University of Stuttgart
Germany

Real Time Prediction of Saccade Landing Position

Niteesh Balachandra Midlagajni
Sowmyashree Shripathi
Jagan Shanmugam

Institute for Visualisation and Interactive Systems

Outline

Motivation and Objective

- Gaze contingent displays render high quality images only in the area where eyes are fixated. In case of saccadic eye movements, performance is hindered due to system latency and is therefore unable to render the saccade end point area on time

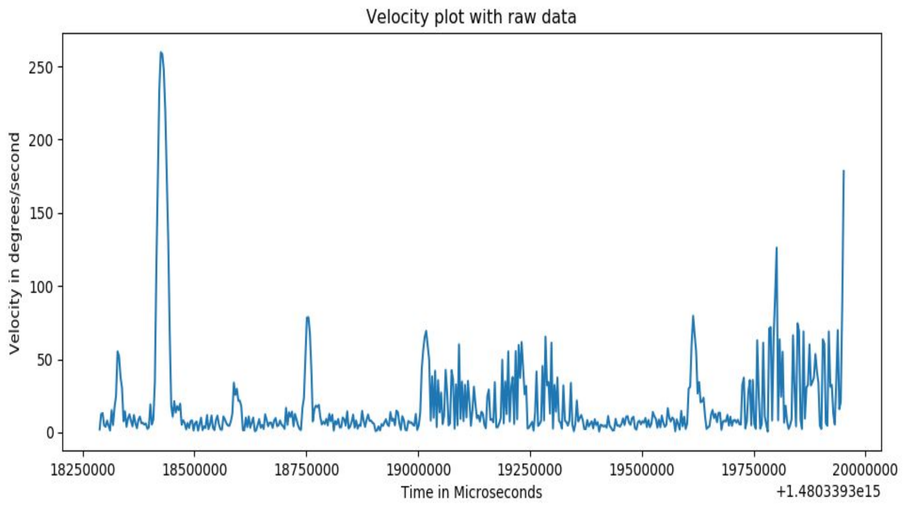
- Objective here is to predict the landing point of the saccade in real time using Deep learning techniques way ahead of time, giving sufficient time for pre-rendering

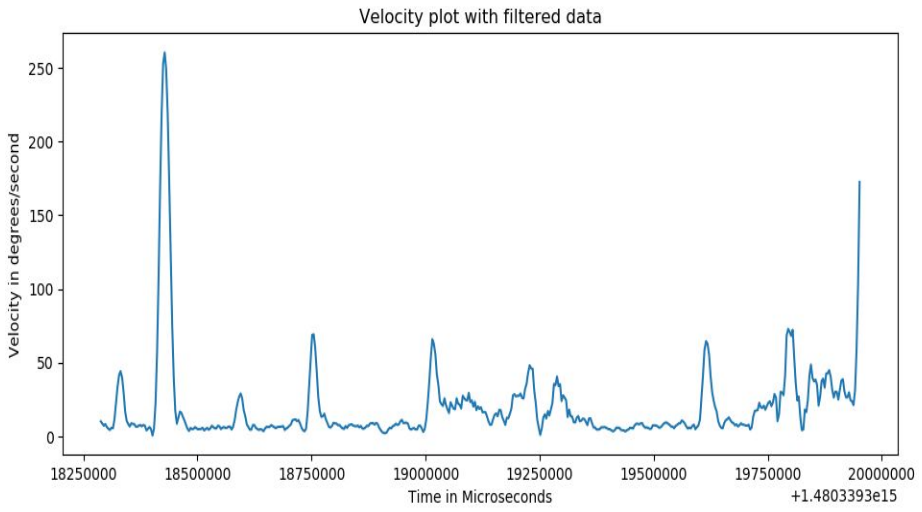
Dataset for project

- Raw gaze data taken from paper [1]
  - Collected using Tobii TX300 @ 300Hz
  - 22 participants
  - Contains gaze coordinates and corresponding time stamps

Data Analysis

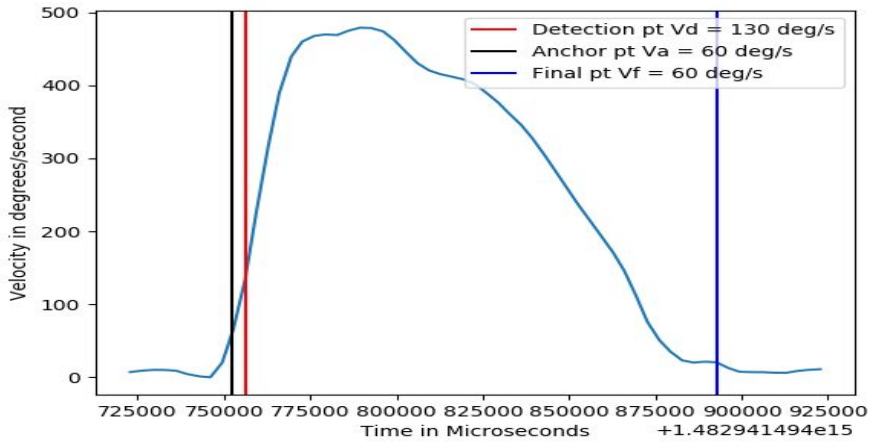
- Velocity calculation
  - Algorithm: 2-point central difference method (Window len = 5)
  - Smoothing: Savitzky-Golay filtering (Order = 3, Window Length = 14) [2]

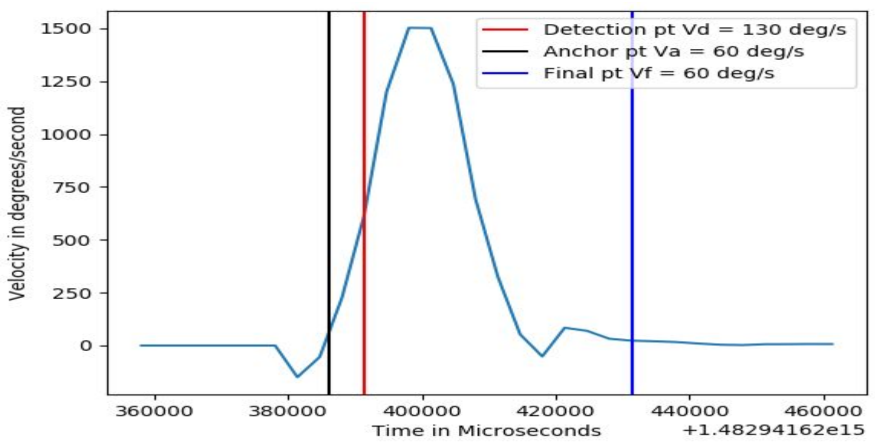




Saccade Detection and Extraction(1)

- Classification technique: Velocity–Threshold Identification(I–VT)
  - Fastest technique to distinguish saccades [5]
- Saccade detection: Algorithm based on [1]
  - Once velocity crosses saccade threshold(V_d), scan back in time to find beginning of saccade(V_a); end point is determined with another threshold(V_f)
  - Chosen values[1]: Anchor velocity V_a: 60°/s, Detection velocity V_d: 130°/s, Final velocity V_f: 60°/s



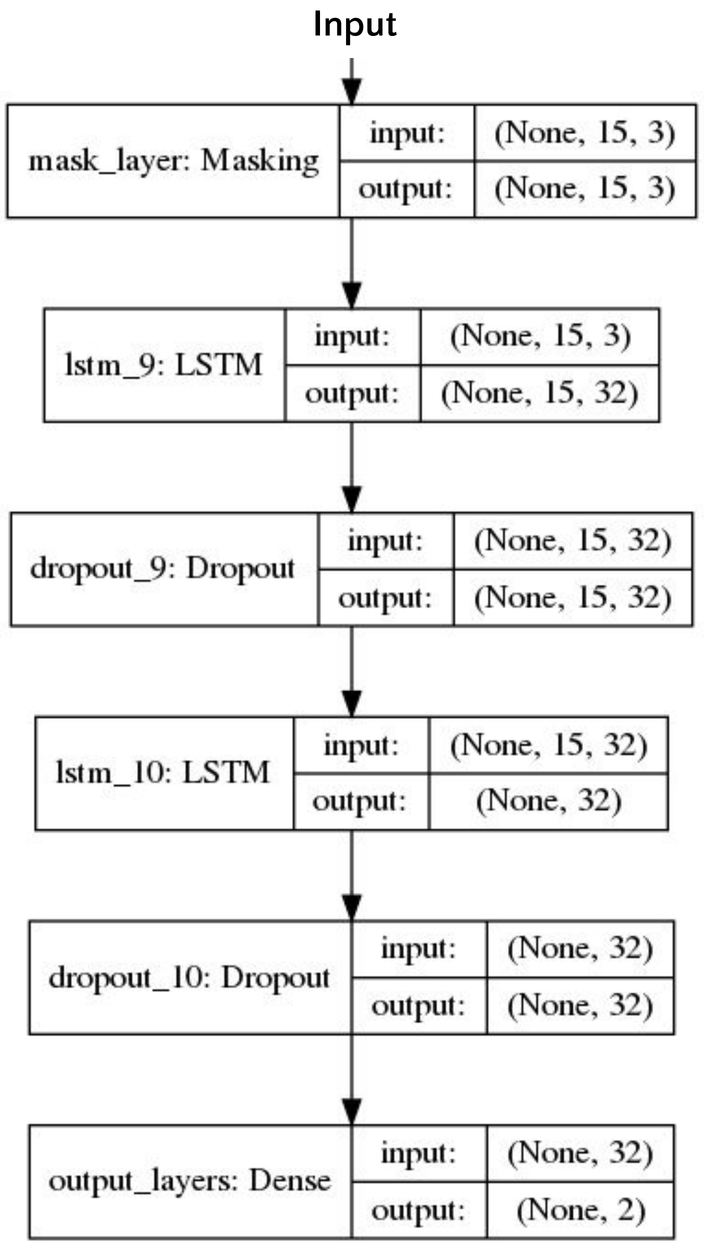


Saccade Detection and Extraction(2)

