Instituto Tecnológico
y de Estudios Superiores de Occidente
Reconocimiento de validez oficial de estudios de nivel superior según acuerdo secretarial 15018, publicado en el Diario Oficial de la Federación del 29 de noviembre de 1976.

Departamento de Electrónica, Sistemas e Informática
Especialidad en Sistemas Embebidos

Embedded Implementation of a Kalman Filter for the Fusion of Automotive Inertial Sensors using CARLA Simulator

TESIS que para obtener el **GRADO** de
ESPECIALISTA EN SISTEMAS EMBEBIDOS

Presenta: **RICARDO DE JESÚS CASTILLO TORRES**

Asesor **LUIS ENRIQUE GONZÁLEZ JIMÉNEZ**

Tlaquepaque, Jalisco. julio de 2021.

Embedded Implementation of a Kalman Filter for the Fusion of Automotive Inertial Sensors using CARLA Simulator

Ricardo de Jesús Castillo Torres
Embedded Systems Graduate Program
ITESO
Guadalajara, Mexico
dejesus.castillo@iteso.mx

Luis Enrique González-Jiménez
Embedded Systems Graduate Program
ITESO
Guadalajara, Mexico
luisgonzalez@iteso.mx

***Abstract*—One of the newest research and development trends in the technology sector involves self-driving vehicles. However, failures and security problems are a main concern in autonomous driving. The aim of this paper is to build a real-time Kalman Filter that collects and fuses sensor data from vehicles to provide more accurate information of the car's position and orientation. This research work uses the Carla simulator as the platform to simulate a real environment. For the sensor fusion, two groups were selected: 1) GPS and acceleration to obtain a better estimation of the position; and 2) magnetometer and gyroscope for a better estimation regarding the car's orientation. For the data processing phase, an ARM Cortex-M4 microcontroller was used. The Kalman filter produced a noise-free estimation of the position and orientation of the vehicle. This implementation is useful for detecting a car's estimated position in a tunnel when GPS signals are weak.**

Keywords—Autonomous, Carla Simulator, Kalman Filter, Gaussian Noise, GPS, Accelerometer, Magnetometer, Gyroscope.

I. Introduction

The research and development of autonomous driving is a growing trend in the automotive industry. According to Mordor Intelligence, the autonomous driving industry was valued at USD 19.46 billion in 2020 [1]. Nonetheless, autonomous driving can create new problems, for instance, the most relevant and impactful event occurred in 2018 when an autonomous car of the Uber company ran over and killed a pedestrian [2]. As a result, safety in self-driving vehicles became a top priority for researchers.

A new tool that has been used to in this field is the Kalman Filter. This algorithm uses measurements of known variables to produce an estimate of unknown variables, which tend to be more accurate than a single measurement variable. The goal of the algorithm is to produce an accurate reading of the position and velocity of the car in situations where it is difficult to obtain a precise measurement, such as passing through a tunnel. In 2019, Dhongade and Khandekar implemented a GPS and an Inertial Measurement Unit (IMU) integration using a Kalman Filter and the LabView Tool [3] to increase reliability in a self-driving vehicle. Although this implementation was successful, it was not implemented in an embedded system, thus making its application difficult in a real-time environment.

This paper aims to develop an embedded implementation of a Kalman Filter using an ARM Cortex-M4 microcontroller and Carla, a real-time environment simulator. The selected sensors for the fusion process were a GPS and an accelerometer to have a more accurate estimate of the car's position; and a magnetometer and a gyroscope to have a more accurate estimate of the direction of the car.

II. Methodology

The implemented system was a closed-loop connection between Carla Simulator and the microcontroller. Carla is a free, open-source simulator for autonomous driving research. The software has multiple built-in examples. The manual_control example code enables a virtual environment with a single car that can be manually driven with selected keys of the keyboard. The vehicle is equipped with multiple sensors, including GPS and IMU, which are the target sensors. The source code was modified to add Gaussian Noise to the virtual sensors to simulate a real-life noisy sensor. The modified sensor information was sent through the serial port of the computer connected to the microcontroller.

