Amrita Vishwa Vidyapeetham

Centre for Excellence in Computational Engineering and Networking

Amrita School of Artificial Intelligence, Coimbatore

Topic: Trajectory Predicition using Kalman filter

21MAT301

Prepared by:

Batch A Group – 6

Team members:

S.NO	NAME	ROLL.NO
1.	MARREDDY MOHIT SASANK REDDY	CB.EN.U4AIE21031
2.	YARRAM SRI SATHWIK REDDY	CB.EN.U4AIE21077
3.	P.SAI VIJAY KUMAR	CB.EN.U4AIE21040
4.	B. HARISH BALAJI	CB.EN.U4AIE21007

Dr. Abhijith

An End Semester Project submitted to the CEN department as a part of course evaluations of Mathamatics for Intelligent Systems for B.Tech in Artificial Intelligence

ACKNOWLEDGMENT

We extend our heartfelt thanks to the Centre for Excellence in Computational Engineering and Networking (CEN) at Amrita Vishwa Vidyapeetham, Coimbatore, for providing an excellent research environment. Our gratitude goes to Dr. Abhijith, Assistant Professor in the Department of CEN, for guiding our research. Under his supervision, we identified our research area, topic, and problem. Special acknowledgment to our team members for their support, and appreciation to our university for this opportunity. Lastly, we express special gratitude to the Centre for Excellence in CEN for facilitating research in this field.

Abstract

This project focuses on the application of the Kalman Filter for predicting the trajectories of vehicles in various real-world scenarios. The need for accurate trajectory predictions is essential for intelligent systems, particularly in the fields of autonomous vehicles, robotics, and surveillance.The project involves simulating and studying the movement patterns of vehicles, integrating the Kalman Filter with object detection techniques. This approach not only advances understanding of fundamental filter concepts but also provides a practical, hands-on learning experience. By predicting vehicle trajectories accurately, the project aims to contribute to informed decision-making in applications such as traffic management, autonomous navigation, and surveillance systems.

MOTIVATION OF PROJECT

This project is all about accurately predicting the movement of objects in real-world situations, like in robots or computer vision. We use a tool called the Kalman Filter to handle the uncertainties that come with predicting object paths. By simulating how objects move and combining it with the Kalman Filter, we not only learn the basics of this filter but also get a practical learning experience. This can be useful in areas like surveillance, robots, and self-driving systems, where knowing exactly how objects will move is crucial for making good decisions and improving system performance. The main goal of this project is to make it easier for people to understand and use the Kalman Filter in dealing with real-world challenges.

INTRODUCTION

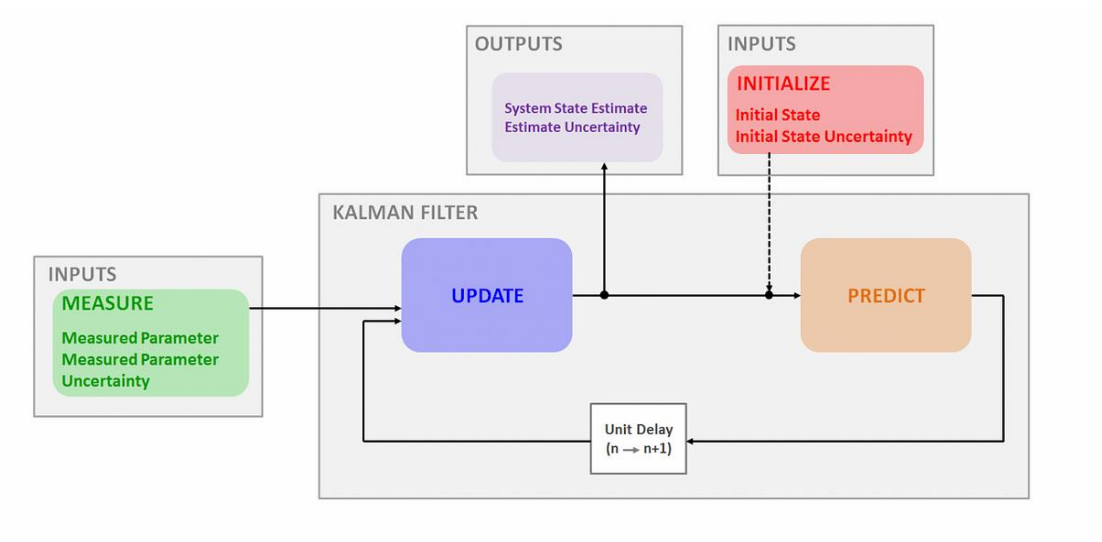
KALMAN FILTER:

Kalman Filtering is an algorithm/technique that does state estimation using measurements that are noisy.

