_t \right)}{\sum_t GT_t} \tag{19}$$

where, FN_t, FP_t, and IDS_t represent the number of false negatives, false positives, and ID switches at frame t, respectively, and GT_t represents the total number of ground truth objects at frame t.

3) Real-time Performance (FPS)

Frames Per Second (FPS) is used to measure the speed performance of the model and is an important indicator of real-time performance. A high FPS value indicates that the system is suitable for practical applications.

4.2. Experimental Parameters and Environment

The experiments in this study were conducted in a high-performance computing environment to ensure the real-time and reliable performance of vehicle detection and tracking tasks. The following details the software and hardware configurations of the experimental environment, as well as the specific hyperparameter settings for the YOLOv8 detection module and Kalman filter.

4.2.1. Software and Hardware Configuration

The software and hardware configurations used in the experiments are shown in Table 2:

Table 2. Software and hardware configuration table.

Software/Hardware Component	Configuration Details
Operating System	Windows 11
Graphics Card	NVIDIA RTX 3060 12GB
Memory	128GB DDR4
Python	3.9
PyTorch	1.12.1
CUDA	11.6

The GPU (RTX 3060) supports rapid training and inference for YOLOv8, and the sufficient memory ensures the loading and processing of the dataset. The detection part of the experiment is implemented based on the PyTorch framework, while the Kalman filter and data processing modules utilize NumPy for matrix operations.

4.2.2. YOLOv8 Hyperparameter Settings

The hyperparameters of the YOLOv8 model have been fine-tuned to ensure a bal-

ance between detection accuracy and speed. The specific configurations are shown in Table 3:

Table 3. YOLOv8 hyperparameter configuration table.

Parameter Names	Setting	Instructions
Image Size	640 × 640	Adjust all input images to a uniform size.
Confidence	0.5	Filter out detection bounding boxes with low confidence scores.
NMS (Non-Maximum Suppression)	0.45	Remove excessively overlapping detection bounding boxes.
Epochs	50	Number of iterations for model training.
Batch Size	16	Number of images input to the model per batch.
Learning Rate (Initial Value)	0.01	Dynamically adjust the learning rate using a cosine annealing strategy.
Optimizer	SGD	Optimize model parameters using stochastic gradient descent.
Data Augmentation	Open	Data augmentation techniques include random cropping, rotation, scaling, and color jittering.

These hyperparameter settings have been optimized on the KITTI and UA-DETRAC datasets, effectively enhancing the detection performance of YOLOv8 in various scenarios.

4.2.3. Kalman Filter Hyperparameter Settings

To adapt to dynamic scenarios in vehicle tracking, the initialization parameters for various Kalman filters are shown in Table 4:

Table 4. Kalman filter hyperparameter configuration table.

Filter Type	State Transition Covariance (Q)	Measurement Noise Covariance (R)	Initial Error Covariance (P)	State Dimension	Observation Dimension
KF	$0.1I$	$0.05I$	$1.0I$	4	2
EKF	$0.15I$	$0.05I$	$1.0I$	4	2
UKF	$0.2I$	$0.1I$	$1.0I$	4	2

Where Q, R, and P represent the process noise covariance matrix, measurement noise covariance matrix, and initial error covariance matrix, respectively; I de-

notes the identity matrix.

The hardware and software environments of the experiment provide powerful computational support for large-scale detection and tracking tasks. The optimized hyperparameters of YOLOv8 enable it to be efficient and robust in multiple scenarios, while the reasonable initialization and dynamic adjustment strategies of the Kalman filter ensure the accuracy and real-time performance of the tracking module.

5. Experimental Results and Analysis

5.1. Comparative Analysis of Detection and Tracking

5.1.1. Analysis of Detection Performance

Vehicle detection serves as the foundational step in achieving multi-object tracking, with its detection accuracy directly impacting the predictive performance of subsequent filters. To evaluate the performance of the YOLOv8 detection model used in this paper, comprehensive comparative experiments were conducted with mainstream object detection models (such as YOLOv5, Faster R-CNN, and RetinaNet) across multiple typical scenarios (including daytime, nighttime, occlusion, low-light, etc.). Table 5 summarizes the performance of different detection models in these typical scenarios.

