



Geoinformatics Project, 2023

Kalman Filter Application

Name	Student ID	E-mail
Ali Badr Eldin Ali Mohamed	10760530	Ali.mohamed@mail.polimi.it

Project Supervisor

- Professor Ludovico Biagi (ludovico.biagi@polimi.it)

Course Instructor

-Professor Mariagrazia Fugin

Abstract

This project presents a Python-based implementation of a Kalman filter for smoothing trajectory of a moving body, which includes a graphical user interface (GUI). The Kalman filter is a widely used algorithm for estimating the state of a system based on noise measurements.

The GUI provides an intuitive interface for users to interact with the filter, allowing them to adjust filter parameters. The GUI also provides tools for saving the resulting filtered data.

The data utilized in this report is based on the emulation of 5G signals, following the standards set by the fifth-generation wireless technology. The data was generated within the framework of the GINTO5G project, which received funding from the European Space Agency (ESA).

Overall, this project demonstrates the feasibility and effectiveness of using a Kalman filter for smoothing trajectory and shows how a GUI can improve user experience and simplify the configuration process.

Contents

Abstract.....	i
List of Figures.....	iii
1.0 Introduction	1
1.1 Objectives	1
2.0 Methodology.....	2
2.1 system design.....	2
2.2 Kalman Filter Algorithm	4
3.0 Prerequisites.....	6
3.1 Libraries	6
4.0 Application interface.....	6
5.0 Outputs.....	9
6.0 Conclusion	12
7.0 References.....	13

List of Figures

Figure 1	User interface	7
Figure 2	CVS File Selection	8
Figure 3	Output Folder Selection	9
Figure 4	Position Trajectory	10
Figure 5	Standard deviation on X over time	10
Figure 6	Standard deviation on Y over time	11
Figure 7	filtered and observed position.	12

1.0 Introduction

In recent years, the development of 5G networks has opened new possibilities for high-precision positioning applications. However, the accuracy and precision of 5G positioning data can be affected by a variety of factors, including multipath interference, signal attenuation, and environmental conditions. To address these challenges, the use of Kalman filters has emerged as a popular method for processing 5G positioning data. Kalman filters are a class of mathematical algorithms that can be used to estimate the state of a system based on noisy or uncertain measurements. In this report, we explore the application of Kalman filters to emulated data, according to 5G signal standard, with a focus on smoothing the trajectory of the moving body. We present the methodology used to implement the Kalman filter algorithm and analyze the results of applying it to the data.

1.1 Objectives

The objectives of the exercise have been to develop a tool that.

- Using the previous estimation to estimate the state of a moving body at a new epoch, incorporating new measurements to update and refine the estimation.
- Produce a trajectory plot showing the filtered position of the body.
- Save the filtered coordinate as CSV file.
- incorporates a Graphical User Interface for accessing the tool.

2.0 Methodology

2.1 system design

2.1.1 Preprocessing

The data used in this study consists of emulated data, according to 5G signal standard, stored in a CSV file. Each row in the CSV file represents a single measurement of the device's position along with the corresponding time stamp.

The data in the CSV file has been preprocessed to remove any outliers or anomalies. This involved filtering the data to remove noise and calculate the Velocity and acceleration using the coordinates (X, Y) as follows:

$$v_{x+} = \frac{X(t+1) - X(t)}{\Delta t}$$

$$v_{x-} = \frac{X(t) - X(t-1)}{\Delta t}$$

$$v_x = \frac{1}{2} [v_{x+} + v_{x-}]$$

$$v_{y+} = \frac{Y(t+1) - Y(t)}{\Delta t}$$

$$v_{y-} = \frac{Y(t) - Y(t-1)}{\Delta t}$$

$$v_y = \frac{1}{2} [v_{y+} + v_{y-}]$$

$$V = \sqrt{V_x^2 + V_y^2}$$

Where:

v_x and v_y are velocity on X and Y respectively, on the epoch t.

Accelerations were computed as follow:

$$a_{x+} = \frac{V(t+1)-V(t)}{\Delta t}$$

$$a_{x-} = \frac{V(t)-V(t-1)}{\Delta t}$$

$$a_x = \frac{1}{2} [a_{x+} + a_{x-}]$$

$$a_{y+} = \frac{Y(t+1)-Y(t)}{\Delta t}$$

$$a_{y-} = \frac{Y(t)-Y(t-1)}{\Delta t}$$

$$a = \frac{1}{2} [a_{y+} + a_{y-}]$$

$$a = \sqrt{Vx^2 + Vy^2}$$

Where:

a_x and a_y are acceleration on X and Y respectively, on the epoch t.

