**TECHNICAL REPORT**

**Resolving Missing Frames in Human Trajectories by Developing Reconstruction and Imputation LSTM Model**

**Overview:**

**1)Introduction and Objectives – (Page-1)**

**2)Project Activities-Page-(2-4)**

**3)Results-(Page-5)**

**4)Plots-(Page-6)**

**5)Positional Noise , velocity and Orientation distributions (Complete Trajectory) (7-17)**

**6) )Positional Noise, velocity and Orientation distributions (Ground truth Trajectory) (18-27)**

**7) Positional Noise, velocity and Orientation distributions (interpolated Region) (Page-(28-33)**

**8)comparison of Velocity for three sections(Page-34)**

**9)Comparison of Positional noise for three sections(Page-(35-40)**

**10)Conclusion and reference (Page-41)**

**Submitted by: Ramavath Suresh A logo with red lines and black text

Description automatically generated**

**Resolving Missing Frames in Human Trajectories by Developing Reconstruction and Imputation LSTM Model**

**1. Introduction:**

Human trajectories, which depict the paths individuals or objects take over time, serve as a fundamental building block for understanding and analyzing various real-world scenarios. These trajectories find applications in diverse fields such as surveillance, where tracking the movement of individuals within a monitored area aids in ensuring security. Additionally, in robotics and autonomous vehicles, accurately predicting the trajectories of moving objects is vital for effective navigation and collision avoidance.

However, the fidelity of trajectory representation is often compromised by the presence of missing frames within the trajectory data. This issue stems from a multitude of sources, including but not limited to object occlusion, sensor limitations, detection errors, and even transient sensor failures. When a trajectory contains gaps due to these missing frames, it can lead to distorted and inaccurate insights during subsequent analysis.

**2. Objectives:**

The primary objective of this project is to create a robust and accurate solution for reconstructing and imputing missing frames in human trajectories.

**The specific goals include:**

* Designing an LSTM-based architecture capable of capturing temporal dependencies in trajectory data.
* Implementing an attention mechanism to focus on relevant information during trajectory reconstruction.
* Developing a model capable of handling varying degrees of missing frames
* Evaluating the performance of the proposed model against existing methods using appropriate metrics.

**Project Activities: Data Preprocessing and LSTM-based Model Development\*\***

**Data Loading and Preprocessing**:

The initial stage of the project involved collecting trajectory data from various CSV files stored within a designated folder path. Each CSV file represented a trajectory clip, and data was organized based on a 'PID' (Person ID) column. Rigorous data validation and cleaning processes were applied to ensure the reliability and quality of the dataset.

**Trajectory Data Interpolation:**

Addressing missing data was a crucial step. Interpolation techniques were employed to fill in gaps within trajectory sequences. Missing frame IDs were assigned temporary values for the neck (x, y) coordinates. Additionally, feature engineering was applied to enrich the data by calculating velocities and orientation angles.

**Trajectory Windowing:**

For effective model training, the interpolated trajectory data was divided into windows, each containing 31 frames (approximately 5 seconds of data). This segmentation facilitated the creation of training and testing samples for the LSTM with attention model.

**Data Normalization:**

To ensure consistent learning, the input trajectory data underwent normalization using the Standard Scaler technique. This step improved model convergence by addressing feature scale discrepancies.

**LSTM with Attention Model Construction**:

The core of the project involved building an LSTM with an attention mechanism. The architecture featured an encoder-decoder structure. LSTM cells captured temporal dependencies, while the attention mechanism focused on significant trajectory segments to enhance reconstruction accuracy.

**Model Training and Trajectory Predictions:**

Following architecture development, the LSTM model was trained using the Mean Squared Error (MSE) loss function and the Adam optimizer. Training utilized the training dataset over 500 epochs, with batch sizes of 256 samples. The trained model was then ready for trajectory reconstruction and imputation tasks.

**Rescaling Predictions, Evaluation, and Visualization:**

Predicted trajectory results were rescaled to their original units for accurate interpretation. The model's performance was evaluated through comprehensive comparisons with actual trajectory points. Visualization techniques aided in assessing the alignment between predicted and actual trajectories, contributing valuable insights for further analysis and validation.

**LSTM with Attention Model Construction: Core of Trajectory Reconstruction and Imputation**

The fundamental aspect driving the success of trajectory reconstruction and imputation within this project lies in the implementation of the LSTM with attention model. This sophisticated model serves as the linchpin, orchestrating the intricate process of restoring missing trajectory data while taking into account the temporal dependencies inherent in the dataset.