- For improved robustness in detection, the following techniques employed from [1]:
  - Saccades with anchor point within 30ms from detection point are only considered
  - Microsaccades (saccades less than amplitude of 5°) are rejected
  - Samples up-to 15ms after final point are also considered as part of saccade to account for glissades
  - Saccades with duration less than 15ms are rejected
- Extracted saccades analytics:
  - Total: 10770 saccades
  - Average saccade duration is 22 samples(73.62ms) with a high of 50 (166ms) and low of 12 (40ms)
- Extracted saccades(from anchor to final points) saved as npy files

Model Architecture

- Input: 3 features
  - xy-coordinates, velocity
- First layer: Masking layer to consider variable length saccades
- Output: xy-coordinates
- Custom metrics:
  - Loss: Euclidean distance
  - Additional Metric: Visual degree error
- Regularization: Dropout
- Trainable params: 12,994
- Non-trainable params: 0



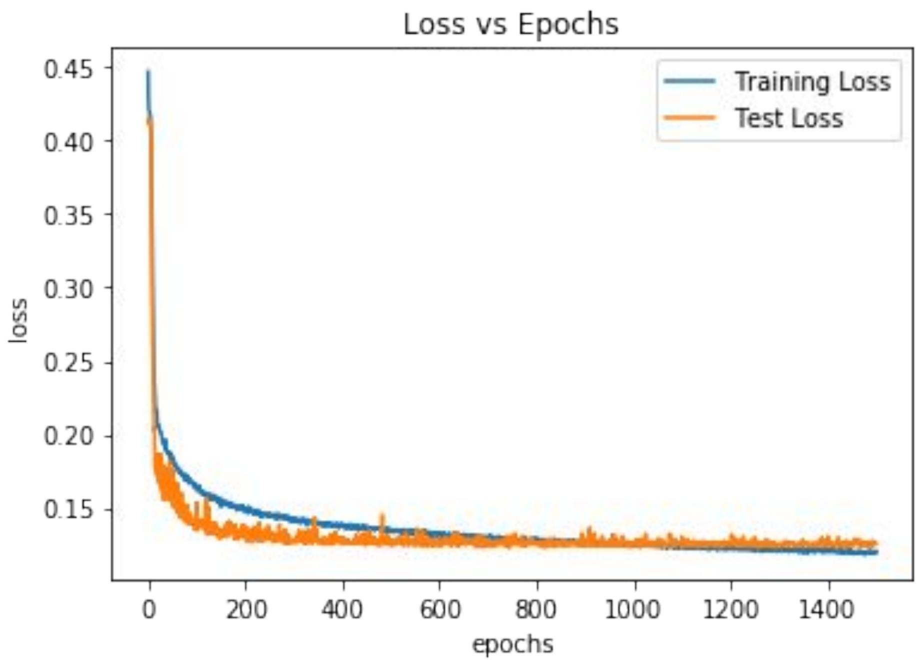
Hyperparameters tuning

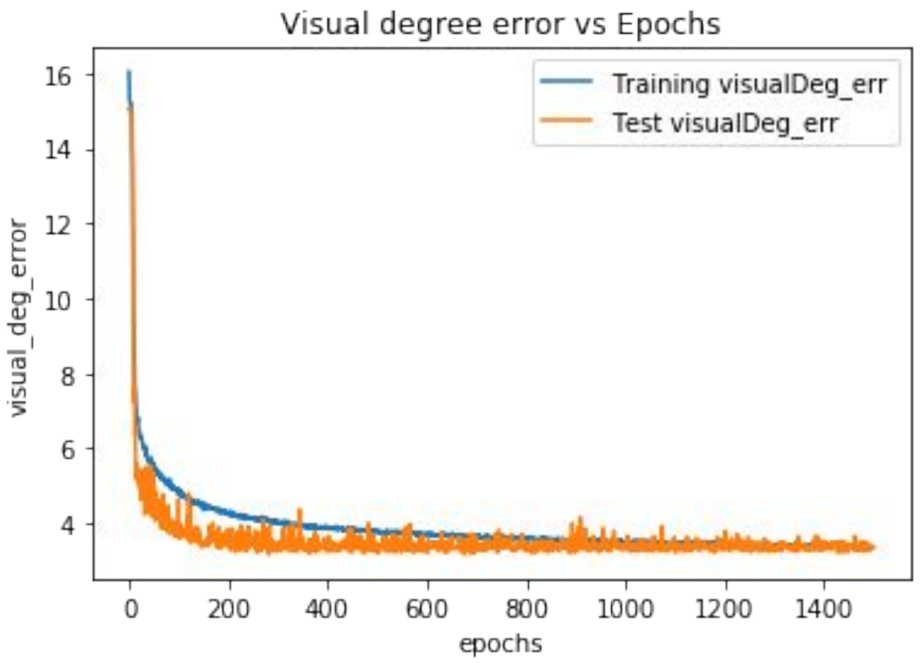
- Tuning learning rate and decay rate had large impact on the model learning, fixed it once an optimum combination was found
  - Adam Optimizer with LR = 0.0005 and decay rate = 1e^{-5}
- Dropout technique was sufficient to overcome overfitting

Model	Training loss	Training visual degree error	Validation loss	Validation visual degree error
3 LSTM layers (32 cells each)	0.0119	3.1014	0.0171	3.4061
3 LSTM layers (16 cells each)	0.0176	3.8175	0.0263	4.7943
2 LSTM + 1 Dense layers (32 cells each)	0.0135	3.2511	0.0170	3.3780
2 LSTM layers (32 cells each) (without velocity)	0.0134	3.271	0.0161	3.0615
2 LSTM layers (32 cells each)	0.0145	3.3854	0.0158	3.3658
2 LSTM layers (16 cells each)	0.0195	3.9591	0.0243	4.9476

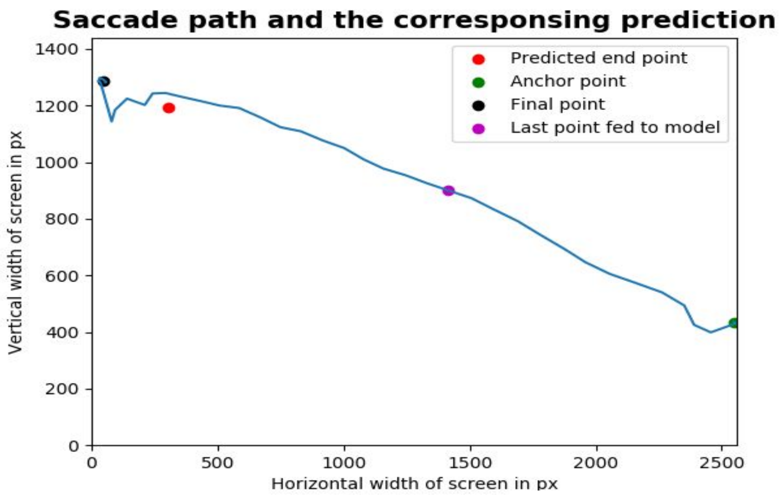
Visualizations: Model learning curve

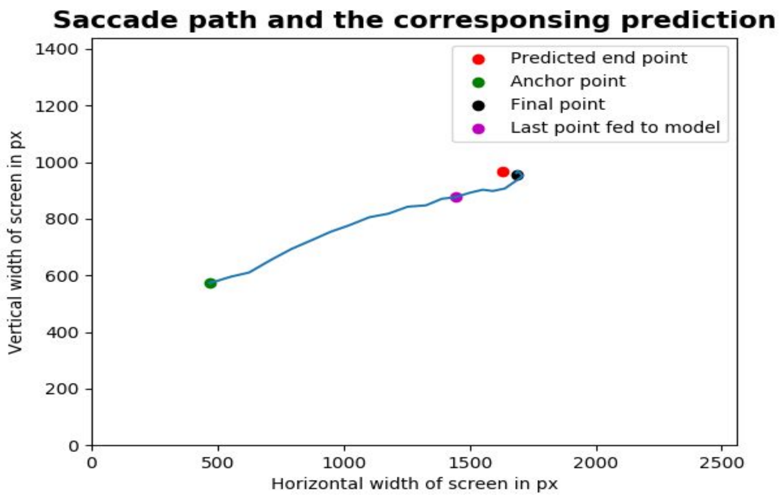
- Learning rate: 0.0005
- Decay rate: 1e^{-5}
- Dropout 1: 0.18, Dropout 2: 0.12
- For 2 LSTM layers (32 cells each), learning curve is given below.