2.2 Kalman Filter Algorithm

The Kalman filter has two steps: the prediction step, where the next state of the system is predicted given the previous measurements, and the update step, where the current state of the system is estimated given the measurement at that epoch. The steps translate to equations as follows:

2.2.1 Prediction:

$$\mathbf{X}_{t+1} = \mathbf{T}_{t+1}\mathbf{X}_t + \boldsymbol{\varepsilon}_{t+1}$$

$$\mathbf{K}_{t+1} = \mathbf{C}_{t+1}^e + \mathbf{T}_t \mathbf{C}_t^e \mathbf{T}_{t+1}^T$$

2.2.2 Update:

$$\mathbf{G} = \mathbf{K}_{t+1} \mathbf{A}_{t+1}^T (\mathbf{C}_{t+1}^v + \mathbf{A}_{t+1} \mathbf{K}_{t+1} \mathbf{A}_{t+1}^T)^{-1}$$

$$\mathbf{X}_{t+1} = \mathbf{G}_{t+1} \mathbf{Y}_{t+1} + [\mathbf{I} - \mathbf{G}_{t+1} \mathbf{A}_{t+1}] \mathbf{T}_{t+1} \mathbf{X}_t$$

$$\mathbf{C}_{t+1}^e = (\mathbf{I} - \mathbf{G}_{t+1} \mathbf{A}_{t+1}) \mathbf{K}_t$$

Where:

- X_t and K_t are the predicted state vector and covariance of the state, respectively, on the time step t before seeing the measurement.
- X_{t+1} and C_{t+1}^e are the estimated state vector and covariance of the state, respectively, on time step t after seeing the measurement.
- G is the Kalman gain, which tells how much the predictions should be corrected on time step t .
- T is transition matrix is used to convert the input of state to the new state matrix.
- A is Design matrix or transformation matrix.
- v_t is measurement noise.

State Vector represents the current state of a system and typically includes relevant variables or parameters. In the context of tracking the position and velocity of a moving body in a two-dimensional space (X, Y), the state vector can be defined as $[X, Y, V_x, V_y]$, where X and Y denote the position coordinates, and V_x and V_y represent the velocity components along the X and Y axes, respectively. This state vector encapsulates the essential information needed to estimate and track the position and velocity of the moving body in the specified coordinate system.

Transition Matrix defines how the state of the system evolves over time. This matrix is used in the prediction step of the Kalman filter. Given the previous state estimate, the transition matrix is applied to project or predict the expected state at the current time step.

$$T = \begin{bmatrix} I & \Delta t \\ 0 & I \end{bmatrix}$$

Where:

- I represent identity matrix.
- Δt represent time difference between epochs.

Design matrix defines the linear relationship between the state variables and the measurements. It maps the state vector, x , to the measurement space. The specific form of the design matrix depends on the system being modeled and the measurements being used. It is typically determined by the physical properties of the system and the sensors being employed.

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

3.0 Prerequisites

3.1 Libraries

Python is the programming language used to develop the application. To run the program, the user is required to install the following packages:

- Matplotlib
- tkinter
- pandas
- Numpy

3.2 Running the application.

To run the application, on the Command Line Interface or Anaconda prompt, navigate to the Kalman filter folder and run using the command below.