Table 5. Comparison of performance among different detection models.

Models	Daytime mAP@0.5	Nighttime mAP@0.5	Occlusion mAP@0.5	low-light	FPS	Loss
YOLOv8	94.3%	90.1%	87.8%	84.5%	45	0.43
YOLOv5	91.2%	85.4%	83.1%	79.8%	35	0.57
Faster R-CNN	88.5%	83.2%	79.5%	75.4%	15	0.62
RetinaNet	89.1%	82.8%	80.3%	77.0%	20	0.59

From the comparison of mean Average Precision (mAP) at IoU threshold of 0.5, it can be seen that YOLOv8 outperforms other models in all scenarios, particularly in nighttime and occlusion scenarios, achieving mAP values of 90.1% and 87.8%, respectively.

The final loss value of YOLOv8 is 0.43, which is approximately 25% lower than that of YOLOv5 and RetinaNet, and approximately 30% lower than that of Faster R-CNN. This verifies the rationality of selecting YOLOv8 as the detection module in this paper, providing robust support for subsequent tracking tasks.

5.1.2. Comparative Analysis of Kalman Filtering

Table 6 presents the performance of the SVM-based adaptive filter selection mechanism across different scenarios. The experimental results indicate that the

adaptive selection mechanism significantly outperforms fixed filter schemes in terms of tracking accuracy and robustness in complex scenarios such as accelerated motion and nonlinear motion.

Table 6. Performance of the SVM-based adaptive filter selection mechanism.

Scene Types	Filter Selection	Tracking Accuracy (MOTA)	Mean Squared Error (MSE)	Computational Latency(ms)
Uniform Motion	KF (Kalman Filter)	89%	1.5	12
Accelerated Motion	EKF (Extended Kalman Filter)	92%	1.2	14
Complex Nonlinear Motion	UKF (Unscented Kalman Filter)	94%	0.9	20
SVM-Based Adaptive Selection	Dynamic Switching	95%	0.8	15

5.2. Effectiveness of Error Feedback Mechanism

5.2.1. Comparison of Tracking Accuracy

In scenarios involving occlusion, sudden motion, and multi-object environments, the error feedback + SVM dynamic selection mechanism significantly improves tracking accuracy compared to other methods. Below is a comparison of the Mean Squared Error (MSE) for each method across different scenarios.

Table 7. Performance of the error feedback + SVM dynamic selection mechanism.

Scene Types	Methods	MSE	Trajectory Smoothness	Computational Latency (ms)
Occlusion Scenarios	Single KF	15.2	Low	10
	Single EKF	12.4	Medium	12
	Single UKF	11.5	Medium	20
	SVM Dynamic Selection Mechanism	9.7	High	18
	Error Feedback + SVM Dynamic Selection Mechanism	7.1	High	20
Sudden Motion Scenarios	Single KF	18.7	Low	10
	Single EKF	13.2	Medium	15
	Single UKF	11.0	Medium	20
	SVM Dynamic Selection Mechanism	9.2	High	18
	Error Feedback + SVM Dynamic Selection Mechanism	6.9	High	20

As shown in Table 7, the error feedback + SVM dynamic selection mechanism significantly reduces MSE in occlusion and sudden motion scenarios, with superior smoothness and real-time performance indicators compared to single filters and the standalone SVM dynamic selection mechanism.

5.2.2. Analysis of Interaction Strategies between Error Feedback and SVM

As shown in Table 8, a further evaluation is conducted to assess the difference in effectiveness between the standalone SVM dynamic selection mechanism and the SVM dynamic selection mechanism combined with error feedback. Two different error feedback mechanisms under distinct strategies are employed and compared across multiple scenarios.

1) Error Feedback Standalone Adjustment Strategy: Error feedback only adjusts the parameters of the currently selected filter without triggering a re-selection.

2) Error Feedback + Filter Re-selection Strategy: Error feedback not only adjusts the parameters but also triggers a re-selection of the filter.

Table 8. Comparative experiments of different strategies.