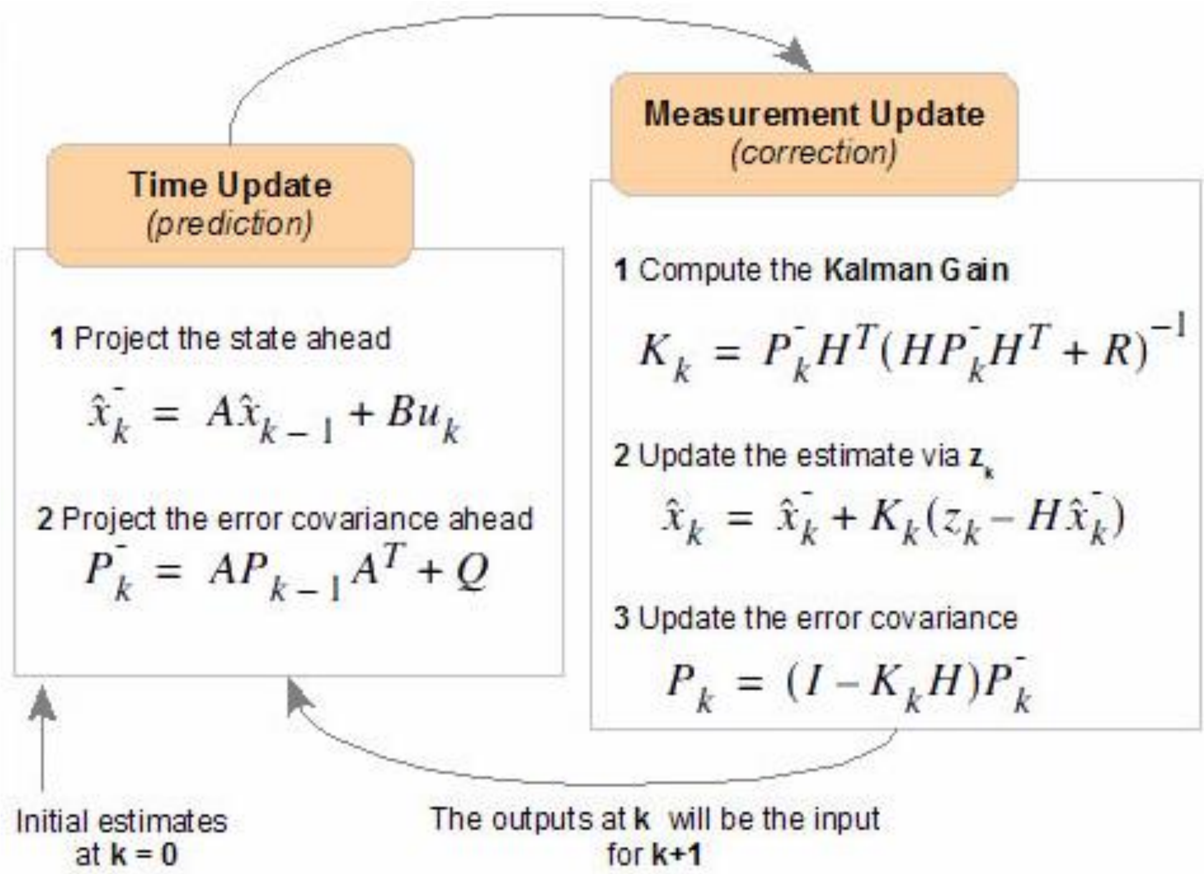
Kalman Filtering is renowned for its optimality in estimating the true state of a system. By dynamically adjusting its predictions based on both measurements and prior estimates, the Kalman filter minimizes the impact of noisy data, providing an optimal and continually refined estimation of the system's parameters.

It's a iterative mathematical process that use a set of equations and consecutive data inputs to quickly estimate the true value value, position, velocity etc of the object being measured, when the measured value contain unpredicted or random error, uncertainty or variation.

Flow chart:



KALMAN FILTER:



- $\overline{\hat{x}}\_k^-$: Priori state estimate
- $\hat{x}\_k$: Posteriori state estimate
- P_k^-: Priori estimate covariance
- P_k: Posteriori estimate covariance
- A: State transition matrix
- B: Control input matrix
- u_k: Control input vector
- K_k: Kalman gain
- H: Measurement matrix
- R: Measurement noise covariance
- Q: Process noise covariance

1)Kalman Filter Trajectory Prediction for Noisy Measurements in 2D Space:

Intialization :

First, it predicts where the ball might be based on its previous movement (like a guess).