Once the necessary information was received in the microcontroller, the actual Kalman Filter was implemented. The Kalman Filter algorithm uses time steps to compute measurements at a given point in time called states. A state of the system is represented by two variables:

$\hat{x}_k$ - *a posteriori state estimate at timestep k*

P_k - *a posteriori estimate covariance matrix*

For the Kalman Filter to calculate these states, a modeled system needs to be given. This model includes the following matrices:

A – state-transition model

B – *control-input model*

C - *observation model*

R – covariance of the observation noise

Q - covariance of the process noise

The process to compute these states is divided into two phases. Phase 1 is called prediction. The prediction phase uses the previous time step estimate to produce a new estimate of the state at the current time step. The computed estimate for phase 1 is known as *a priori state estimate.* To compute the *a priori* *state estimate* and *a priori estimate covariance matrix*, the following formulas are followed, respectively:

$$\hat{x}_k^- = A\hat{x}_{k-1} + Bu_k \tag{1}$$

$$P_k^- = AP_{k-1}A^T + Q \tag{2}$$

where the k-1 subindex represents the previous time step estimates and the u_k represents the input value of the desired sensor at the current time.

After phase 1 estimate is calculated, the process continues to phase 2, known as update. In the update phase, the predicted state is compared with the current observation from the sensors, and the difference is multiplied by an optimal Kalman gain to refine the prediction phase. The computed estimate for phase 2 is known as *a posteriori state estimate*. To calculate the optimal Kalman gain and both *a posteriori state estimate* variables, the following formulas must be computed:

$$K_k = \frac{P_k^- C^T}{CP_k^- C^T + R} \tag{3}$$

$$\hat{x}_k = \hat{x}_k^- + K_k(y_k - C\hat{x}_k^-) \tag{4}$$

$$P_k = (I - K_k C)\, P_k^- \tag{5}$$

where y_k represents the input value of the desired second sensor that is being fused and I represents an identity matrix of the same size as the number of input variables. [4]

A general state-space analysis was performed for the sensors that were used in the project: GPS, accelerometers, gyroscope, and magnetometer. Based on the fact that the acceleration is the double derivative of the position, the state space model for the GPS and accelerometer sensors was represented by the following matrixes:

$$A = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix}$$

$$B = \begin{bmatrix} 0 \\ T \end{bmatrix}$$

$$C = \begin{bmatrix} 0 & 1 \end{bmatrix}$$

$$R = 0.1$$

$$Q = \begin{bmatrix} \frac{R}{100} & 0 \\ 0 & \frac{R}{100} \end{bmatrix}$$

T represents the sampling time. The state estimate variables are then transferred through the serial port connection and back to the PC for comparison.

III. Results

Virtual GPS and IMU sensors were extracted from Carla through the serial port and into the microcontroller to implement the filtering process. In the following figure, the Carla simulator user view is shown. Sensor data is visible to the user in the left-hand side of the screen.

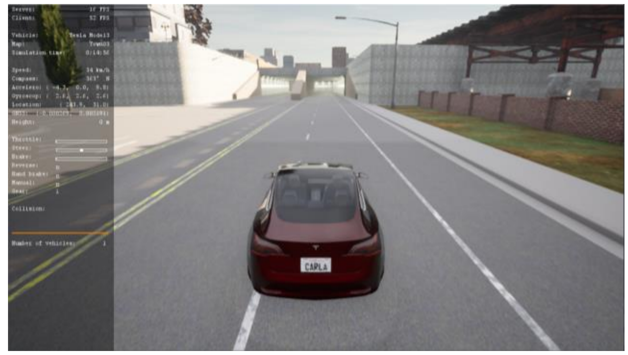

Figure 1. Carla simulator user view

The first experiment used the x and y coordinates of the GPS sensor along with the x and y components of the accelerometer sensor to produce a better estimation of the car's position in real time. Both sensors had Gaussian noise added to the output value beforehand being sent out through the serial port. Several R (covariance) values were tested. Fig. 2 - Fig. 4 show the difference between the noisy coordinates (red line) compared with the noise-free output of the Kalman filter algorithm (blue line) with their respective R value being tested.