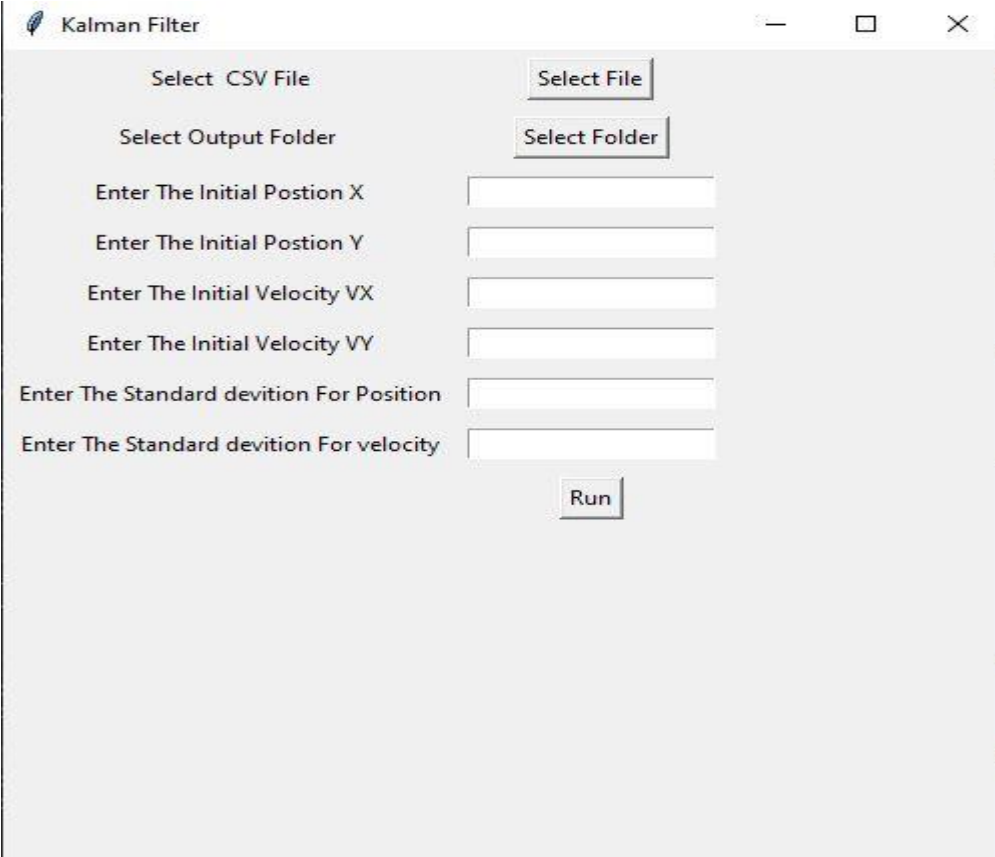
```
python Main.py
```

This will open the GUI OF Kalman Filter tool.

4.0 Application interface

The Kalman Filter application has a graphical user interface (GUI) built using the tkinter library in Python. The GUI has several buttons and entry fields to allow the

user to input data and run the Kalman Filter algorithm. The main interface consists of four input fields for the initial position and velocity of the object being tracked, and two standard deviation fields for the noise of the measurements. There are also three buttons, "Run", "Select Output Folder " and "Select CSV file". The "Run" button runs the algorithm and displays the plot of the object's position and velocity over time. The " Select Output Folder " button allows the user to save the plot and CSV file generated by the "Run" button to a specified directory. Additionally, there is a " Select CSV file " button that opens a file dialog for the user to select the input data file in CSV format.



The image shows a window titled "Kalman Filter" with a standard Windows-style title bar (minimize, maximize, close buttons). The window contains the following elements:

- Select CSV File**: A label with a **Select File** button to its right.
- Select Output Folder**: A label with a **Select Folder** button to its right.
- Enter The Initial Postion X**: A text label followed by an empty input field.
- Enter The Initial Postion Y**: A text label followed by an empty input field.
- Enter The Initial Velocity VX**: A text label followed by an empty input field.
- Enter The Initial Velocity VY**: A text label followed by an empty input field.
- Enter The Standard devition For Position**: A text label followed by an empty input field.
- Enter The Standard devition For velocity**: A text label followed by an empty input field.
- Run**: A button located below the standard deviation input fields.

Figure 1 : User interface

4.1 import CSV file.

To import the CSV file, the user should "Select file" and navigate to the location where the CSV file is stored on your computer.