**Encoder:**

At the foundation of the LSTM with attention model stands the Encoder. This critical component is responsible for ingesting the trajectory windows as input and subjecting them to the processing power of an LSTM layer equipped with 256 units. What sets this layer apart is its ability to furnish the model with the complete sequence of hidden states. This feature is instrumental in allowing the model to decipher the intricate web of temporal dependencies woven into the trajectory data.

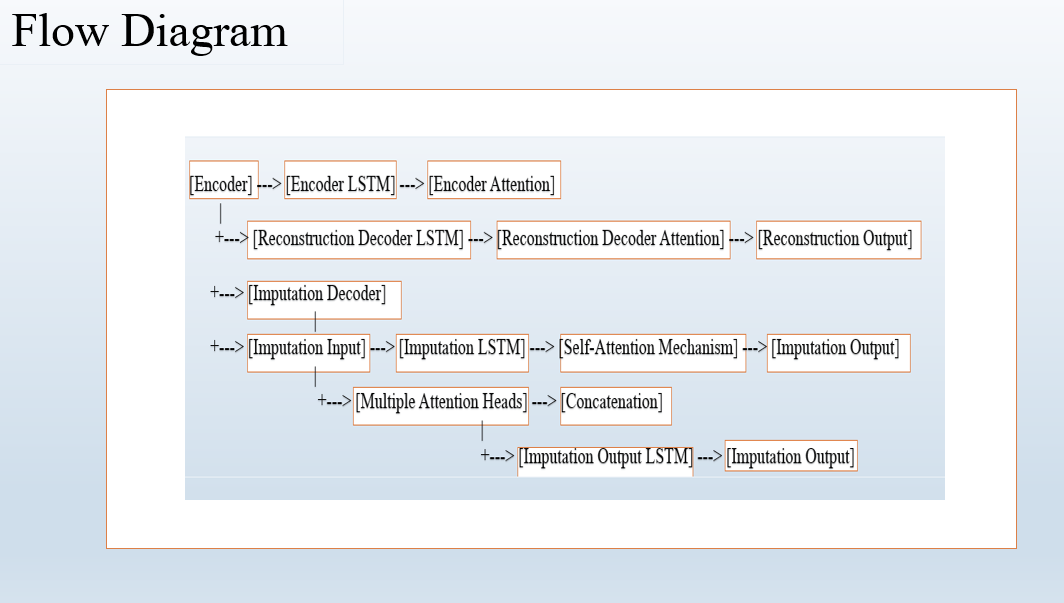
**Reconstruction Decoder:**

The role of the Reconstruction Decoder is to undertake the task of piecing together the entirety of the trajectory, including those points that had been interpolated to bridge the gaps. The process unfolds by first taking the encoded representation of the input data, a gift from the Encoder. Subsequently, the data is further processed through another LSTM layer, again comprising 256 units. However, what truly distinguishes the Reconstruction Decoder is the seamless integration of an attention mechanism. This mechanism adds an extra layer of refinement, enabling the model to focus its attention on the most pertinent aspects of the trajectory data during the reconstruction phase.

**Imputation Decoder with Self-Attention:**

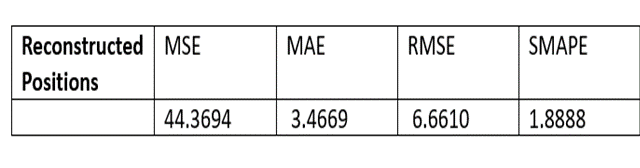
While the Reconstruction Decoder excels at its role, the Imputation Decoder enters the scene with a unique approach. Much like its counterpart, the Reconstruction Decoder, the Imputation Decoder aspires to recreate the trajectory while meticulously accounting for the relationships between observed and interpolated points. However, the Imputation Decoder deploys a self-attention mechanism—an innovation that introduces a new dimension of effectiveness.

The self-attention mechanism within the Imputation Decoder empowers the model to explore dependencies intrinsic to the trajectory sequence itself. It seamlessly considers both observed and interpolated points, effectively assigning weights to each point's significance within the sequence. This adaptive weighting mechanism lends the model an elevated comprehension of the trajectory's temporal structure, significantly enhancing its capacity to predict and impute missing values with remarkable accuracy.



