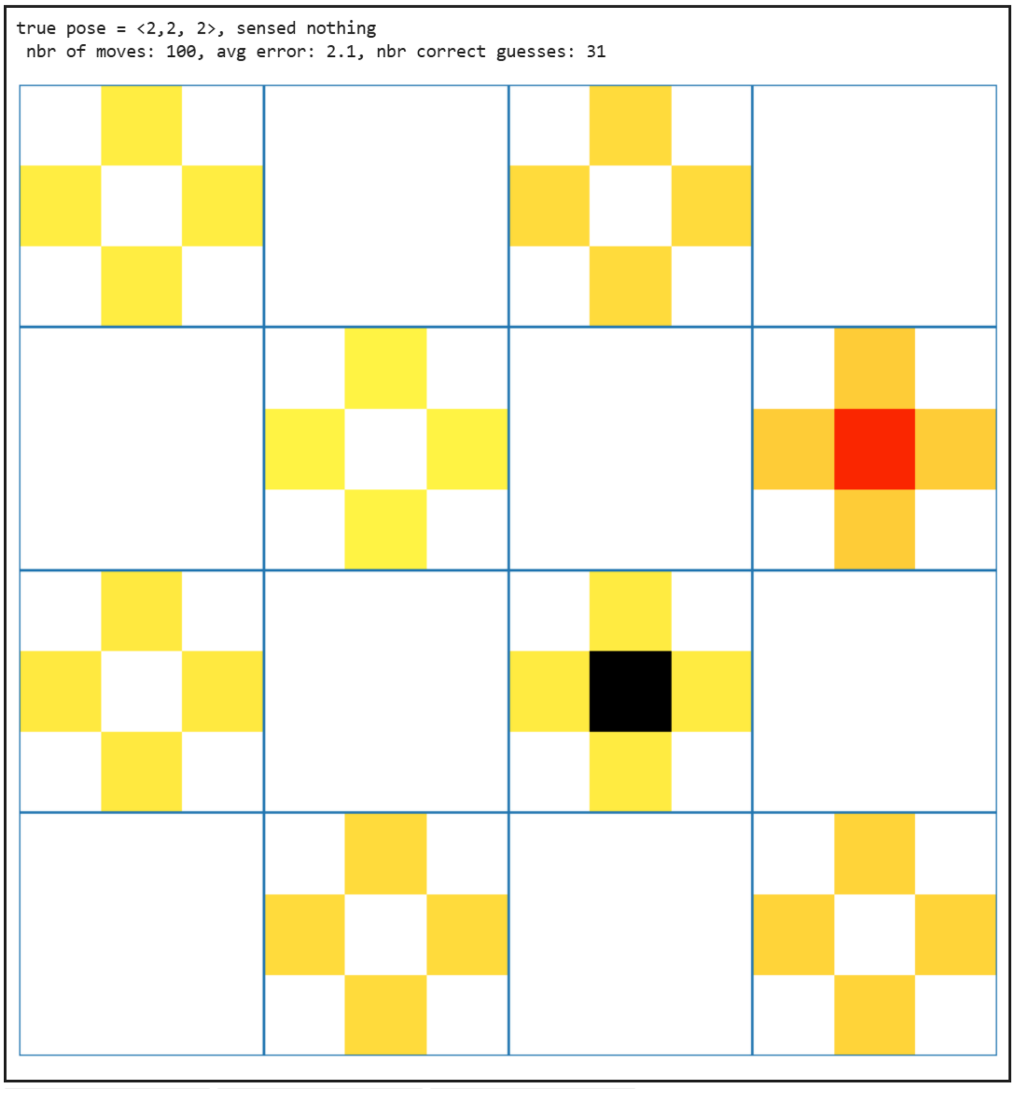
“Robot” localisation with HMM based forward-filtering



```
true pose = <2,2, 2>, sensed nothing
 nbr of moves: 100, avg error: 2.1, nbr correct guesses: 31
```

VAHID FARAJI

March 5, 2025

Contents

1 Introduction and Task Description

1.1 Overview

The task involves implementing a robot localization system using Hidden Markov Models (HMMs), with the primary objective of estimating the robot's position on a grid while incorporating noisy sensor readings. The robot's movement is simulated in a grid environment, and the goal is to track its position by processing sensor data in the presence of sensor failures. The system needs to estimate the robot's true location over time, and this estimation is improved using two methods: **Forward Filtering** and **Forward-Backward Smoothing**. These methods are based on the probabilistic framework of HMMs, which are applied to a discrete grid environment where the robot moves and senses its surroundings.