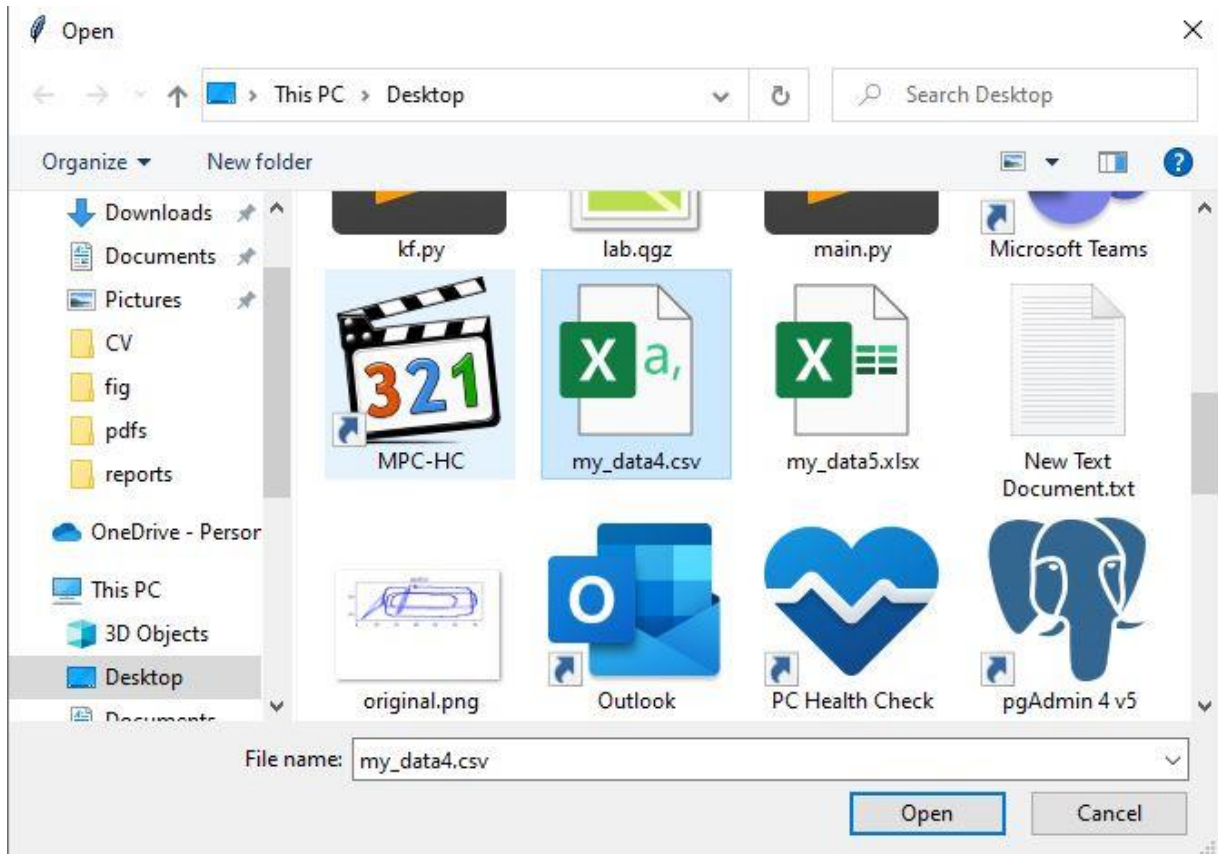


Figure 2 : CVS File Selection

4.2 Plots Folder

To select the folder for plots, click on the "Select Output Folder" button located on the main interface, A file dialog box will appear, allowing you to browse and select the folder where you want to save the plots. Once you have selected the desired folder, click on the "OK" button to confirm your selection. From now on, whenever you run the Kalman filter application and generate plots, they will automatically be saved in the folder you have previously selected.