Scene Types	Strategies	MSE	Switching Frequency (times/second)
Occlusion Scenarios	Standalone Adjustment Strategy	8.8	2
	Filter Re-selection Strategy	7.1	3
Sudden Motion Scenarios	Standalone Adjustment Strategy	8.5	3
	Filter Re-selection Strategy	6.9	4

The performance advantages of SVM in dynamic filter selection primarily stem from the following aspects:

a) Discriminative power of motion features: Experiments have demonstrated that velocity, acceleration, and direction change angles exhibit significant distribution differences across different motion patterns, providing a clear basis for filter classification.

b) Nonlinear classification capability: Compared to traditional linear classifiers, SVM effectively handles nonlinear patterns through kernel function mapping, enabling more precise selection among KF, EKF, and UKF.

c) Matching of errors with scenario characteristics: By incorporating an error feedback mechanism, SVM dynamically adjusts selection strategies based on real-time error variations, further enhancing the system's adaptability.

5.3. Robustness of Adaptive Kalman Filtering

To validate the robustness of the SVM-based adaptive Kalman filtering, this paper analyzes the visualization results of tracking and recognition in various complex environments, as shown in Figure 4.

(a) The four frames in row (a) illustrate the tracking and recognition perfor-

mance of the model for a target vehicle with ID 17 in a turning scenario. It can be observed that the model demonstrates good performance.

(b) The four frames in row (b) show the tracking and recognition performance of the model for a target vehicle with ID 1 at different angles during nighttime. It is evident that the model maintains good locking on the target vehicle, even during the process of being overtaken.

(c) The four frames in row (c) present the tracking and recognition performance of the model for a target vehicle with ID 1 across multiple scenarios on a road segment. It is clear that the model can effectively identify and track the target vehicle even in multi-target and complex scenarios.

From this, it can be concluded that the SVM-based adaptive Kalman filtering exhibits strong robustness in complex environments. It can provide decision support for intelligent transportation, autonomous driving, security, and other fields.

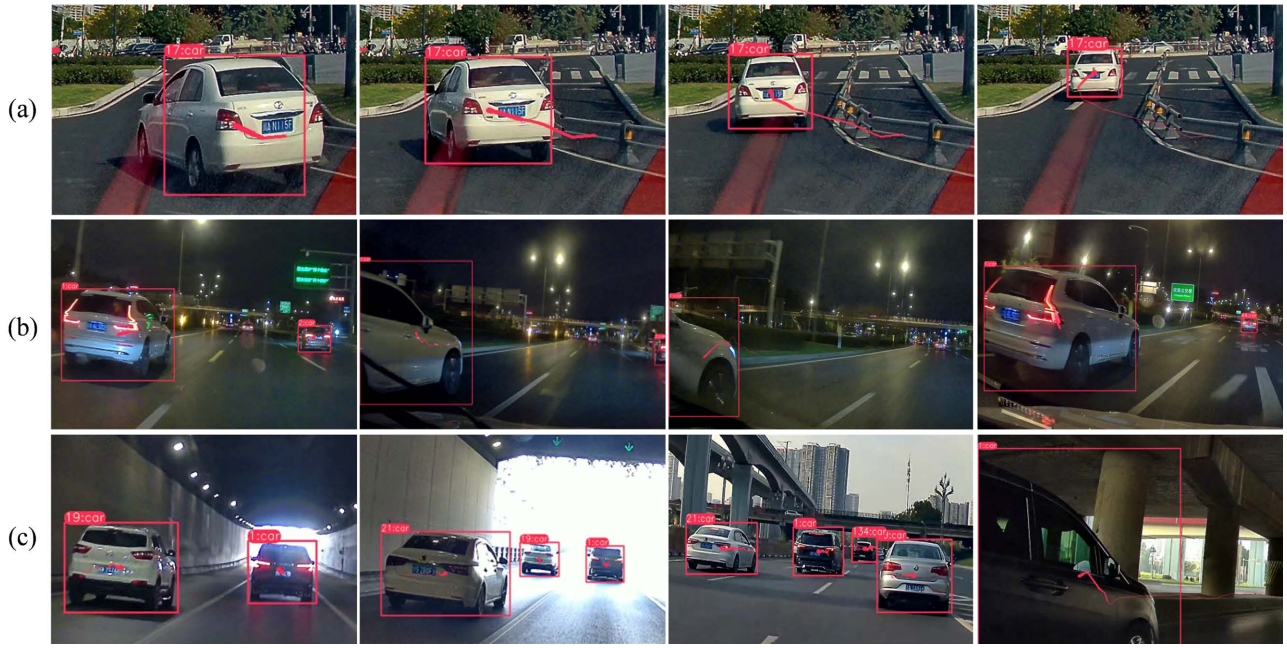

Figure 4. Visualization of inference results.