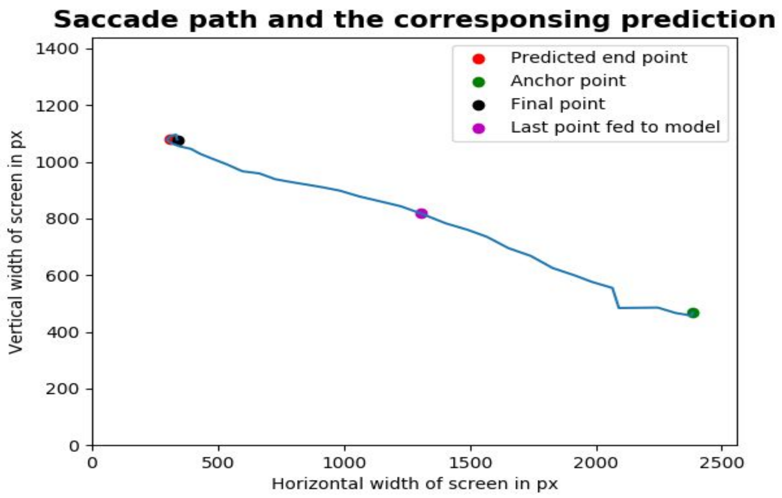




Visualizations: Saccade path and model prediction





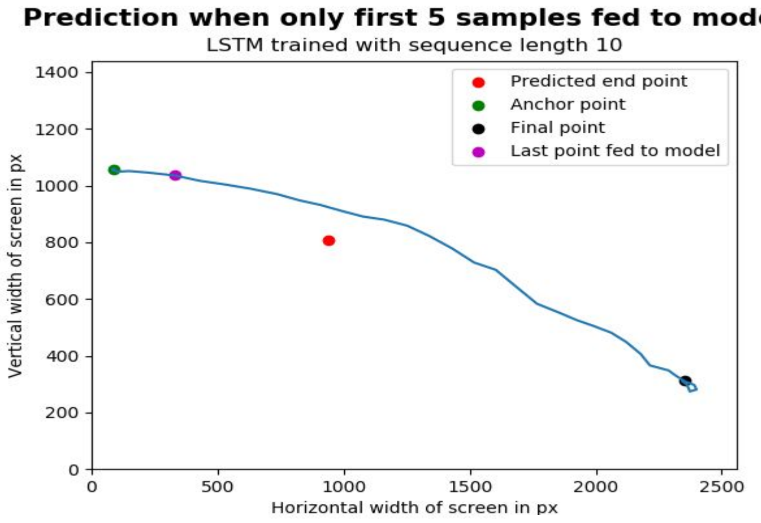


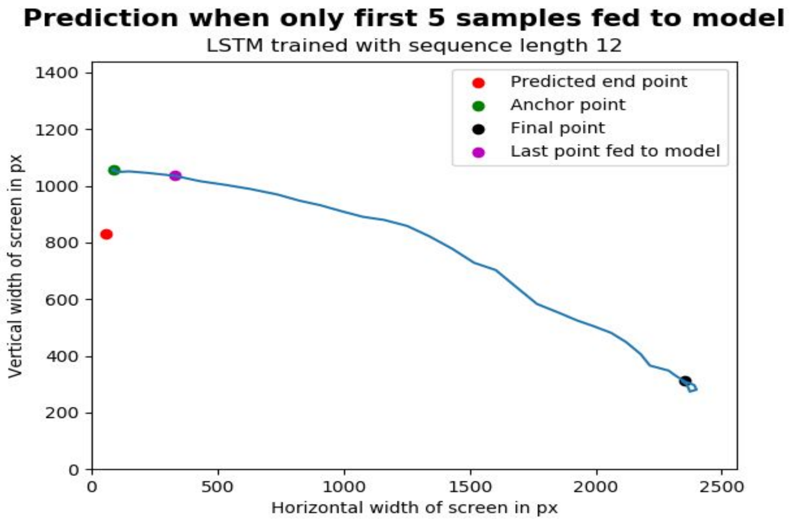
Training for different time samples

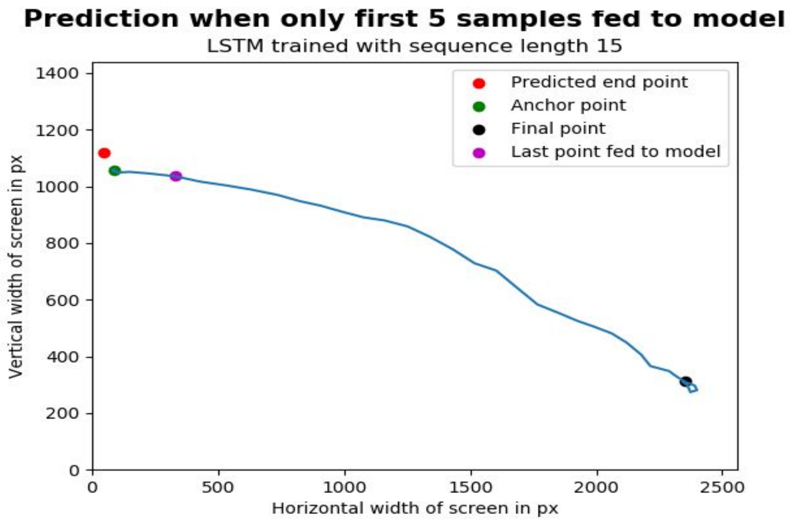
- For the selected model(trained with 15 time samples), training was repeated for time samples of 10 and 12

Time sequence (2 LSTM layers (32 cells each) model)	**Training loss**	**Training visual degree error**	**Validation loss**	**Validation visual degree error**
15 samples	0.0145	3.3854	0.0158	3.3658
12 samples	0.0227	4.1614	0.0257	3.5588
10 samples	0.0229	4.3333	0.0277	4.1848

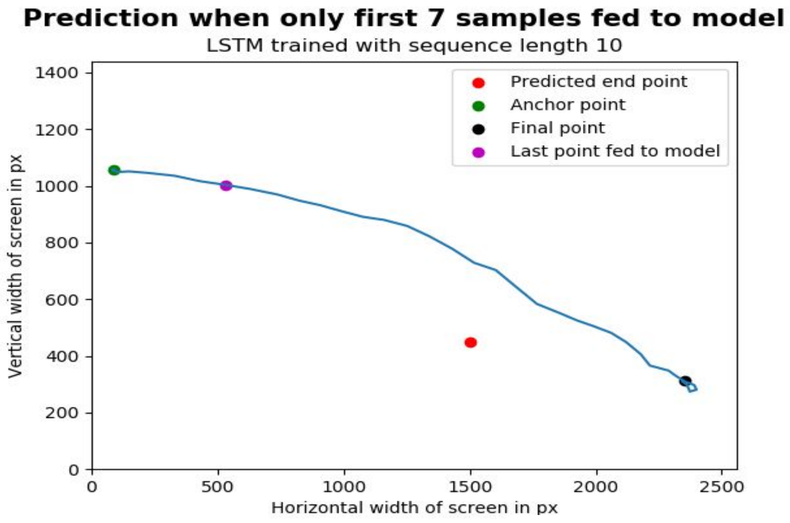
Visualizations: Early prediction performance – First 5 samples