The core task is to evaluate the localization performance using different sensor models (non-uniform sensor failure and uniform sensor failure) and compare the effectiveness of **Forward Filtering** versus **Forward-Backward Smoothing** under different configurations. The robot's position is tracked by comparing the estimated positions with the true positions, and this comparison is made using the average Manhattan distance.

1.2 Theoretical Background

Hidden Markov Models (HMMs):

- **HMMs** are a statistical tool used for modeling systems that are governed by hidden states. In the context of robot localization, the hidden state represents the robot's true position on the grid, which is not directly observable. Instead, the robot's sensor readings provide noisy observations that help infer the hidden state.
- The model consists of two main components:
    1. **Transition Model**: Describes the probability of moving from one state to another based on the robot's motion and environment.
    2. **Observation Model**: Describes the likelihood of observing certain sensor readings given the current state.

Forward Filtering:

- This method is used to estimate the robot's position in real time based on the previous estimated state and the new sensor reading. It propagates the probability distribution of the robot's location over time by applying the **Bayesian update** rule.

Forward-Backward Smoothing:

- Unlike Forward Filtering, which only uses the current and previous sensor readings, **Forward-Backward Smoothing** utilizes both past and future data to refine the estimate of the robot's location. This method typically leads to more accurate results since it considers the entire sequence of sensor readings.

1.3 Problem Setup

The robot operates on a grid with a specified size (such as 4x4, 8x8, 16x20, or 10x10), and it receives sensor readings at each step. These readings may be unreliable due to sensor failures, which are modeled in two ways:

1. **Non-uniform Sensor Failure (NUF)**: The sensor has a non-uniform failure rate across different states, meaning that some states are more likely to fail than others.
2. **Uniform Sensor Failure (UF)**: The sensor failure rate is uniform, meaning that each sensor reading has an equal probability of failure.

The robot's task is to navigate through the grid while its position is continuously estimated using the methods of forward filtering and forward-backward smoothing. The effectiveness of these methods is evaluated by measuring the Manhattan distance between the estimated and true positions.

In summary, the report evaluates the performance of HMM-based localization techniques under various conditions (different grid sizes and sensor failure models), with the aim of improving the robot's ability to localize itself in an uncertain environment. The key metric for performance is the **average Manhattan distance** between the estimated and true positions.

2 Filtering and Smoothing Methods

The two main techniques used for estimating the robot's position are **Forward Filtering** and **Forward-Backward Smoothing**.

2.1 Forward Filtering

Forward Filtering is a recursive algorithm used to update the robot's belief about its position over time. The algorithm operates by maintaining a probability distribution over the robot's state at each time step. Given the robot's previous position estimate and the new sensor reading, the filtering process updates the probability distribution to reflect the new information. The filtering method is implemented using the Bayes rule, where the belief is updated based on both the transition and observation models.

At each step, the Forward Filtering method produces a belief that represents the most probable position of the robot given all previous observations and movements. However, this method only uses the available information up to the current time step and does not utilize future data.

2.2 Forward-Backward Smoothing

Forward-Backward Smoothing improves upon Forward Filtering by considering both past and future sensor readings. In this method, the robot's position is first estimated using Forward Filtering, and then a backward pass is performed to refine the estimate using all available sensor readings. This backward pass updates the estimate by incorporating future sensor readings, leading to a more accurate estimate of the robot's position.

Forward-Backward Smoothing typically results in a lower error compared to Forward Filtering because it uses a broader range of data. However, it requires access to the entire sequence of sensor readings, making it computationally more expensive and less suitable for real-time applications.

2.3 Evaluation Metrics

To evaluate the performance of the localization methods, the following metrics are used:

- **Manhattan Distance**: The Manhattan distance between the estimated position and the true position is used as the primary evaluation metric. It measures the absolute difference between the estimated and actual positions along the grid's axes.

- **Average Error**: The average Manhattan distance error is computed over multiple steps to evaluate the overall accuracy of the position estimation. A lower average error indicates better localization performance.
- **Smoothing Effect**: The effectiveness of Forward-Backward Smoothing is evaluated by comparing the smoothed errors to those from Forward Filtering. The goal is to show how smoothing improves the accuracy of the robot's position estimate over time.