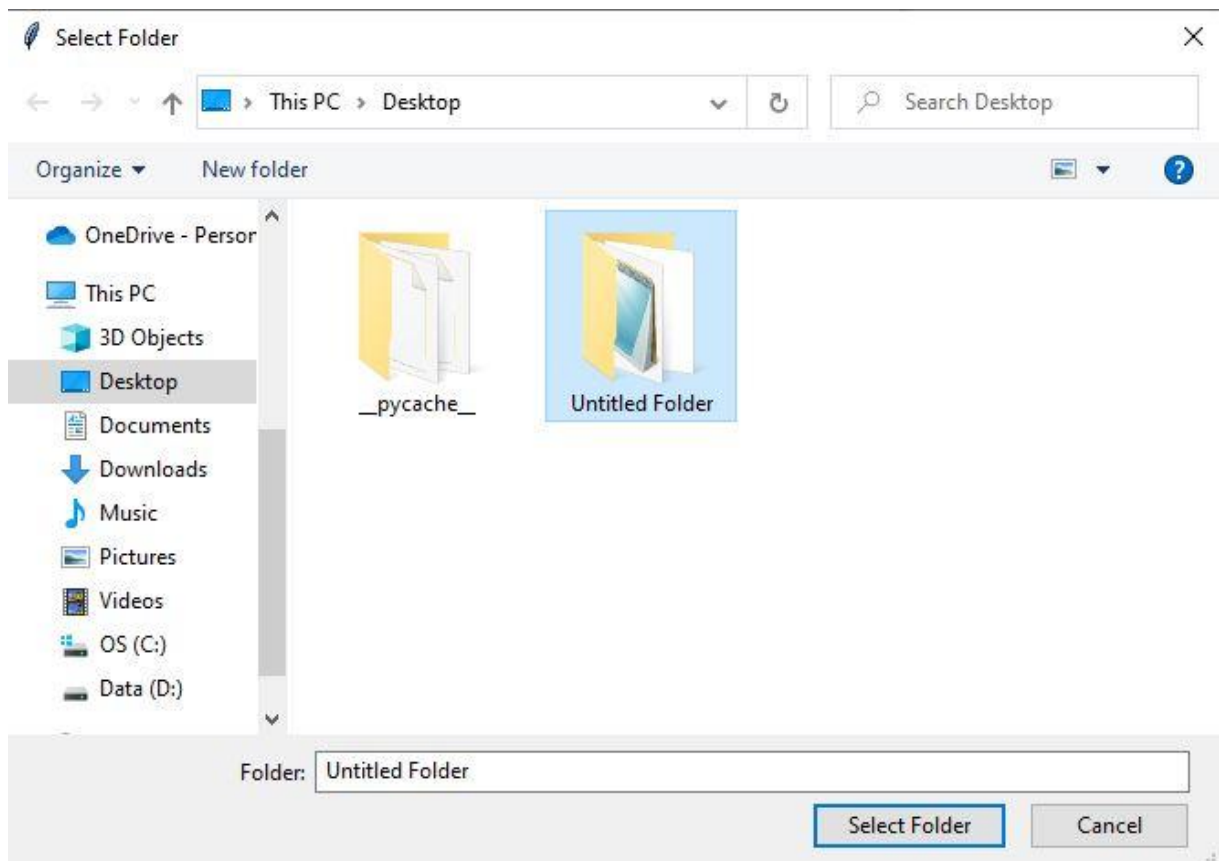


Figure 3 : Output Folder Selection

5.0 Outputs

4.3 Graphical outputs

The Kalman filter application generates two plots to help visualize the filtering process. The first plot shows the raw data and the filtered data superimposed on each other. The second plot shows the error covariance, which is a measure of the uncertainty of the filter. These two plots are useful for analyzing the performance of the Kalman filter and for tuning the filter parameters to improve its accuracy.