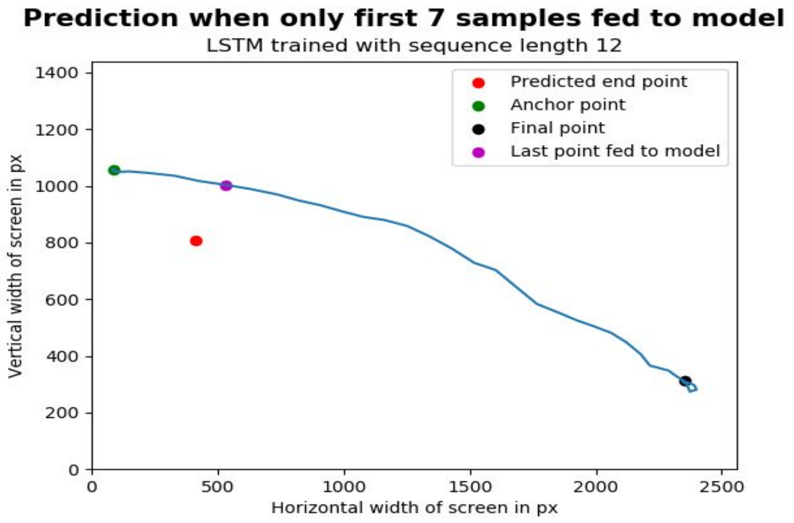


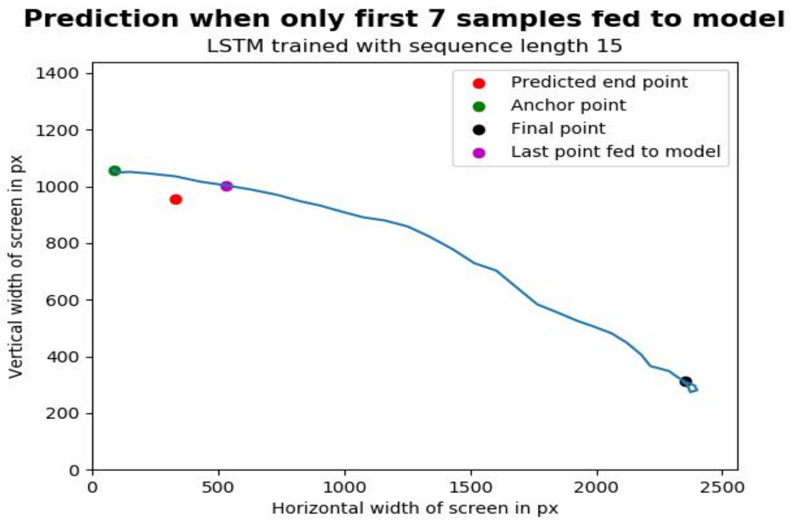




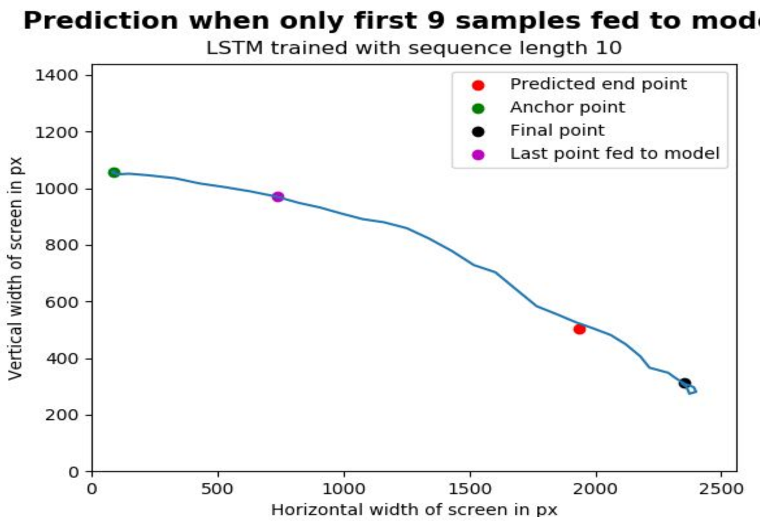
Visualizations: Early prediction performance – First 7 samples

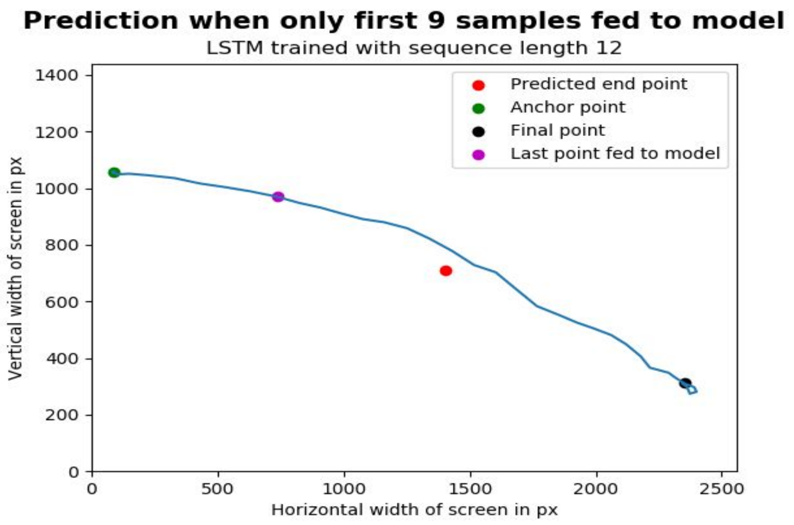


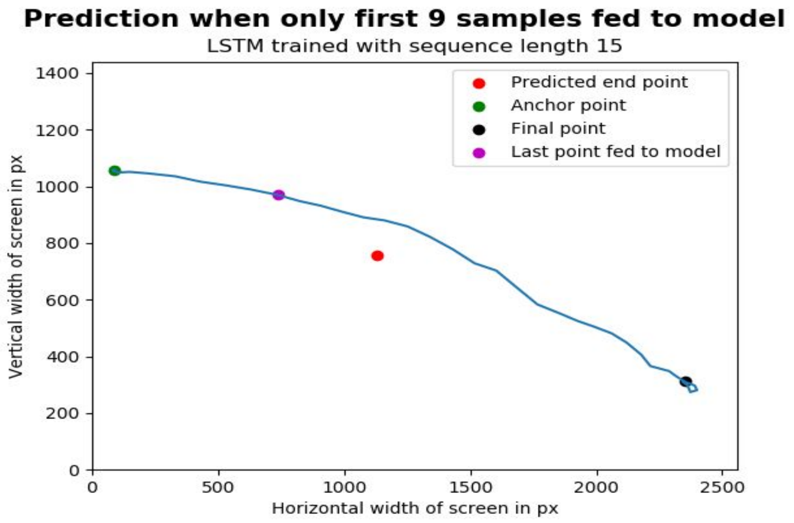




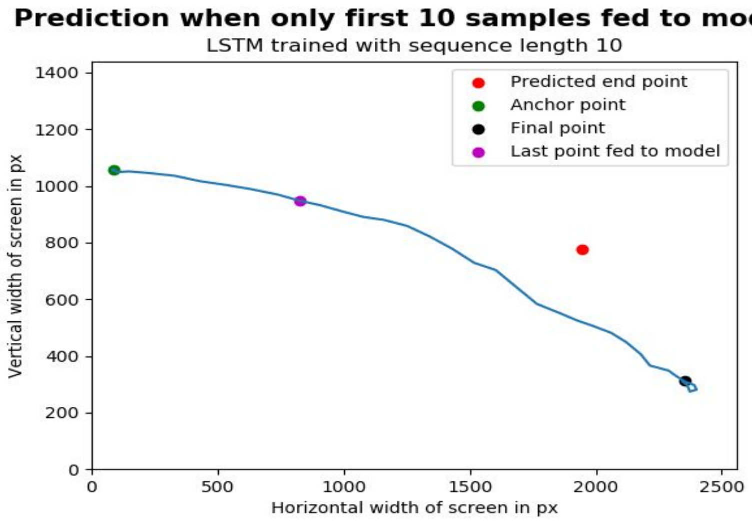
Visualizations: Early prediction performance – First 9 samples

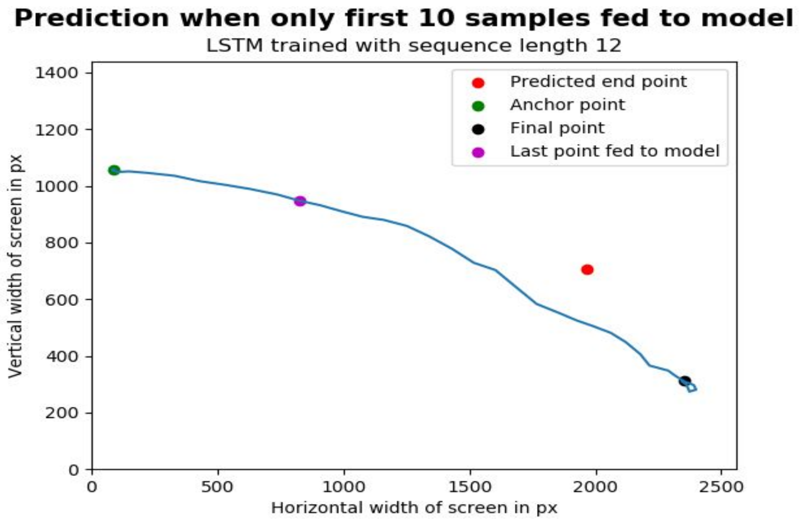


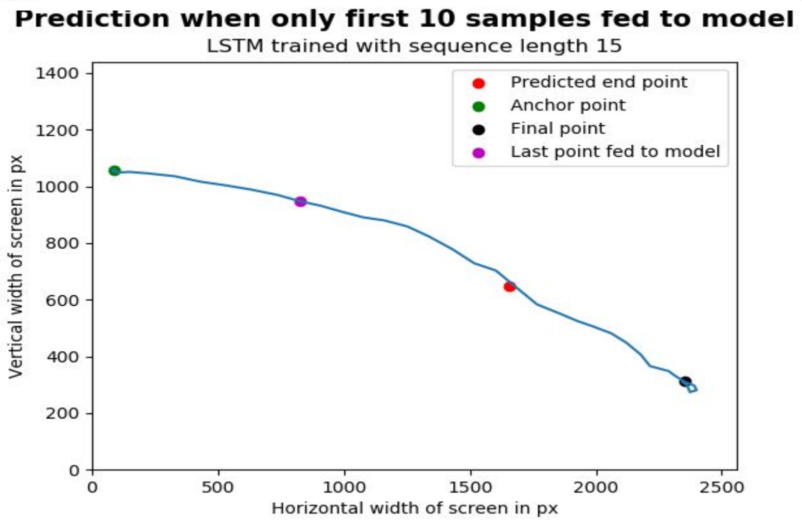




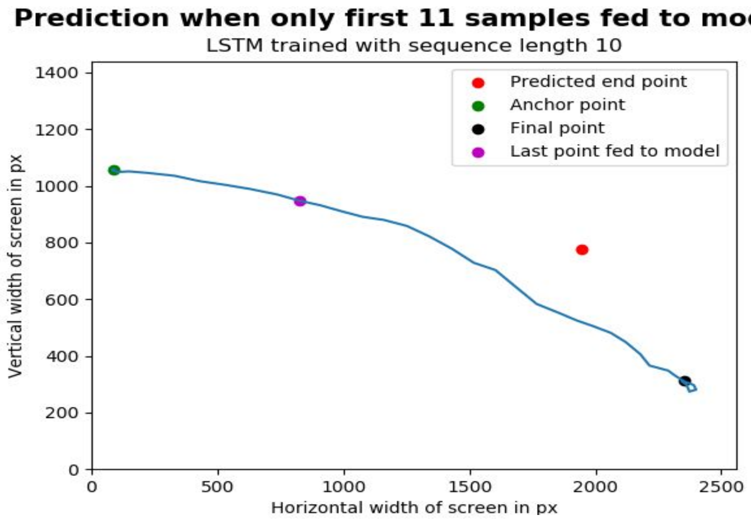
Visualizations: Early prediction performance – First 10 samples