3 Answer to Task 1

3.1 Observable Difference Between the Two Sensor Models (NUF and UF)

The Non-Uniform Sensor Failure (NUF) model has a varied failure rate across different states. Some positions in the grid are more prone to sensor failure than others, affecting the reliability of the sensor readings. In contrast, the Uniform Sensor Failure (UF) model assumes that the sensor has an equal chance of failing across all positions on the grid.

3.2 Impact on Localization with Forward Filtering

With NUF, localization errors might be higher in certain areas of the grid where sensor failure is more likely. The Forward Filtering method may struggle to provide accurate position estimates in these high-failure areas, but over time, the filter compensates by relying on previous data.

UF, on the other hand, provides a more consistent failure rate across the grid. However, its performance in Forward Filtering might be more predictable but less robust in areas with high sensor failure.

3.3 "No Reading" Visualization

The "No Reading" visualization highlights the areas where the sensor fails to provide any data. In the NUF model, these areas will be scattered across the grid according to the failure probabilities, while in the UF model, the failure will be uniform. This difference in failure distribution affects how well the Forward Filtering algorithm can estimate the robot's position, especially in regions where no sensor reading is available.

This observation helps explain how sensor reliability and sensor failure distributions directly influence the localization accuracy with filtering methods. The more reliable the sensor model, the more accurate the position estimate will be.

4 Comparison of HMM and Monte Carlo Localization (MCL)

To address the question, *"Is the HMM approach as implemented suitable for solving the problem of robot localization?"*, with reference to the article *"Monte Carlo Localization: Efficient Position Estimation for Mobile Robots"* by Dieter Fox et al., we consider the following:

4.1 HMM-Based Approach for Localization

The HMM-based approach in our implementation utilizes probabilistic models, such as the non-uniform and uniform sensor failure models, allowing for robot localization in uncertain environments. This method relies on belief updates using Forward Filtering and Forward-Backward Smoothing to estimate the robot's position based on sensor readings and movement transitions.

4.2 Monte Carlo Localization (MCL)

Monte Carlo Localization (MCL), as presented in the article, offers a more efficient and accurate solution, particularly in complex environments where multi-modal distributions exist. Unlike grid-based Markov localization, MCL utilizes a sample-based density approximation, which allows it to:

- Adapt the sample set size dynamically based on the robot's uncertainty.
- Provide higher accuracy, especially in global localization, where the robot has no prior knowledge of its position.
- Handle non-Gaussian, multi-modal probability distributions efficiently.

4.3 Comparison and Conclusion

While the HMM approach is useful for tracking the robot's position and adjusting beliefs based on sensor readings, MCL offers significant improvements in terms of:

- **Accuracy:** MCL can maintain precise localization by dynamically adjusting its sample density.
- **Computational Efficiency:** MCL reduces computational costs compared to grid-based methods by focusing resources on the most probable regions.
- **Scalability:** MCL is more suitable for large, complex environments where sensor reliability and movement uncertainty vary significantly.

Thus, while the HMM approach can work in some scenarios, Monte Carlo Localization offers a superior, more scalable, and computationally efficient solution for robot localization, especially in dynamic environments with varying sensor reliability.

This conclusion is drawn based on a direct comparison between the two methods, considering their strengths and limitations in different localization tasks.