Then, it compares that guess to the actual measurement (like checking your guess with a hint).

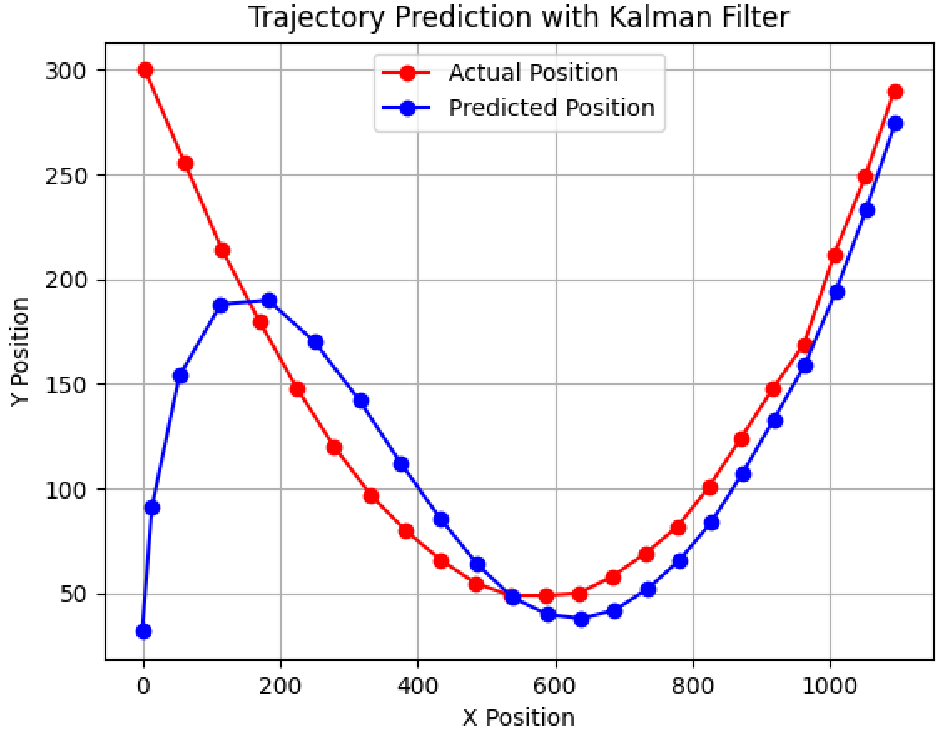
Prediction :

It uses this comparison to refine its guess for the next step, getting closer to the true secret code (ball's position).

Trajectory prediction :

This process repeats with each measurement, making the guesses (predictions) increasingly accurate.

OUTPUT :



2)Kalman Filter Cursor(Mouse) Tracking in OpenCV:

Initialization:

Kalman Filter is initialized with matrices A, H, Q, R, initial state, and initial covariance matrix (P).

Mouse Event Handling:

- OpenCV window "KalmanDemo" is set up to handle mouse events using `cv2.setMouseCallback`.
- Left-click initializes the cursor position and velocity, triggering the Kalman Filter initialization.
- Double-click clears the display, and mouse movement updates the cursor's position.

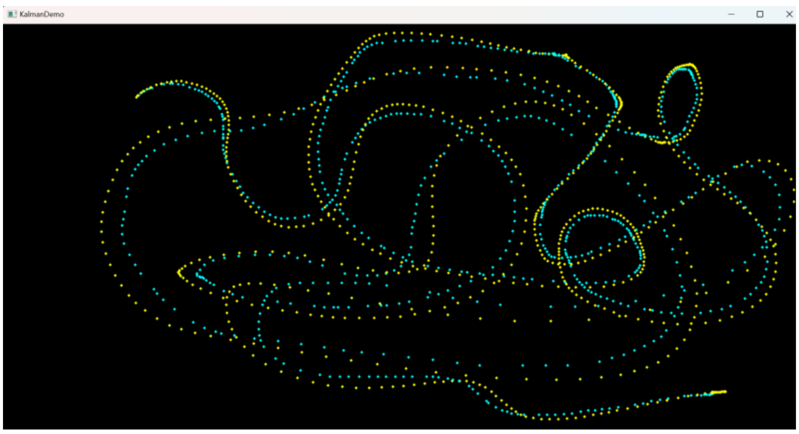
Visualization:

- - Display is continuously updated as the cursor moves.
- - **Yellow** circle marks the measured cursor position, while **cyan(blue)** circle marks the estimated position from the Kalman Filter.