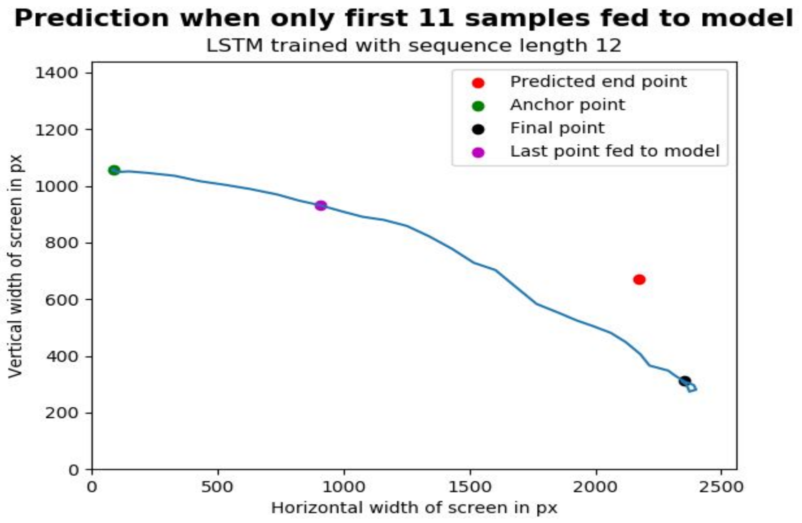


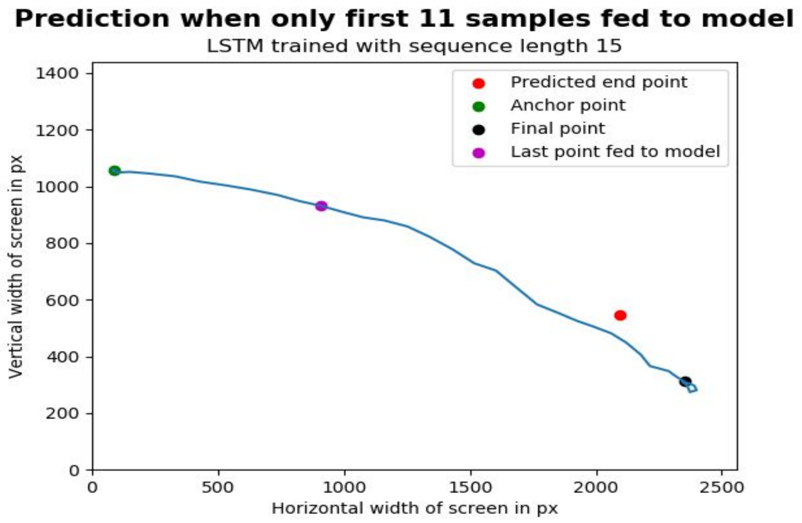




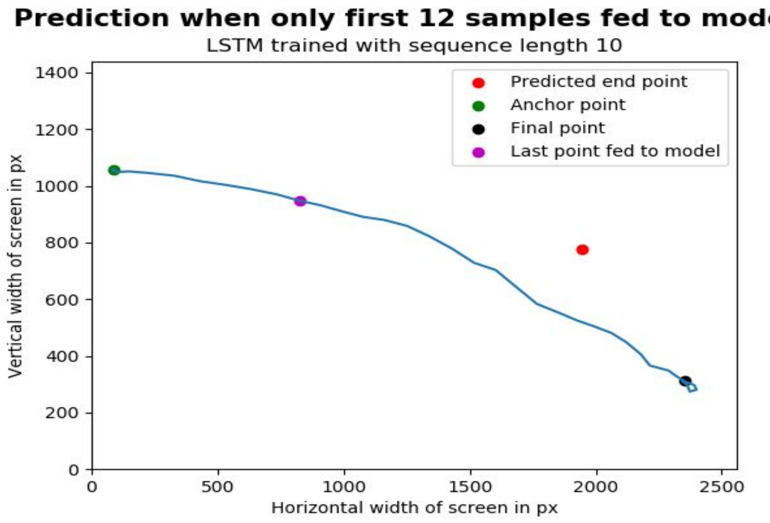
Visualizations: Early prediction performance – First 11 samples

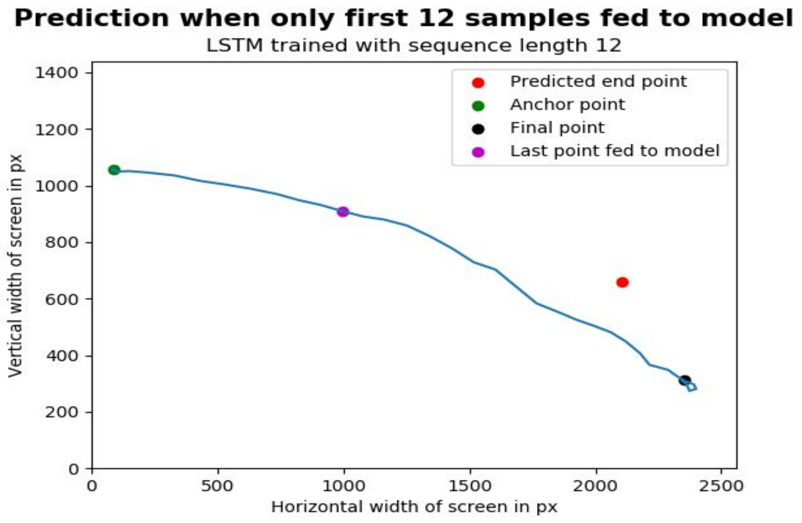


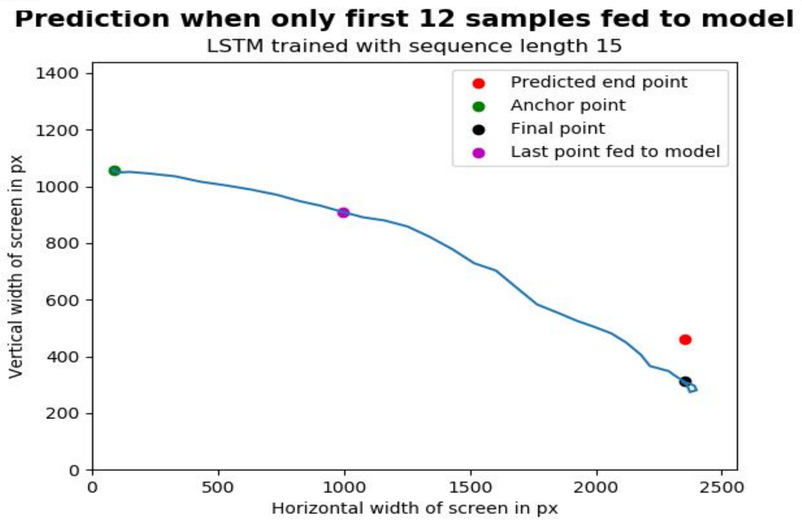




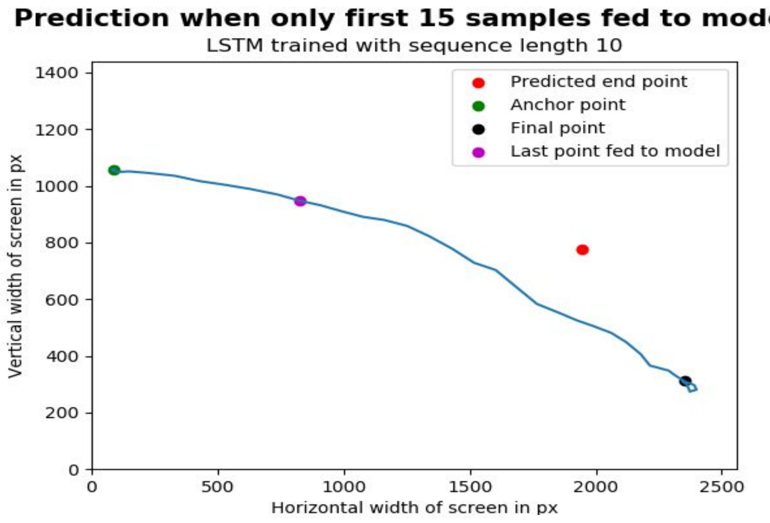
Visualizations: Early prediction performance – First 12 samples