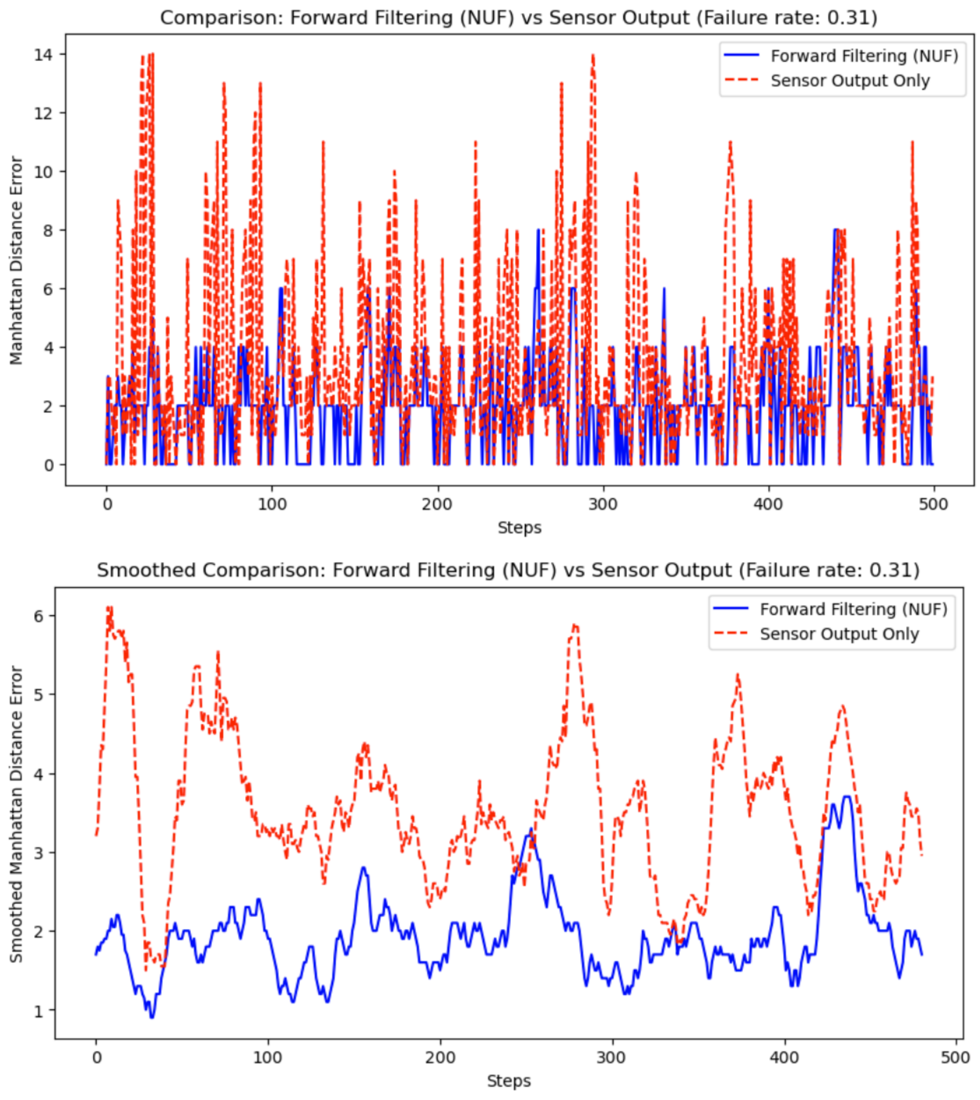
5 Results

5.1 Forward Filtering with a Non-Uniform Failure (NUF) VS Sensor output - Grid Size: 8×8

In this experiment, we compare two localization methods:

- **Forward Filtering (NUF):** This method uses a probabilistic filtering approach, considering sensor readings over time to improve localization accuracy.
- **Sensor Output Only:** This method uses only the current sensor reading, ignoring previous readings or the historical state of the system.

The comparison was conducted under the condition of a 31% sensor failure rate, where the sensor occasionally fails to provide reliable readings. The grid size used in the experiment was 8×8, providing a space with a total of 64 possible positions for the robot to occupy.



```
Average error (NUF Filtering): 1.92
Average error (Sensor Output Only): 3.51
Sensor failure rate: 0.31
```

Figure 1: Comparison of NUF vs Sensor Output in a 8×8 grid

5.1.1 Raw Sensor Output (Dashed Red Line in Graph)

- The error fluctuates significantly, with many high peaks.
- The average Manhattan distance error is 3.51, indicating that the sensor readings alone are highly unreliable.
- High variance is observed, showing instability in raw sensor outputs.

5.1.2 Forward Filtering (Solid Blue Line in Graph)

- The error trend is more stable, and large fluctuations are significantly reduced.
- The average Manhattan distance error is 1.92, which is notably lower than the raw sensor output.
- The filtering process helps in providing a more accurate and consistent estimation of the robot's position.

5.1.3 Smoothed Graph Analysis

- After applying a moving average filter, the overall trend becomes clearer.
- Forward Filtering (NUF) consistently outperforms the raw sensor output, confirming the effectiveness of the filtering technique.
- The error remains relatively stable throughout the steps, demonstrating that the HMM-based approach is robust against sensor noise and failures.