Dynamic Tracking:

- - Kalman Filter dynamically adjusts predictions, providing smooth and accurate cursor tracking even in the presence of noise.

OUTPUT :



3)Kalman Filter and Optical Flow-based Car Detection and Tracking:

Explanation :

White Car Detection and Tracking:

- Convert the frame to HSV and create a mask to detect white cars. Extract white car regions and apply Haar Cascade for car detection in those regions.
- For each detected car: **Without Optical Flow**: Predict the position using Kalman Filter (predicted_position_no_optical_flow).
- Draw circles at the predicted positions **without optical flow**.
- Optical Flow-based Motion Estimation: Use Lucas-Kanade optical flow to estimate motion between consecutive frames.
- Update the Kalman Filter with the mean motion.
- Predict the position using Kalman Filter with optical flow (predicted_position_with_optical_flow).
- Draw circles at the predicted positions with optical flow.
- **Combine Predictions**: Calculate the resultant position as the average of both predictions. Draw a circle at the resultant position in red.
- **Visualize**: Draw circles around the detected car and predicted positions.

Lucas-Kanade Optical Flow Initialization:

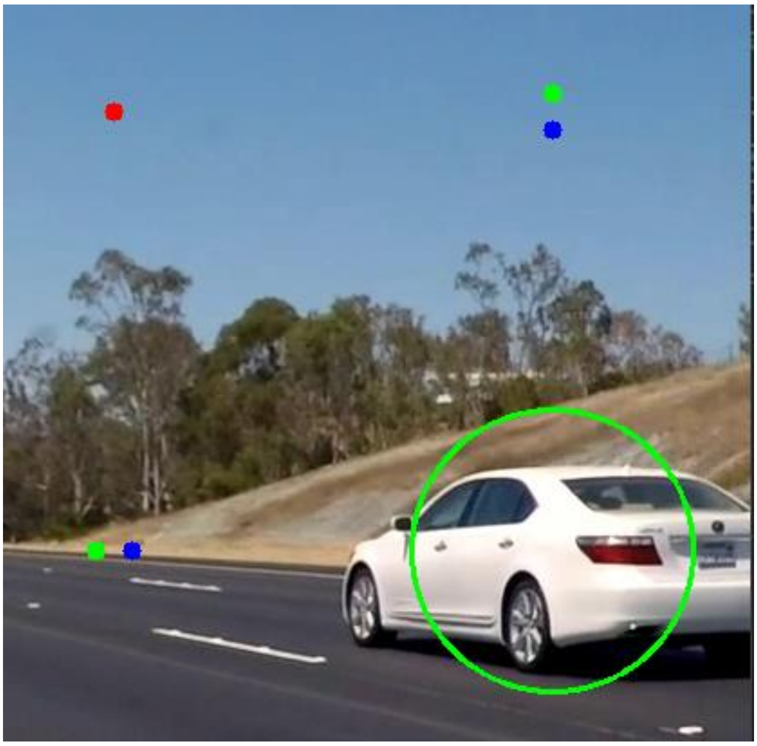
1. Initialize parameters for Lucas-Kanade optical flow, such as window size, maximum pyramid level, and termination criteria.
2. Lucas-Kanade optical flow is applied to estimate the motion of these objects between frames. Optical flow helps understand how objects are moving in the scene.

Video Capture and Processing:

1. Open a video capture from a file ('pv_2.mp4').
2. Read frames from the video, detect and track white cars using Kalman Filter and optical flow, and visualize the results.

OutPut :

Result 1:



Result 2:

Conclusion :

In conclusion, the Kalman filter emerges as a linchpin in the realm of object tracking, showcasing its resilience in handling noisy measurements and uncertainties. This project has illustrated the filter's effectiveness in predicting and refining the trajectories of white cars by seamlessly integrating it with Haar cascade detection and Lucas-Kanade optical flow. The filter's iterative nature, informed by both predictions and measurements, underscores its adaptability in real-world scenarios.

Beyond its immediate application in vehicle tracking, the Kalman filter's robustness positions it as a fundamental tool with broader implications in robotics, autonomous systems, and surveillance. Its unique ability to provide optimal state estimation, even in the presence of dynamic and unpredictable variables, solidifies its role in enhancing decision-making processes. As technology advances, the Kalman filter remains a cornerstone in the landscape of state estimation, contributing significantly to the precision and reliability of object tracking systems.

Future work :

1. Improved Object Recognition
2. Dynamic Adaptive Parameters
3. Multi-Object Tracking
4. Integration with Semantic Segmentation
5. Real-time Processing Optimization
6. Adaptive Optical Flow Strategies

THANK YOU