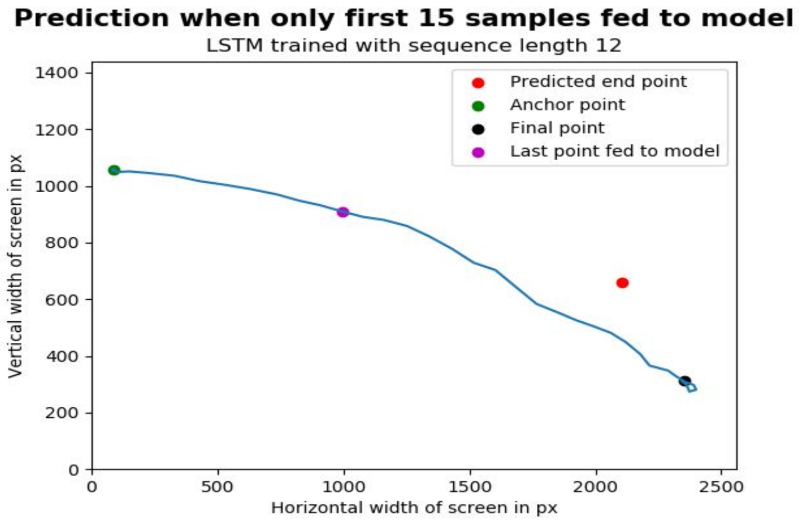


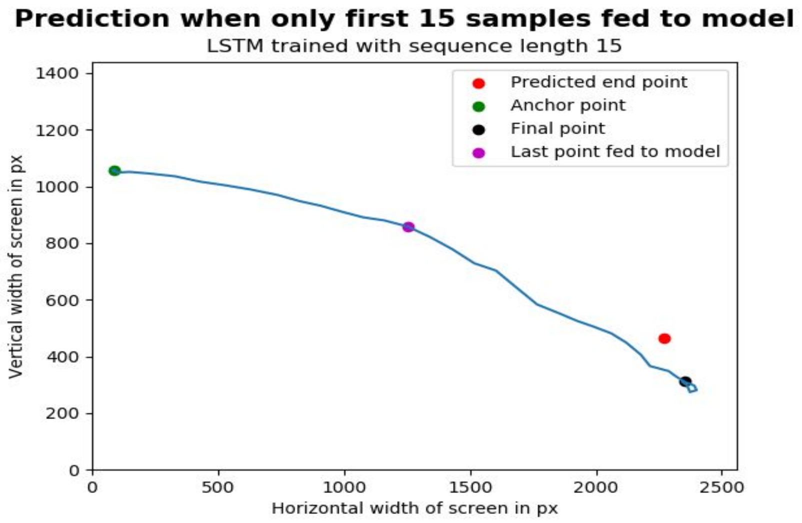




Visualizations: Early prediction performance – First 15 samples





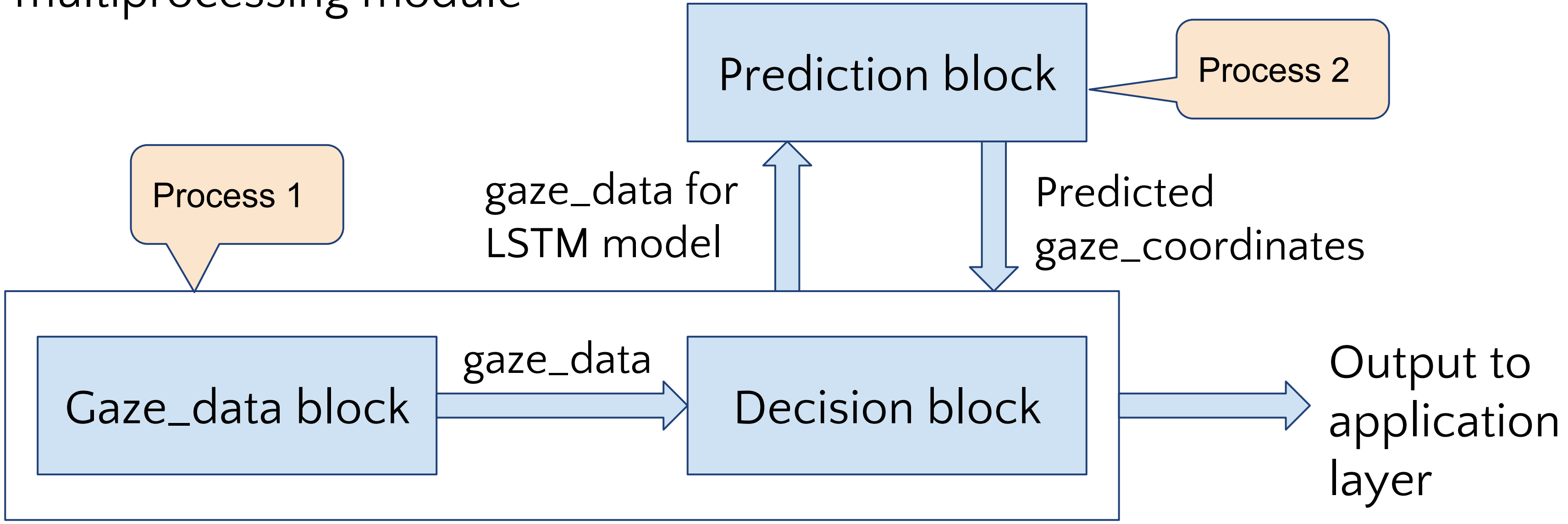


Tobii EyeX Feasibility Study

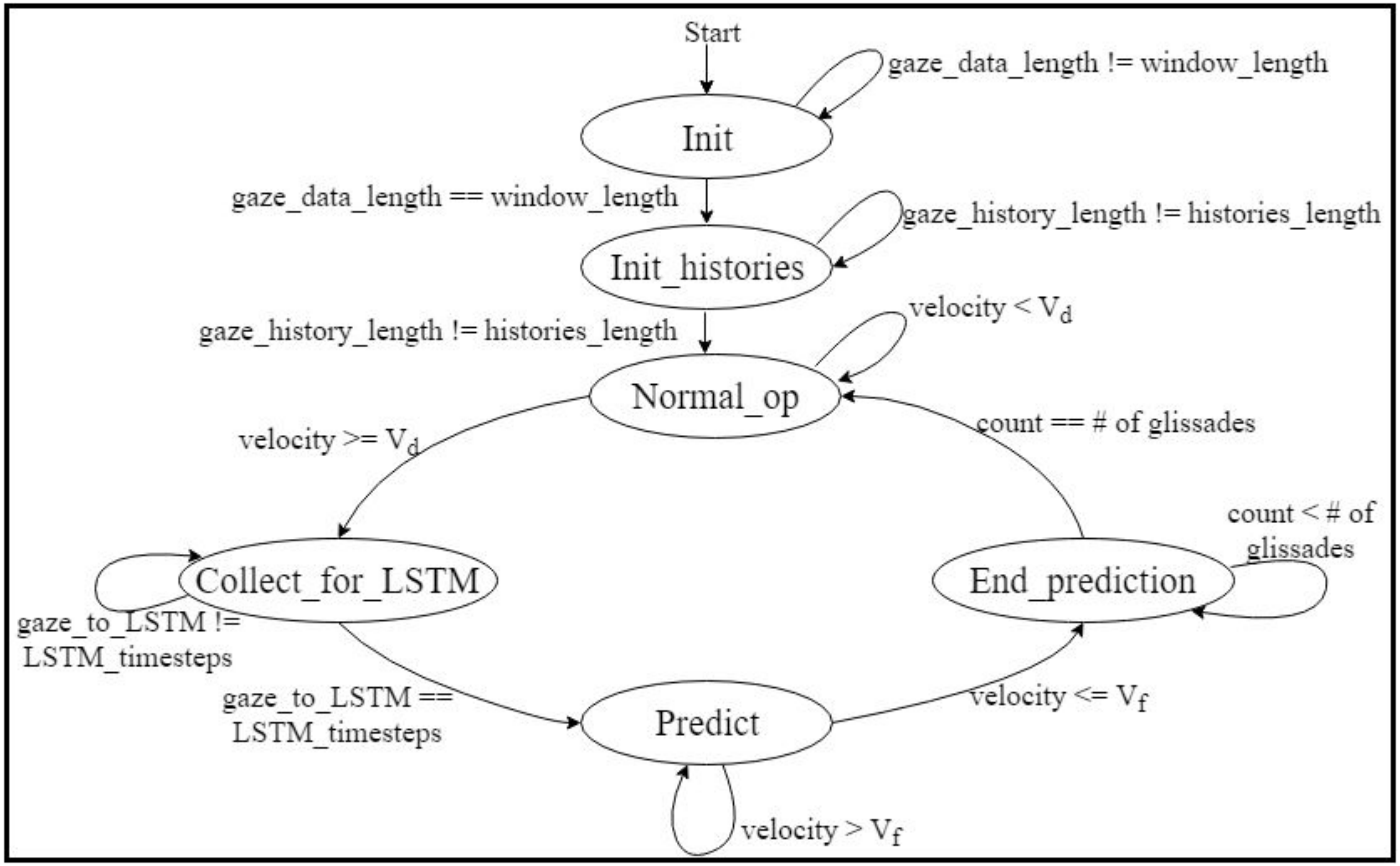
- Built C# application to stream gaze data as EyeX doesn't support Tobii pro sdk
- Sampling frequency: 65Hz
- Number of samples per saccade:
  - Average: 6 samples
  - Maximum: 13 samples
- Bad prediction as average number of samples in a saccade are low
  - Prediction good only for longer saccades