5.1.4 sensor Failures Rate: %31

In the case of sensor failures, especially observed in the Non-Uniform Sensor Failure (NUF) model, the error fluctuations are significant, leading to a decrease in the robot's position estimation accuracy. These fluctuations are mainly due to the sensor failures, which introduce noise into the sensor readings. However, the Forward Filtering method effectively reduces these fluctuations, demonstrating the HMM's capability to handle sensor failures and improve localization accuracy. Even with a high sensor failure rate of 31%, the average error continues to decrease, indicating the high efficiency of the HMM filter in various settings. This highlights the robustness of the filtering technique in overcoming sensor noise and ensuring more reliable position estimates.

5.2 Forward Filtering with a Non-Uniform Failure (NUF) VS Forward Filtering with a Uniform Failure (UF) - Grid Size: 4×4

5.2.1 Raw Error Trends (First Graph)

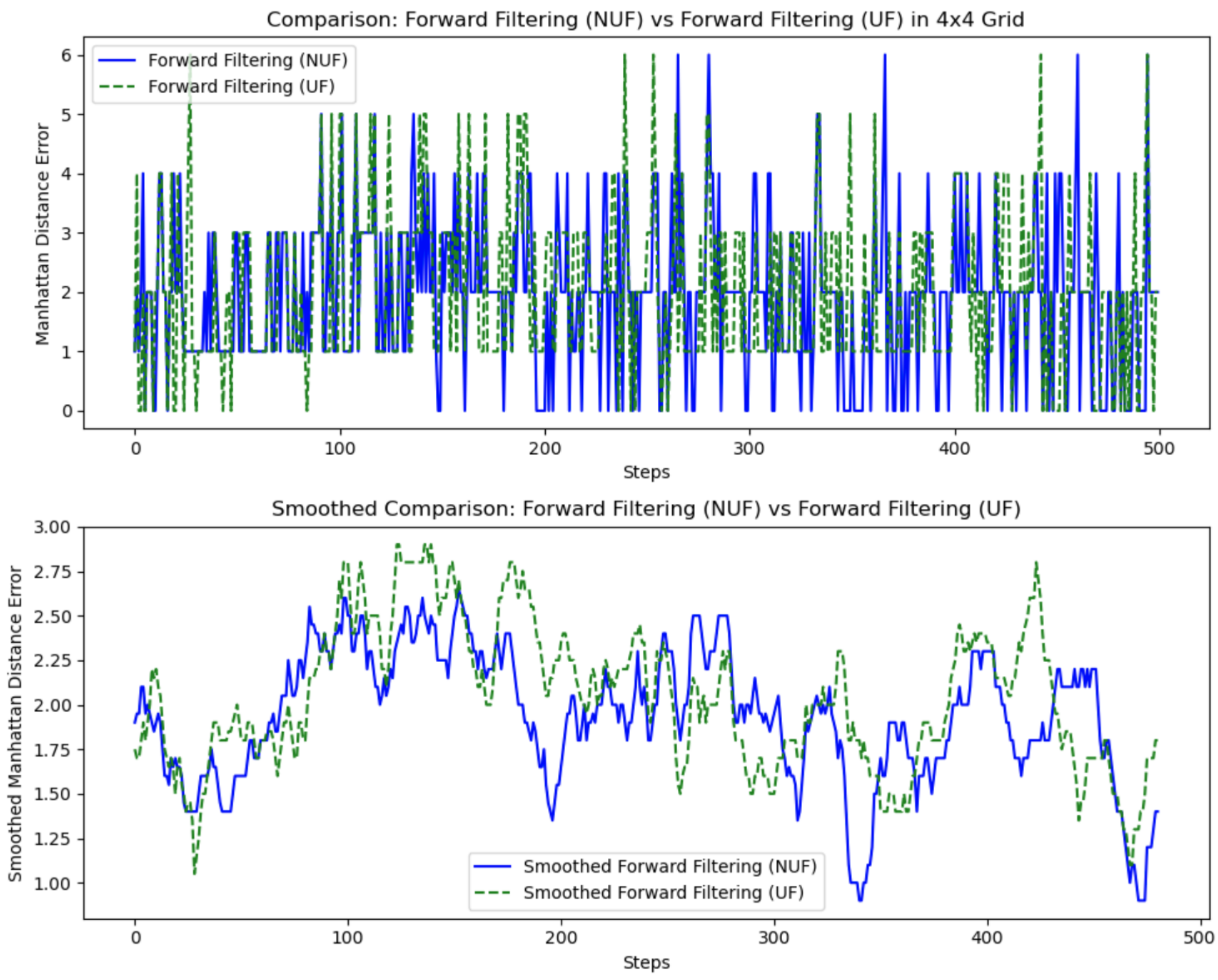
- Both NUF and UF filtering methods show fluctuations in error.
- Due to the small grid size, the error values change frequently.
- The visual difference between NUF and UF filtering is not highly significant.

