**Written by Zaynab Mourtada:**

# Machine Learning Plan for Adaptive HCI:

I’ll be taking a hybrid approach, with the Kalman Filter as the primary tool and the RNN model as a secondary layer for additional smoothing as needed. After implementing both, I’ll evaluate their combined effectiveness to ensure everything runs smoothly and accurately. Once that’s solid, we can expand the model to include letter recognition training.

## Dataset Preparation

* **Start with Existing Handwriting Datasets**: I’ll use online datasets that include 2D trajectories, which can then be expanded to 3D. I’ll hand over this task to the 3D team.
* **Simulate Noise**: I will add noise to the datasets, so I can train the model to handle shaky handwriting, making the data more realistic.
* **Preprocess Data for Model Use**: The datasets will be normalized and scaled, ready for the Kalman filter and RNN model to process.

## Kalman Filter for Primary Smoothing

* **Noise Reduction**: The Kalman filter will be used first to clean up the raw glove data and reduce high-frequency noise.
* **Optimized for Mobile**: I’ll fine-tune the filter so it runs smoothly on mobile, balancing speed and effectiveness.

## RNN Model as a Backup

* **Secondary Measure**: I’ll use a GRU-based RNN model to further smooth any remaining noise that the Kalman filter doesn’t handle fully.
* **Conditional Activation**: The RNN will only kick in if the noise level is high enough, so it won’t slow things down unnecessarily.
* **Mobile Optimization**: I’ll also make sure the RNN is optimized for mobile performance, so the app stays responsive.

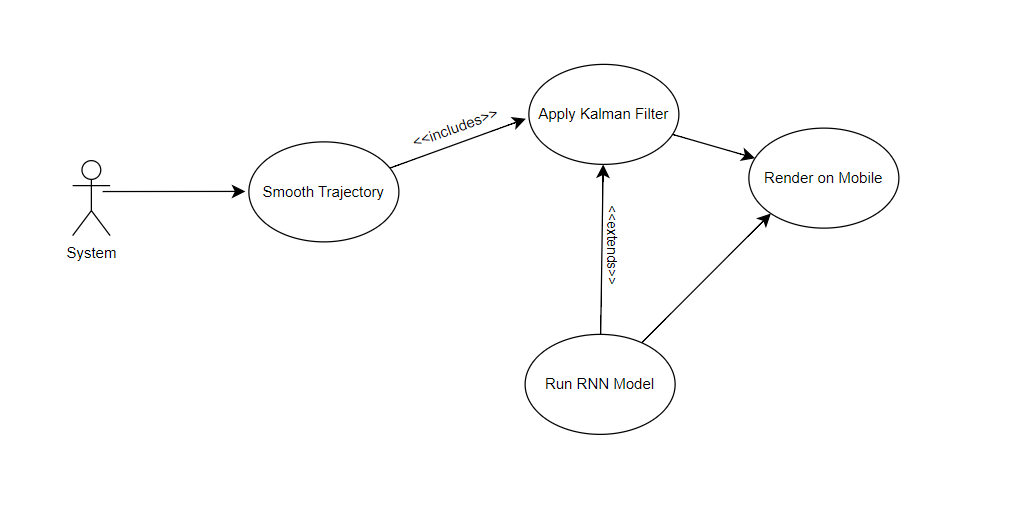
## Evaluation

* **Measure with MSE**: I’ll use Mean Squared Error to check how well the Kalman filter and RNN are working together. This will help ensure the final output is smooth and accurate.
* **Real-Time Testing on Mobile**: Testing will include running on actual mobile devices to confirm that everything functions in real-time as intended.

**Zhang’s recommendation:**

* Do more research on related work for smoothing moving trees or smooth curves.
* For the purpose of convincing you which is the best approach to take.

# Machine Learning Use Case:



**Explanation:**

This diagram illustrates the machine learning process. The “**System”** actor initiates “**Smooth Trajectory”**, which includes two steps:

1. **“Apply Kalman Filter”**: The primary method for noise reduction, always included in the smoothing process.
2. **“Run RNN Model”**: A secondary method that extends **Apply Kalman Filter** if additional smoothing is required. It activates only when the Kalman filter’s output needs further refinement.

After smoothing, the result is **“Rendered on Mobile”** to display the final trajectory to the user.

# New approach: 10/15/2024

1. **Data Collection:**
   1. Use existing online 3D training sets for air-writing
2. **Preprocessing:**
   1. Apply bicubic interpolation to standardize input signal length
   2. Use Kalman Filter for trajectory smoothing
3. **Feature Extraction:**
   1. Extract kinematic and geometric features
   2. Include frequency domain and entropy-based features
4. **Normalization:**
   1. Normalize extracted features before feeding into the neural network
5. **Deep Learning Model:**
   1. Implement a hybrid approach using LSTM and CNN networks
   2. CNN for learning spatial features
   3. LSTM for capturing temporal dependencies

## Resources:

<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8705512/pdf/sensors-21-08407.pdf>

<https://www.ijraset.com/best-journal/a-machine-learning-approach-for-air-writing-recognition>

<https://www.mdpi.com/1424-8220/20/2/376>