System design(1)

- Two processes run in parallel to achieve real time behaviour
  - Process 1 handles gaze data and decision block
  - Process 2 is active only when prediction is needed, runs in a 1ms raster
- Message sharing via “shared memory” functionality in python multiprocessing module



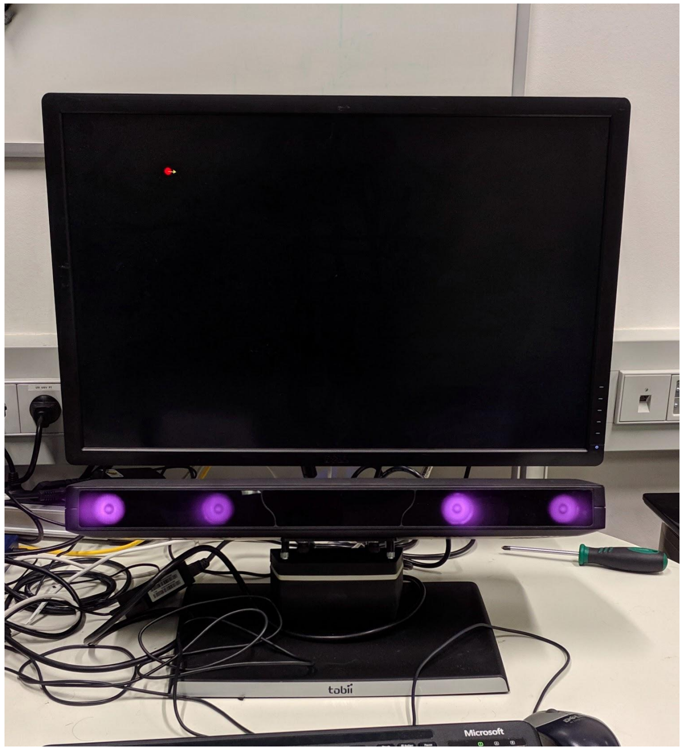
System design(2): Decision block



Start
gaze_data_length != window_length
Init
gaze_data_length == window_length
gaze_history_length != histories_length
Init_histories
velocity < V$_d$
gaze_history_length != histories_length
Normal_op
count == # of glissades
velocity >= V$_d$
count < # of glissades
Collect_for_LSTM
End_prediction
gaze_to_LSTM != LSTM_timesteps
gaze_to_LSTM == LSTM_timesteps
velocity <= V$_f$
Predict
velocity > V$_f$

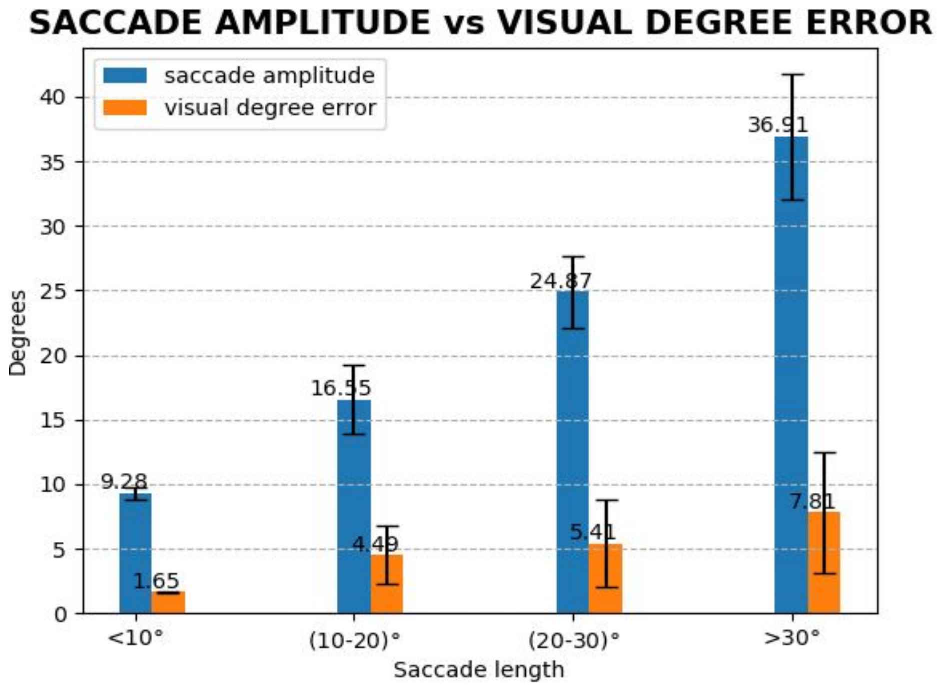
Test setup: Tobii TX300

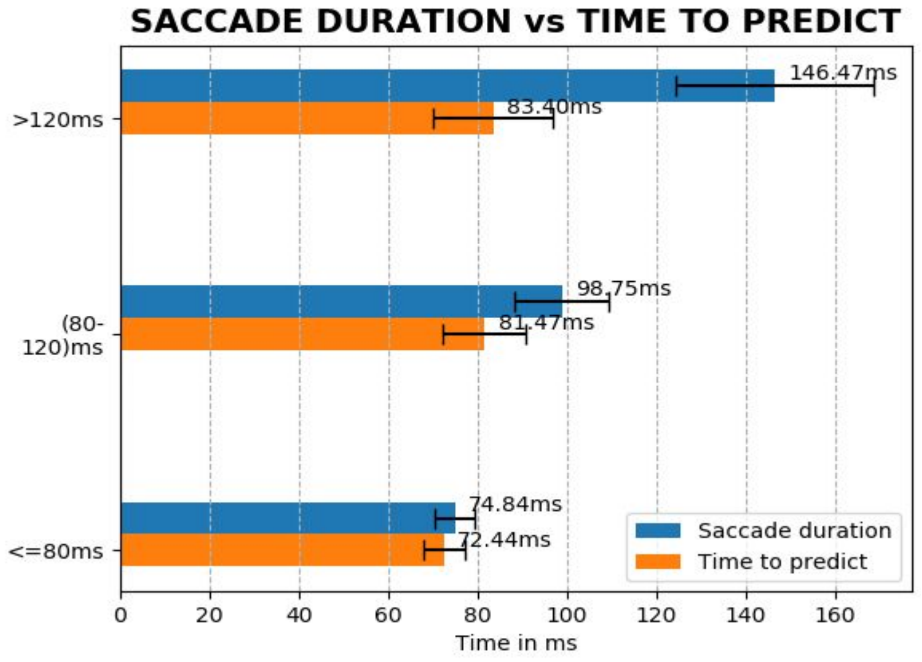
- Test Application using Pygame
  - Red dot appears at random positions on the screen
  - Arrows to indicate the direction of the next point
- Tobii TX300 eye tracker
  - Sampling frequency: 300Hz
- Display: Dell U3014, 2560x1600
- User at a distance of 65cm from the monitor, without chin rest
- Collected gaze data for 60 seconds by running pygame application