5.4. Comparison with Existing Methods

Through experimental comparisons with current mainstream vehicle tracking methods such as DeepSORT and FAIR MOT, the proposed research method in this study demonstrates advantages in detection accuracy, tracking accuracy, and system stability.

As shown in Table 9, compared to DeepSORT's 91.5% mAP on the KITTI dataset and FAIR MOT's 89.2% mAP, the YOLOv8 detection module combined with a dynamic filter selection mechanism proposed in this paper achieves a 94.3% mAP, exhibiting stronger adaptability to complex scenarios. Additionally, compared to DeepSORT's 35 FPS and FAIR MOT's 30 FPS, the real-time performance of the proposed method (45 FPS) significantly enhances processing speed while

meeting accuracy requirements, making it suitable for practical deployment.

Table 9. Comparison of experimental results table.

Datasets	Methods	mAP	MOTA	FPS
KITTI	DeepSORT	91.5%	87.5%	35
	FAIR MOT	89.2%	88.3%	30
	Ours	94.3%	90.6%	45
UA-DETRAC	DeepSORT	90.2%	85.4%	33
	FAIR MOT	88.1%	86.1%	28
	Ours	92.1%	89.7%	40

6. Conclusions and Future Work

This paper proposes an SVM-based adaptive filter selection mechanism and an error feedback mechanism, addressing the challenges of vehicle tracking in dynamic and complex scenarios with impressive results.

Although the integration of the components (YOLOv8, KF variants, SVM dynamic selection, and error feedback mechanism) represents incremental innovation, it yields the following significant system-level advantages:

1) Enhanced adaptability in complex dynamic scenarios, reducing accuracy degradation due to changing motion patterns through dynamic filter selection.

2) Improved coordination between detection and prediction, with the error feedback mechanism ensuring rapid system response to unexpected events.

3) Simplified complexity in multi-module development through deep integration, while optimizing both real-time performance and resource utilization.

Future research will focus on multimodal data fusion, incorporating sensor data from LiDAR and mmWave radars to enhance system applicability in adverse weather conditions, and large-scale application deployment, exploring distributed computing methods to meet real-time demands of large-scale traffic monitoring systems.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Tong, R.H., Li, J., Chen, H.S., *et al.* (2002) A Solution for Vehicle Tracking System Based on GPS and GIS Integration. *Computer Engineering and Science*, **1**, 29-32.

[2] Brown, M., Funke, J., Erlien, S. and Gerdes, J.C. (2017) Safe Driving Envelopes for Path Tracking in Autonomous Vehicles. *Control Engineering Practice*, **61**, 307-316. https://doi.org/10.1016/j.conengprac.2016.04.013

[3] Liu, H.M., Guan, H., Yu, M.H., *et al.* (2020) Research and Implementation of a Vehicle Tracking Algorithm Based on Multi-feature Fusion. *Journal of Chinese Computer*

Systems, **41**, 1258-1262.

[4] Wang, H.Y. (2022) Research on Vehicle Tracking Method Based on Intelligent Reflecting Surfaces in Complex Environments. Jilin University.