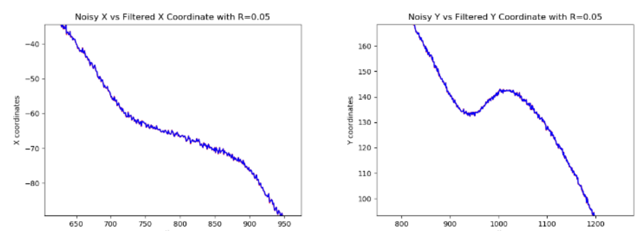

Figure 2. Noisy vs Filtered X coordinate (left). Noisy vs Filtered Y coordinate (right). Using R as 0.05.

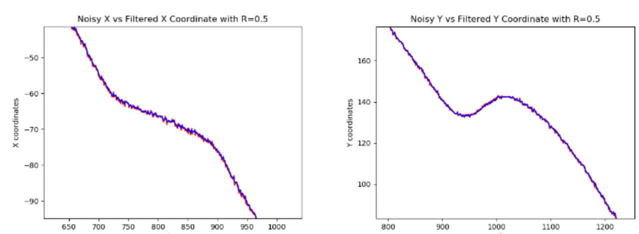

Figure 3. Noisy vs Filtered X coordinate (left). Noisy vs Filtered Y coordinate (right). Using R as 0.5.

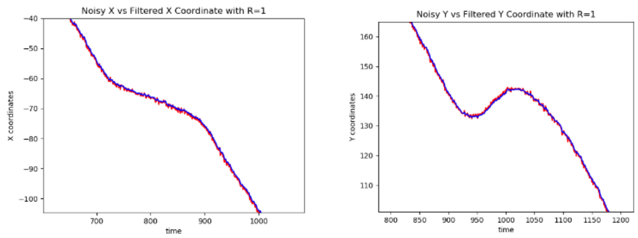

Figure 4. Noisy vs Filtered X coordinate (left). Noisy vs Filtered Y coordinate (right). Using R as 1.

As it can be seen in the figures above when the covariance value is set as 0.05 the difference between the noisy and filtered values is almost imperceptible. The difference is clearer when R value is increased to 0.5. Finally, the best result is achieved when R is increased to 1. Figure 4 shows a clear difference between a noise red graph compared to a smooth filtered blue graph. Therefore, for the final experiment the R value was set at 1.0 for the GPS and accelerometer sensor fusion. The result is shown in Figure 5.