5.2.2 Smoothed Error Trends (Second Graph - Moving Average Applied)

- Applying a moving average filter makes the trends clearer.
- NUF filtering (solid blue line) consistently performs better than UF filtering (dashed green line).
- The error remains relatively stable over time, showing that NUF filtering provides a slightly more accurate localization estimate.

5.2.3 Overall Error Comparison

- Average error with NUF Filtering: 1.93
- Average error with UF Filtering: 2.03
- The difference between NUF and UF is relatively small in this small grid, suggesting that the impact of sensor model differences is limited in small environments.



```
Average error (NUF Filtering): 1.93
Average error (UF Filtering): 2.03
```

Figure 2: Comparison of NUF and UF 4×4 grid

5.2.4 Conclusion

The results indicate that Forward Filtering (NUF) outperforms Forward Filtering (UF) in terms of accuracy, but the difference is marginal in a small 4×4 grid. The NUF model reduces error more effectively, but its advantage may become more significant in larger grids where errors accumulate over longer trajectories.

5.3 Forward Filtering with a Non-Uniform Failure (NUF) VS Forward Filtering with a Uniform Failure (UF) - Grid Size: 16×20

5.3.1 Raw Error Trends (First Graph)

- Both NUF and UF filtering methods show fluctuating error values.
- The two filtering approaches produce very similar error trends.
- The difference between NUF and UF is minimal in this larger grid.

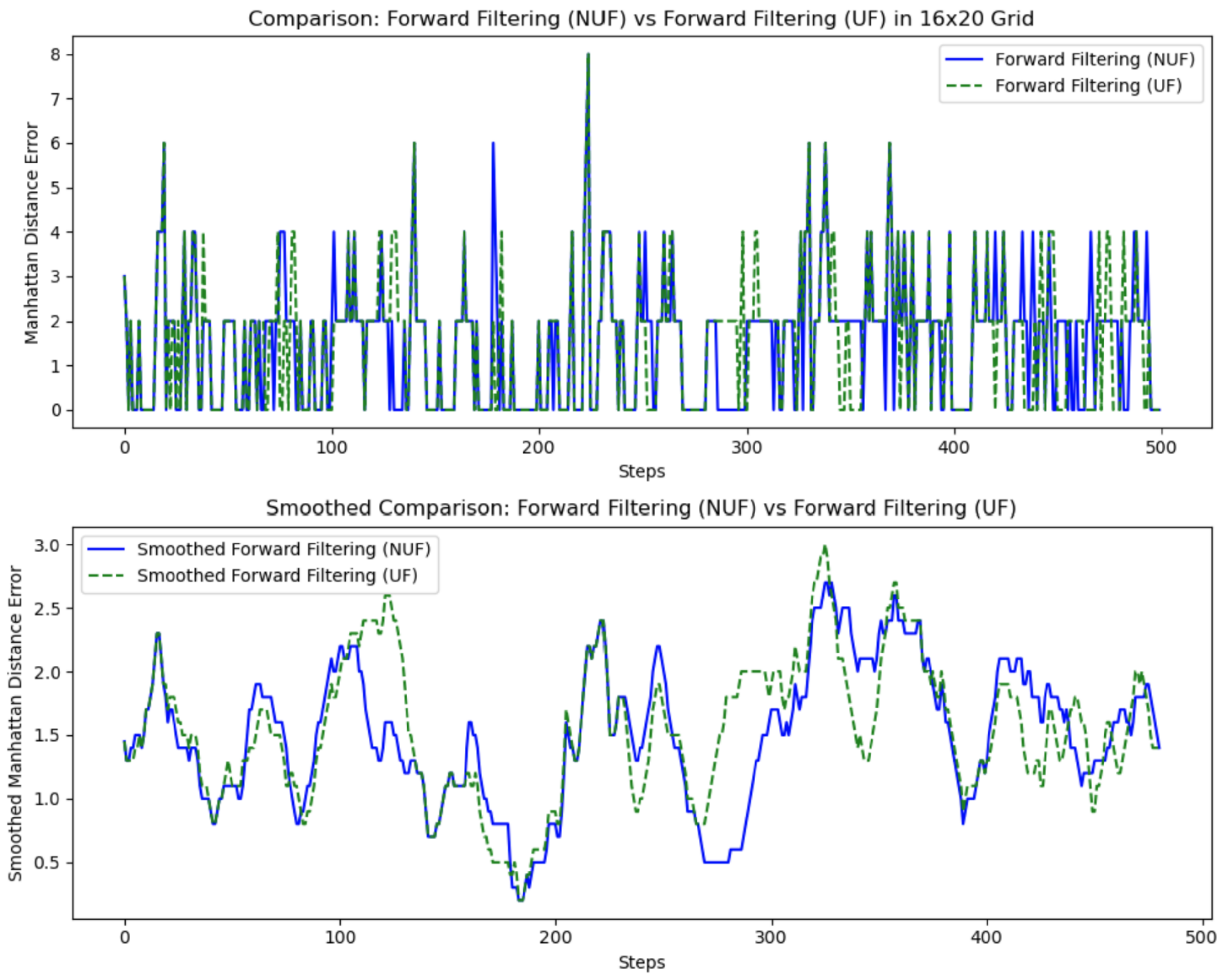
5.3.2 Smoothed Error Trends (Second Graph - Moving Average Applied)