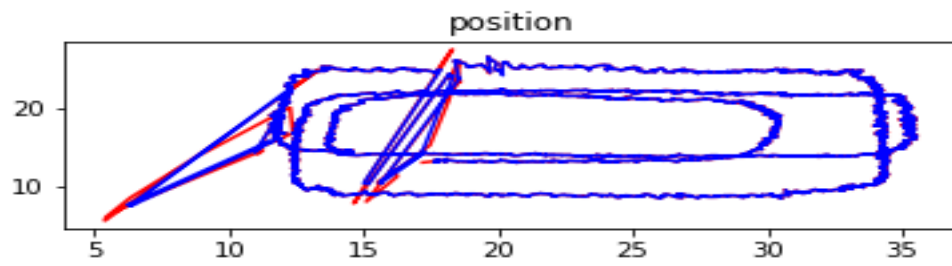


Figure 4 : Position Trajectory

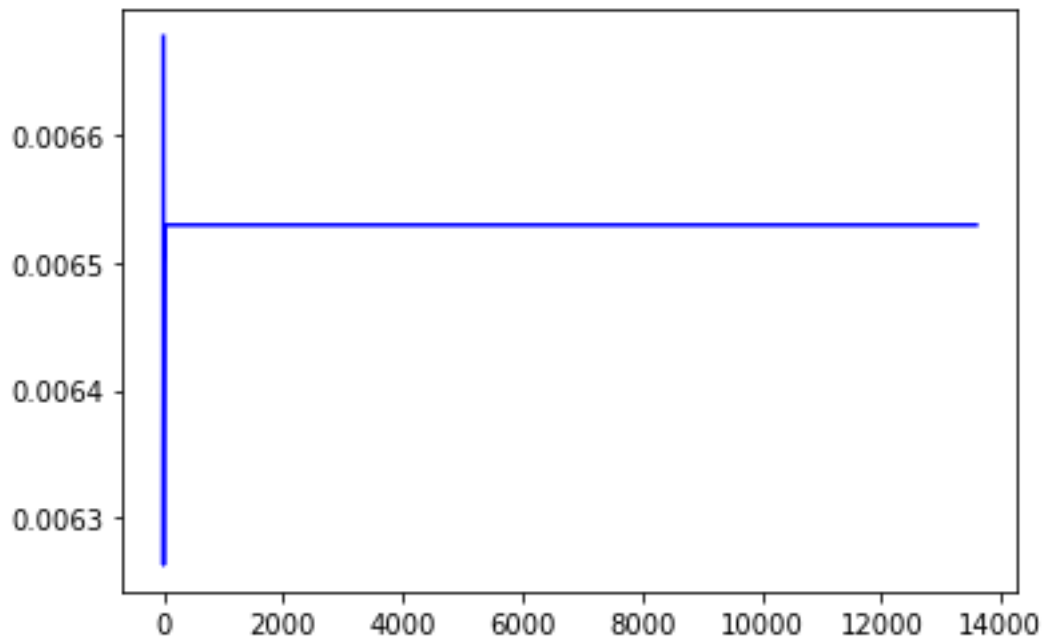


Figure 5: Standard deviation on X over time

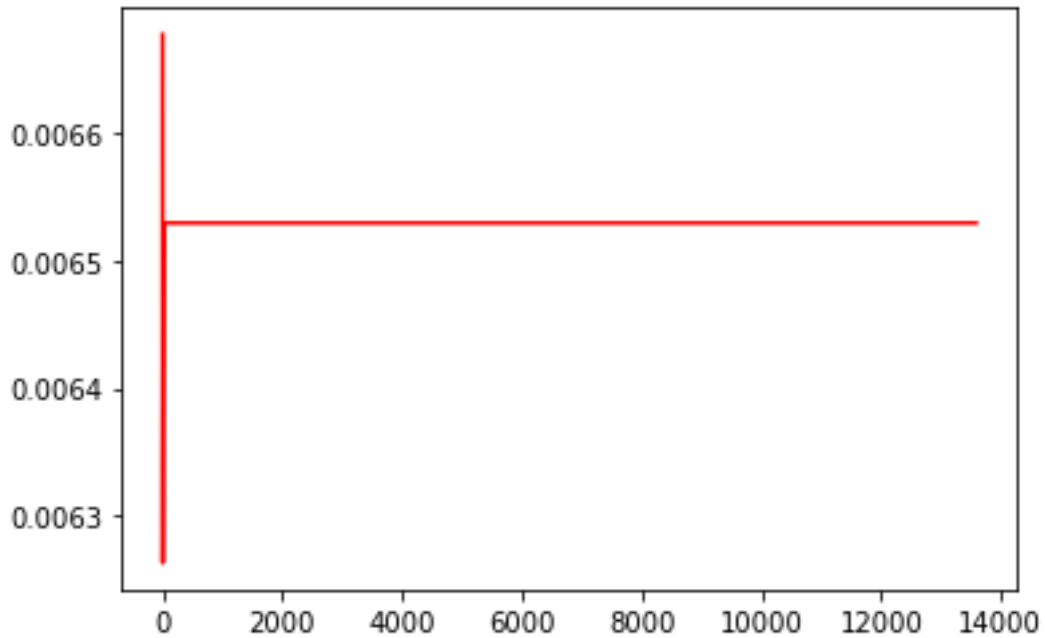


Figure 6: Standard deviation on Y over time

4.4 CSV Outputs

The CSV file generated by this application contains the position data of an object over time. This file contains the estimates of the object's position with respect to time. The resulting estimates are saved as two columns (x and y coordinates) in the CSV file. In addition to the position estimates, the CSV file also contains information about the estimated error covariance for each epoch. This can be used to assess the accuracy of the position estimates at each epoch. The error covariance is saved as two columns representing the standard deviation on the x and y.

Overall, the CSV file generated by this application provides a comprehensive record of an object's position and estimated error covariance over time, which can be used for further analysis and visualization.

6.0 Conclusion

In conclusion, the application of the Kalman filter in Python not only provides reliable and efficient estimation of a moving body's position amidst measurement noise and uncertainties but also showcases its ability to smooth the trajectory, as shown in the figure below. The resulting trajectory plot vividly demonstrates how the Kalman filter refines the estimated trajectory, reducing the impact of noise and uncertainties. The resulting CSV file, plots, and user-friendly graphical interface make this tool invaluable for a wide range of applications such as robotics, autonomous vehicles, and navigation systems.