**▪ RESULTS**

* Training Loss =0.0099
* Dense Loss: The dense loss, with a value of **0.0050**, represents the error loss specifically for the reconstruction decoder. It indicates the extent of deviation between the predicted and actual positions in the reconstructed trajectory.
* Dense Loss: The dense loss, with a value of **0.0049**, refers to the error loss for the imputation decoder. It measures the difference between the predicted and actual positions for the imputed trajectory using the self-attention mechanism.
* The evaluation of the model on the reconstructed and imputedpositions yielded the following results:

A close-up of a number

Description automatically generated

**Plots:**

A graph of a graph with a line graph

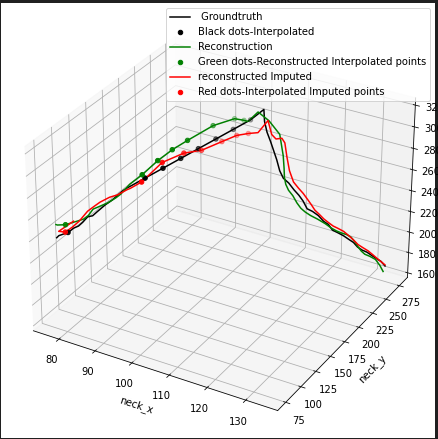
Description automatically generated with medium confidence

Fig 1 is Ground-truth with Gaps. Fig- below is the result for Fig-1. The ground truth trajectory is represented by a black line, black dots represents interpolated points. Reconstructed trajectory, obtained from the LSTM with attention model, is shown as a green line,green dots represents reconstructed interpolated points. Imputed trajectory, obtained from the LSTM with self-attention model, is shown as a red line , red dots represents imputed points from the interpolated points.