- NUF filtering (solid blue line) and UF filtering (dashed green line) perform almost equally.

- The overall error trend is more stable after applying the moving average.
- NUF has slightly lower error in some sections but does not significantly outperform UF.

5.3.3 Overall Error Comparison

- Average error with NUF Filtering: 1.49
- Average error with UF Filtering: 1.52
- NUF performs slightly better than UF, which aligns more closely with theoretical expectations.
- Compared to previous experiments, the difference remains relatively small, suggesting that the impact of the sensor model is less pronounced in large grids.



```
Average error (NUF Filtering): 1.49
Average error (UF Filtering): 1.52
```

Figure 3: Comparison of NUF and UF in a 16 × 20 grid

5.3.4 Conclusion

Unlike previous experiments where NUF consistently outperformed UF, the difference in accuracy is minimal in this larger 16 × 20 grid. This could indicate that:

- As the grid size increases, the impact of sensor model differences diminishes.
- Motion models may dominate error correction, reducing the effect of different sensor models.
- Filtering alone may not be sufficient in larger environments, and additional techniques like smoothing could improve accuracy further.

5.4 Forward Filtering with a Non-Uniform Failure (NUF) VS Forward-Backward Smooothing - Grid Size: 10×10

5.4.1 Raw Error Comparison (Top Graph)

- **Forward Filtering (NUF)** shows significant fluctuations in the error, with the robot's position estimation varying between steps. The error peaks are often quite high, which is expected due to the sensor failures and the uncertainty inherent in the system.
- **Forward-Backward Smoothing (Modified)** exhibits a similar pattern but with more noticeable reductions in error at specific points. This suggests that the backward smoothing helps mitigate some of the noise in the position estimation by considering past and future observations together.