[5] Varghese, R. and M., S. (2024). Yolov8: A Novel Object Detection Algorithm with Enhanced Performance and Robustness. 2024 *International Conference on Advances in Data Engineering and Intelligent Computing Systems*, Chennai, 18-19 April 2024, 1-6. https://doi.org/10.1109/adics58448.2024.10533619

[6] Kim, H. (2019) Multiple Vehicle Tracking and Classification System with a Convolutional Neural Network. *Journal of Ambient Intelligence and Humanized Computing*, **13**, 1603-1614. https://doi.org/10.1007/s12652-019-01429-5

[7] Fu, C., Lu, K., Zheng, G., Ye, J., Cao, Z., Li, B., *et al.* (2023) Siamese Object Tracking for Unmanned Aerial Vehicle: A Review and Comprehensive Analysis. *Artificial Intelligence Review*, **56**, 1417-1477. https://doi.org/10.1007/s10462-023-10558-5

[8] Redmon, J., Divvala, S., Girshick, R. and Farhadi, A. (2016) You Only Look Once: Unified, Real-Time Object Detection. 2016 *IEEE Conference on Computer Vision and Pattern Recognition*, Las Vegas, 27-30 June 2016, 779-788. https://doi.org/10.1109/cvpr.2016.91

[9] Azimjonov, J. and Özmen, A. (2021) A Real-Time Vehicle Detection and a Novel Vehicle Tracking Systems for Estimating and Monitoring Traffic Flow on Highways. *Advanced Engineering Informatics*, **50**, Article 101393. https://doi.org/10.1016/j.aei.2021.101393

[10] Bakirci, M. (2024) Enhancing Vehicle Detection in Intelligent Transportation Systems via Autonomous UAV Platform and Yolov8 Integration. *Applied Soft Computing*, **164**, Article 112015. https://doi.org/10.1016/j.asoc.2024.112015

[11] Ren, S., He, K., Girshick, R. and Sun, J. (2017) Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **39**, 1137-1149. https://doi.org/10.1109/tpami.2016.2577031

[12] Zhang, Y., Liu, Z.L. and Wan, W. (2021) UAV-Based Vehicle Target Detection Using Faster R-CNN. *Electronic Science and Technology*, **34**, 11-20. https://doi.org/10.16180/j.cnki.issn1007-7820.2021.11.002

[13] Ouyang, B., Zhu, Y.J., Yang, L.K., *et al.* (2024) FA-SORT: A Lightweight Multi-Vehicle Tracking Algorithm. *Computer Engineering and Applications*, **60**, 122-134.

[14] Gordon, N.J., Salmond, D.J. and Smith, A.F.M. (1993) Novel Approach to Nonlinear/non-Gaussian Bayesian State Estimation. *IEE Proceedings F Radar and Signal Processing*, **140**, 107-113. https://doi.org/10.1049/ip-f-2.1993.0015

[15] Reid, D. (1979) An Algorithm for Tracking Multiple Targets. *IEEE Transactions on Automatic Control*, **24**, 843-854. https://doi.org/10.1109/tac.1979.1102177

[16] Wan, E.A. and Van Der Merwe, R. (2000) The Unscented Kalman Filter for Nonlinear Estimation. *Proceedings of the IEEE* 2000 *Adaptive Systems for Signal Processing, Communications, and Control Symposium*, Lake Louise, 4 October 2000, 153-158. https://doi.org/10.1109/asspcc.2000.882463

[17] Farag, W. (2022) Multiple Road-Objects Detection and Tracking for Autonomous Driving. *Journal of Engineering Research*, **10**, 237-262.

[18] Bui, T., Wang, G., Wei, G. and Zeng, Q. (2024) Vehicle Multi-Object Detection and Tracking Algorithm Based on Improved You Only Look Once 5s Version and Deepsort. A*pplied Sciences*, **14**, Article 2690. https://doi.org/10.3390/app14072690

[19] Gao, J., Han, G., Zhu, H. and Liao, L. (2024) Multiple Moving Vehicles Tracking Al-

gorithm with Attention Mechanism and Motion Model. *Electronics*, **13**, Article 242. https://doi.org/10.3390/electronics13010242

[20] Geiger, A., Lenz, P., Stiller, C. and Urtasun, R. (2013) Vision Meets Robotics: The KITTI Dataset. *The International Journal of Robotics Research*, **32**, 1231-1237. https://doi.org/10.1177/0278364913491297

[21] Wen, L., Du, D., Cai, Z., Lei, Z., Chang, M., Qi, H., *et al.* (2020) UA-DETRAC: A New Benchmark and Protocol for Multi-Object Detection and Tracking. *Computer Vision and Image Understanding*, **193**, Article 102907. https://doi.org/10.1016/j.cviu.2020.102907