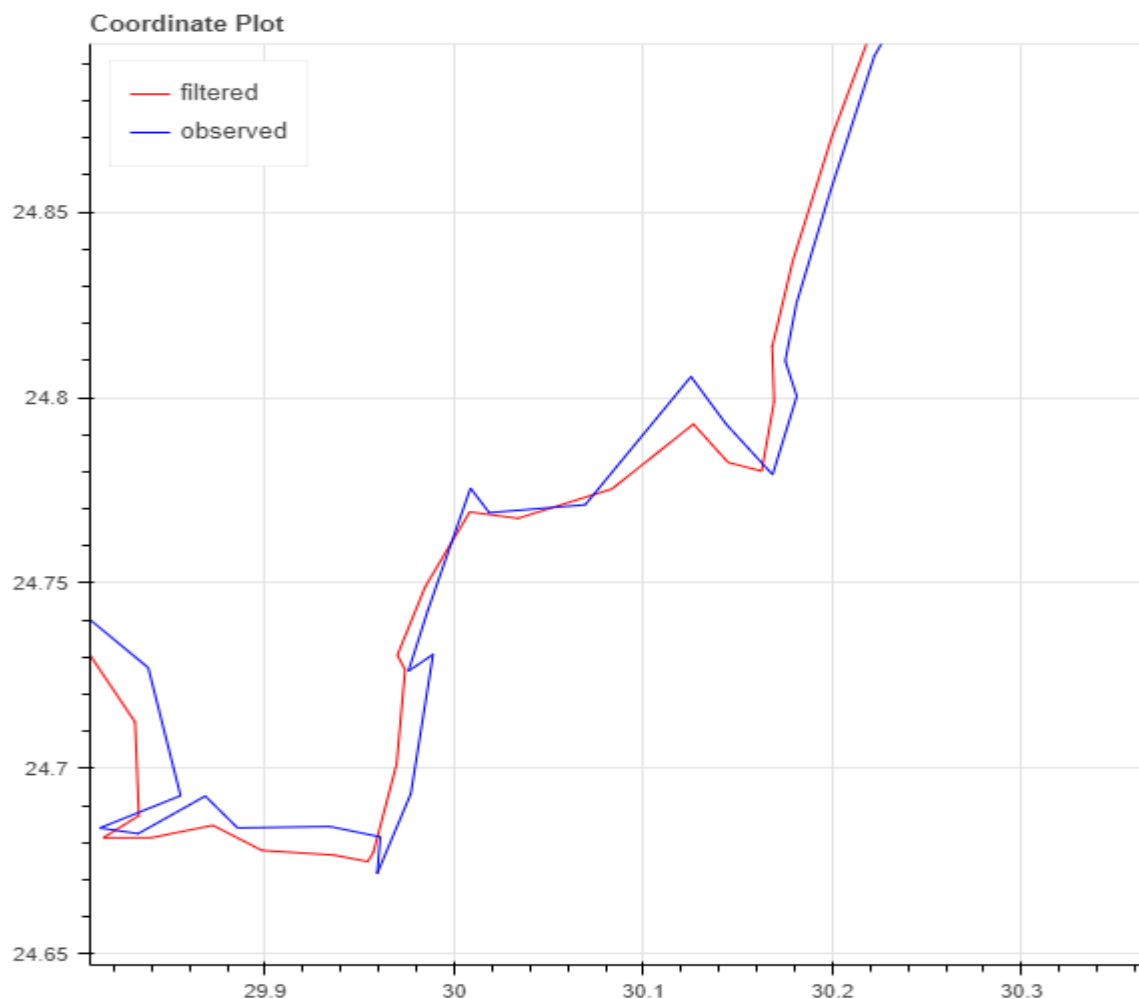


Figure 7: filtered and observed position.

7.0 References

- Introduction and Implementations of the Kalman Filter
<https://www.intechopen.com/chapters/63164>
- Positioning in 5G networks
<https://arxiv.org/pdf/2102.03361.pdf>
- Biagi, Ludovico. (2023). Kalman Filter: Theory, Positioning and Location Based services
- Special Topics - The Kalman Filter
<https://www.youtube.com/watch?v=tk3OJjKTDnQ>