**Positional Noise, Velocity and Orientation Angles**

**Estimated for Every 5 , 1 , 0.6 Seconds for the whole Trajectory**

**Velocity In terms of Pixels/sec.**

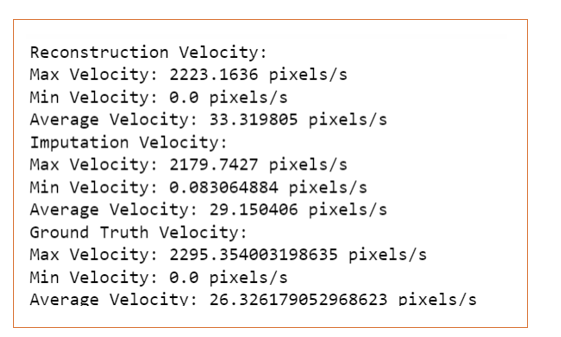
A graph with numbers and lines

Description automatically generated

A graph with green and brown bars

Description automatically generated

**Values of Velocity and Orientation Angle**

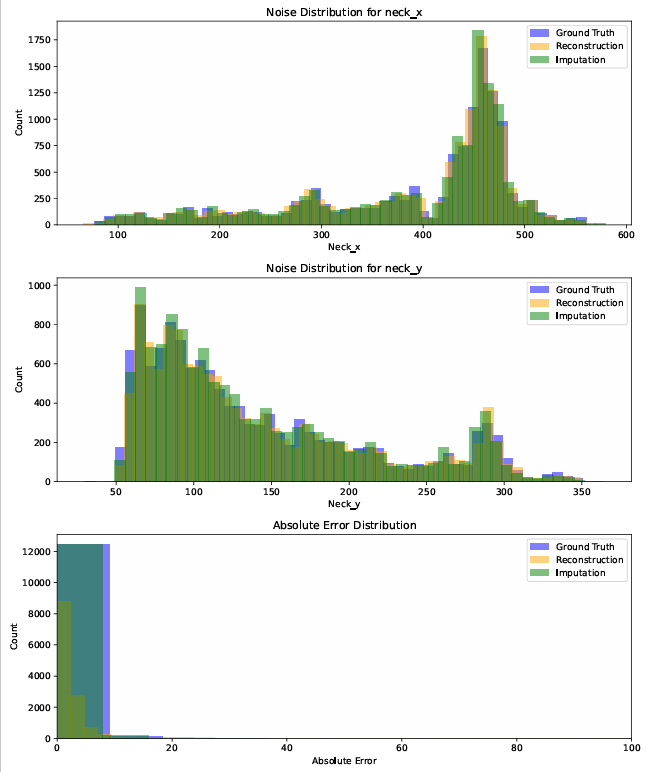


A computer screen shot of a number

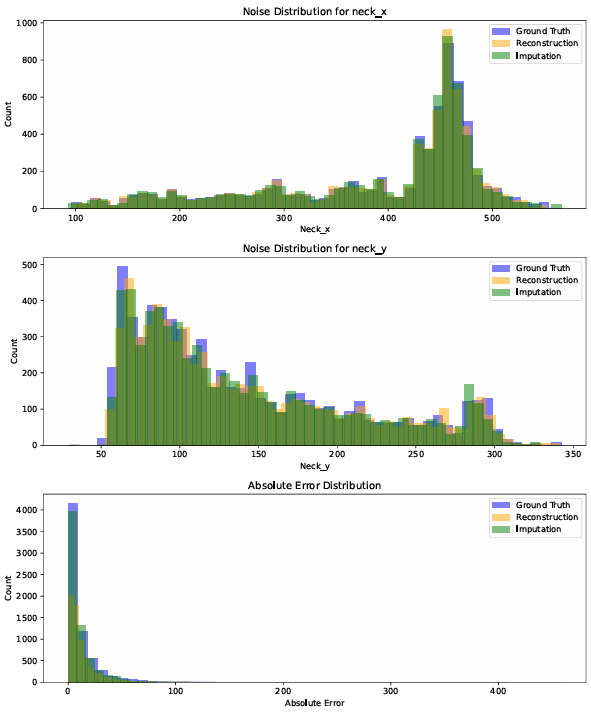
Description automatically generated

**Positional Noise (Whole Trajectory)**

**3 Points (Pixels)**



**6 Points (Pixels)**



**26 Points (Pixels)**

A graph of different sizes and colors

Description automatically generated with medium confidence

**Values of**

**Positional Noise (Min\_Max\_mean)**

**3 Points (Pixels)**

A number on a white background