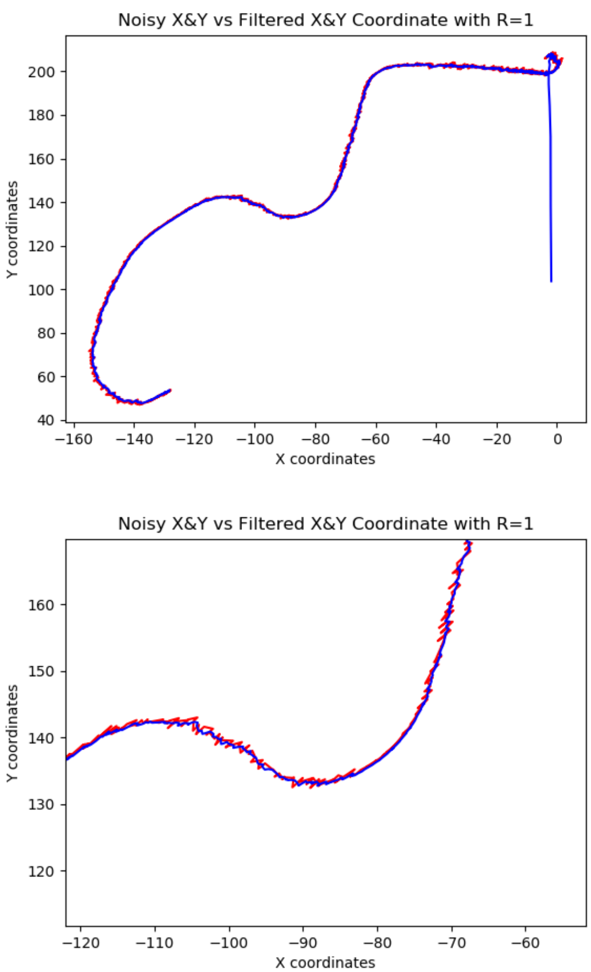

Figure 5. Noisy X and Y coordinates (red) vs Filtered X and Y coordinates (blue) using R as 1.

The blue line shows a smooth, noise-free plotting that shows the real values that the sensors were outputting from the self-driving car.

Likewise, the virtual gyroscope and magnetometer sensors were extracted from the simulator and transferred to the microcontroller through the serial port. Both sensors also had Gaussian noise added beforehand. The same filtering process was repeated using the Kalman filter algorithm. The comparison between noisy and filtered values can be seen in Figure 6.

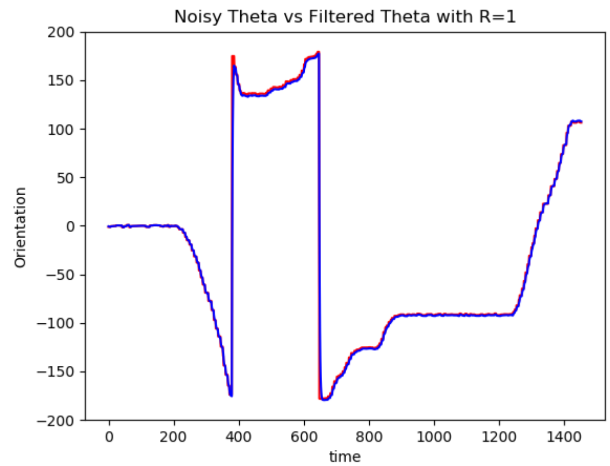

Figure 6. Noisy vs filtered theta using R as 0.01.

IV. CONCLUSION

The Kalman filter proves to be a light, efficient method to improve real time data acquisition. Combining more than one sensor and cleaning the output data through a Kalman filter, a more accurate and smoother estimation of the position and orientation of an autonomous car is acquired even under challenging situations like passing through a tunnel or a mountainous landscape. Hence, the Kalman filter improves the safety of both self-driving vehicles users and pedestrians.

V. ACKNOWLEDGEMENTS

This work was supported by the Mexican federal government through the CONACyT agency.

REFERENCES

[1] “Autonomous/Driverless Car Market Size & Forecast | Industry Growth 2021 to 2026 - Mordor Intelligence.” https://www.mordorintelligence.com/industry-reports/autonomous-driverless-cars-market-potential-estimation (accessed Jun. 17, 2021).

[2] K. Conger, “Driver Charged in Uber’s Fatal 2018 Autonomous Car Crash,” The New York Times, Sep. 15, 2020. Accessed: Nov. 14, 2020. [Online]. Available: https://www.nytimes.com/2020/09/15/technology/uber-autonomous-crash-driver-charged.html

[3] A. P. Dhongade and M. A. Khandekar, “GPS and IMU Integration on an autonomous vehicle using Kalman filter (LabView Tool),” in 2019 International Conference on Intelligent Computing and Control Systems (ICCS), May 2019, pp. 1122–1125. doi: 10.1109/ICCS45141.2019.9065851.

[4] MATLAB, Understanding Kalman Filters, Part 6: How to Use a Kalman Filter in Simulink. Accessed: Jul. 12, 2021. [Online Video]. Available: https://www.youtube.com/watch?v=ouRM4sgoVs8