5.4.2 Smoothed Error Comparison (Bottom Graph)

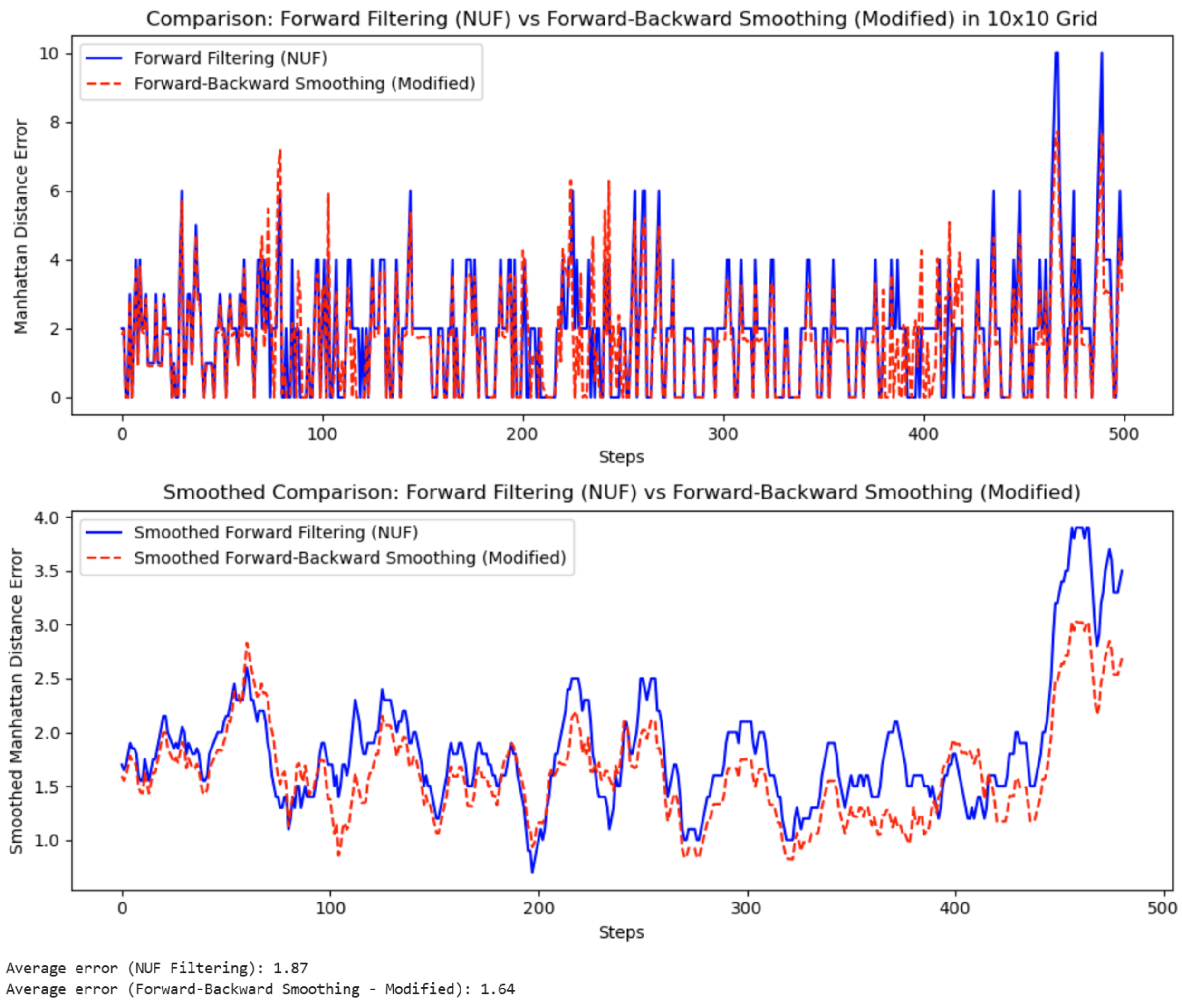
- After applying a moving average to the raw errors, the **Forward Filtering (NUF)** results show a much more consistent and stable error trajectory. The peaks are less frequent, indicating that filtering has effectively reduced the fluctuation.
- The **Forward-Backward Smoothing (Modified)** also smooths the error, but there are still noticeable fluctuations, although they are less severe than in the raw error graph. This suggests that while smoothing does help reduce some errors, the overall trend is still similar to the raw error, indicating that the forward-filtering technique already handles the majority of the noise.

3. Key Findings

- **Average Error Comparison:**
  - The average error for **Forward Filtering (NUF)** is **1.87**.
  - The average error for **Forward-Backward Smoothing (Modified)** is slightly lower at **1.64**.
- This shows that while both techniques are relatively close in terms of error reduction, smoothing provides a slight improvement over pure forward filtering. However, the difference in performance is minimal.

4. Conclusion

The results demonstrate that both **Forward Filtering (NUF)** and **Forward-Backward Smoothing (Modified)** provide reliable localization in the 10x10 grid. The slight improvement in average error through smoothing suggests that it can help in certain situations, especially when the system has high uncertainty. However, in this case, the difference in performance is marginal.



```
Average error (NUF Filtering): 1.87
Average error (Forward-Backward Smoothing - Modified): 1.64
```

Figure 4: Comparison of NUF and Forward-Backward Smoothing in a 10×10 grid

This implies that for environments where the robot is operating in a relatively stable or predictable manner, **Forward Filtering (NUF)** might be sufficient. In contrast, **Forward-Backward Smoothing** may be more beneficial when dealing with more dynamic or noisy environments, as it can use future observations to correct earlier estimations.