Description automatically generated

**6 points**

A number on a white background

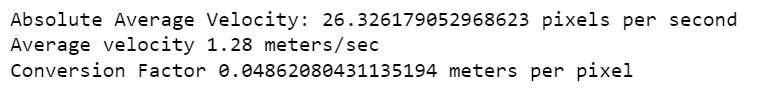
Description automatically generated

**26 Points**

A close up of numbers

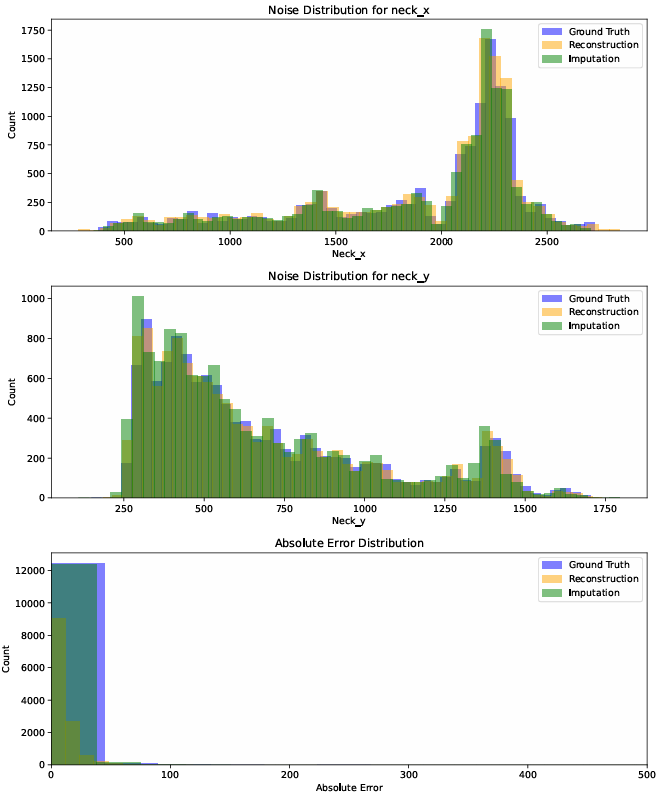
Description automatically generated

**Conversion Factor:**

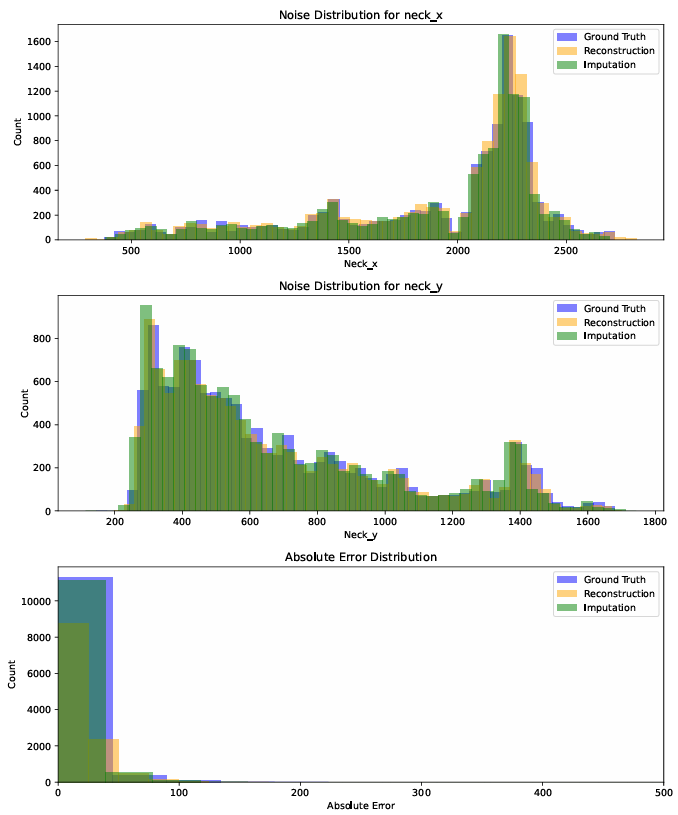


**Positional Noise in cm(Complete Trajectory)**

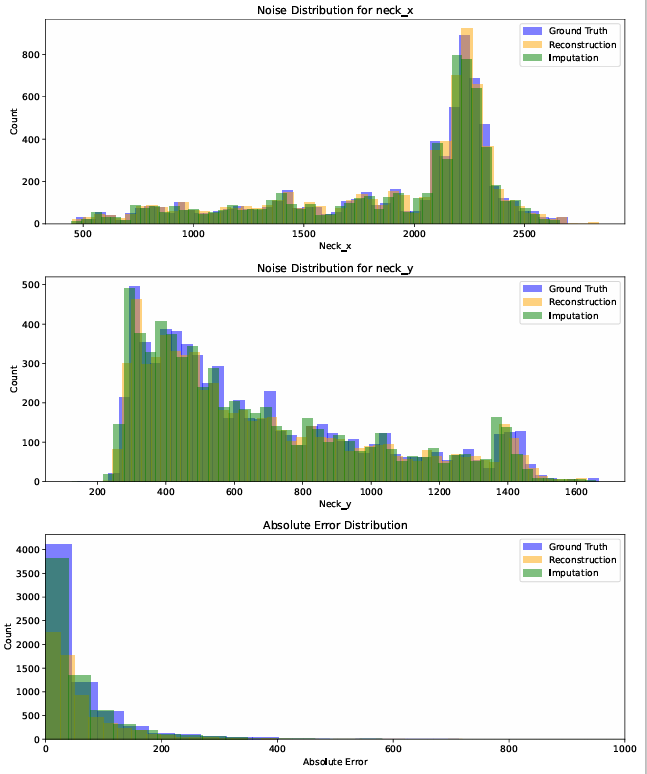
**3 points (cm)**



**6 points(cm)**



**26 Points (cm)**



**Values of positional error in cm**

**3 points (sec)**

A number on a white background

Description automatically generated

**6 points (sec)**

A number on a white background

Description automatically generated

**26 Points (sec)**

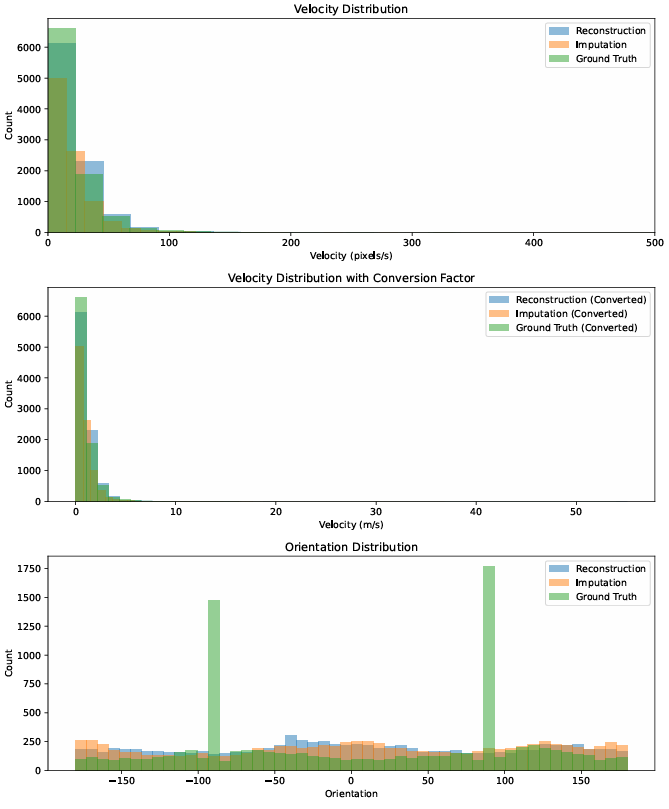
A white background with numbers