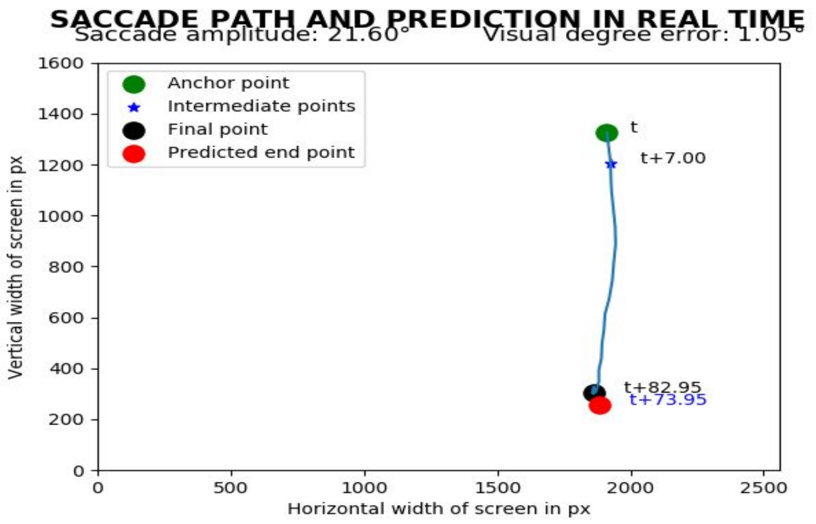
Evaluation: Real-Time Scenario

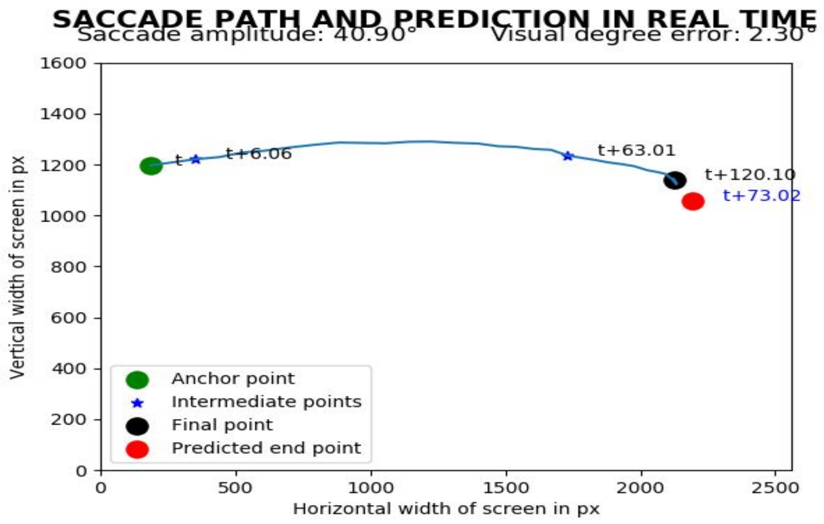
- System is robust to handle blinks
- Evaluation study performed with 6 users: Two sessions of 60 secs each
- Total number of saccades recorded and predicted : **267**

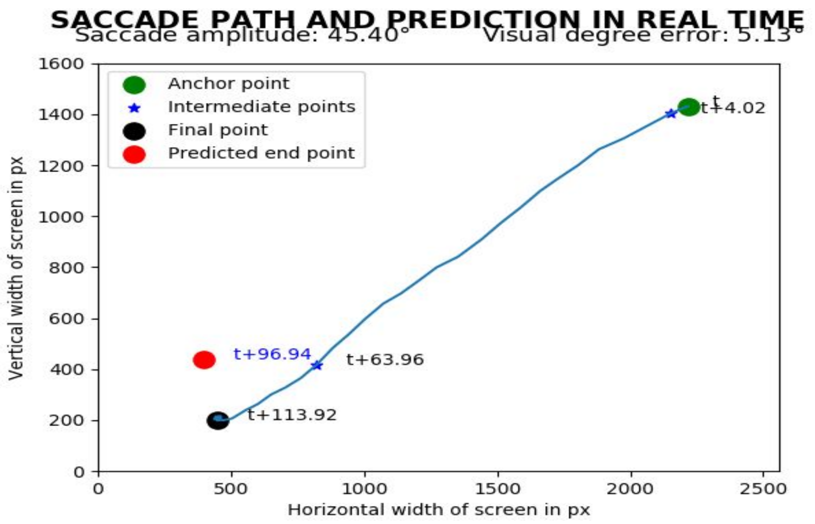




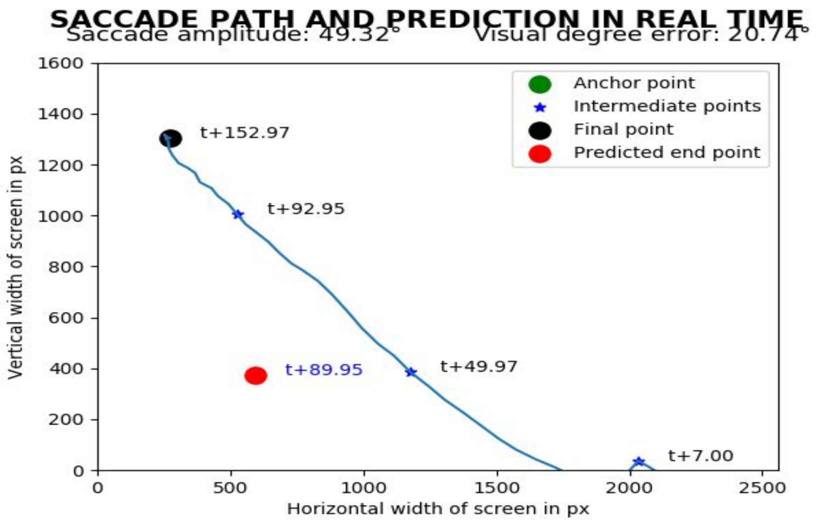
Real-Time Scenario: Best case

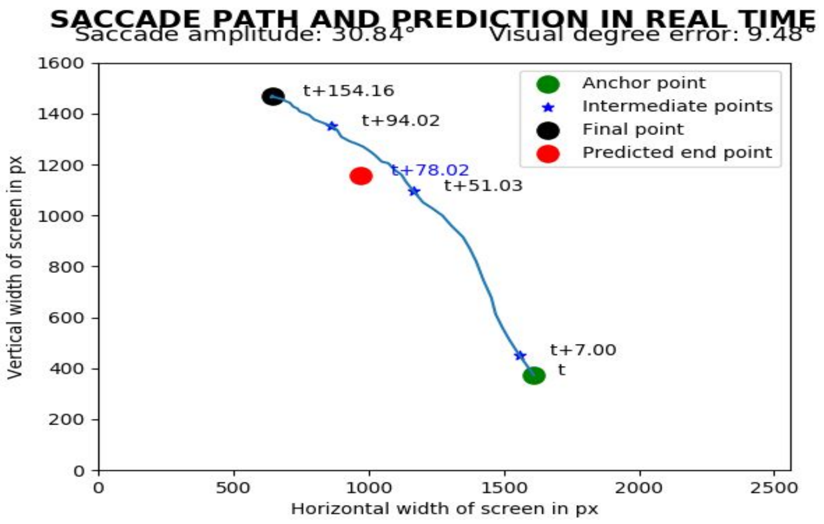


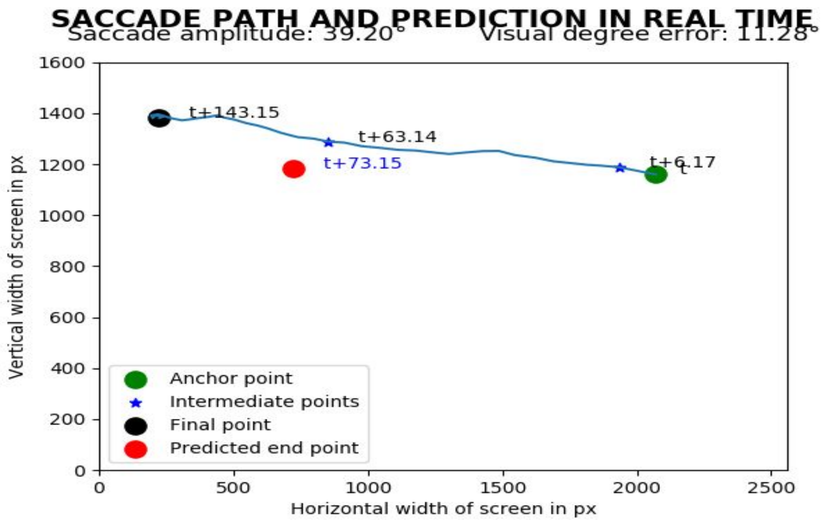




Real-Time Scenario: Worst case







Real-Time Scenario: Issues

- Each Keras prediction takes approximately 30-40 ms on the current PC setup
- System not fast enough to predict on time (before end of saccade)
- Happened in 52.66% of all the cases during evaluation study
  - Overall out of 564 detected saccades, only 267 were predicted on time
- These correspond to smaller saccades with mean amplitude of 13.22°
- Model behaviour when gaze is at screen extremities is uncertain

Summary and future work

- Improving model performance by collecting data with current setup and training with it
- Designing a model to give confidence interval of predictions
- Improving system design to handle gaze at screen extremities
- Improve system design to discard false positives
  - Abrupt change in sensor reading due to noise which gets detected as saccade

References

1. Arabadzhiyska, E., Tursun, O. T., Myszkowski, K., Seidel, H. P., & Didyk, P. (2017). **Saccade landing position prediction for gaze-contingent rendering**. ACM Transactions on Graphics (TOG), 36(4), 50
2. Morales, A., Costela, F. M., Tolosana, R., & Woods, R. L. (2018, May). **Saccade Landing Point Prediction: A Novel Approach based on Recurrent Neural Networks**. In Proceedings of the 2018 International Conference on Machine Learning Technologies (pp. 1-5). ACM
3. Wang, S., Woods, R. L., Costela, F. M., & Luo, G. (2017). **Dynamic gaze-position prediction of saccadic eye movements using a Taylor series**. Journal of vision, 17(14), 3.
4. Judd, T., Ehinger, K., Durand, F., & Torralba, A. (2009, September). **Learning to predict where humans look**. In Computer Vision, 2009 IEEE 12th international conference on(pp. 2106-2113). IEEE.
5. Dario D. Salvucci and Joseph H. Goldberg. 2000. **Identifying fixations and saccades in eye-tracking protocols.** In Proceedings of the 2000 symposium on Eye tracking research & applications (ETRA '00). ACM, New York, NY, USA, 71-78.

THANK YOU