Description automatically generated

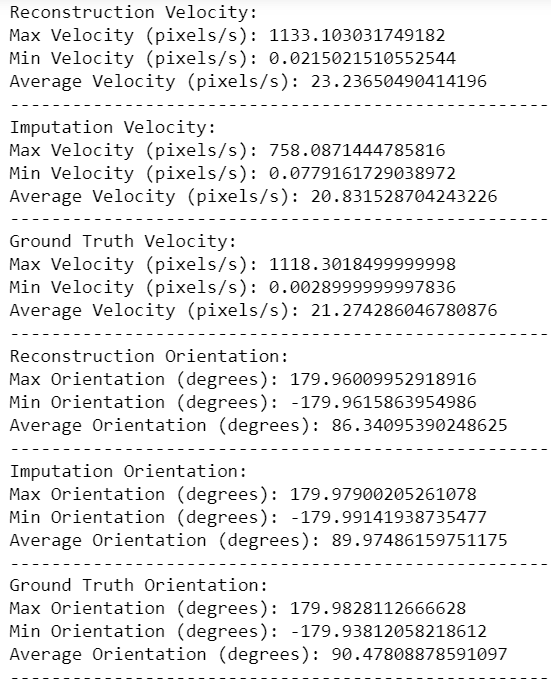
**Ground truth Trajectory**

**Position Noise and Velocity Distribution for Ground truth (Observed Trajectory)**

**Velocity Distributions In terms of Pixels/sec.**

****

**Values of Velocity (pixels/sec) and orientation(Degrees) distribution**

****

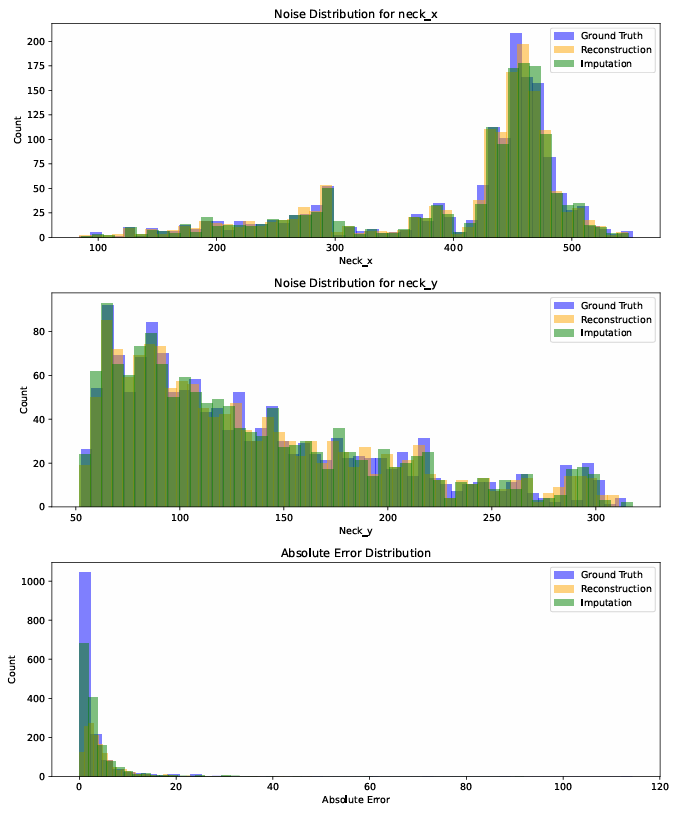
**Positional Noise for Ground-truth Trajectory(Pixels)**

**3 points(Pixels )**

**A screenshot of a graph

Description automatically generated**

**6 points (Pixels)**

****

**26 Points (pixels)**

**A graph of different colored bars

Description automatically generated with medium confidence**

**Positional Noise in cm (Ground truth Trajectory)**

**3 points (cm)**

**A screenshot of a graph

Description automatically generated**

**6 points (cm)**

**A screenshot of a graph

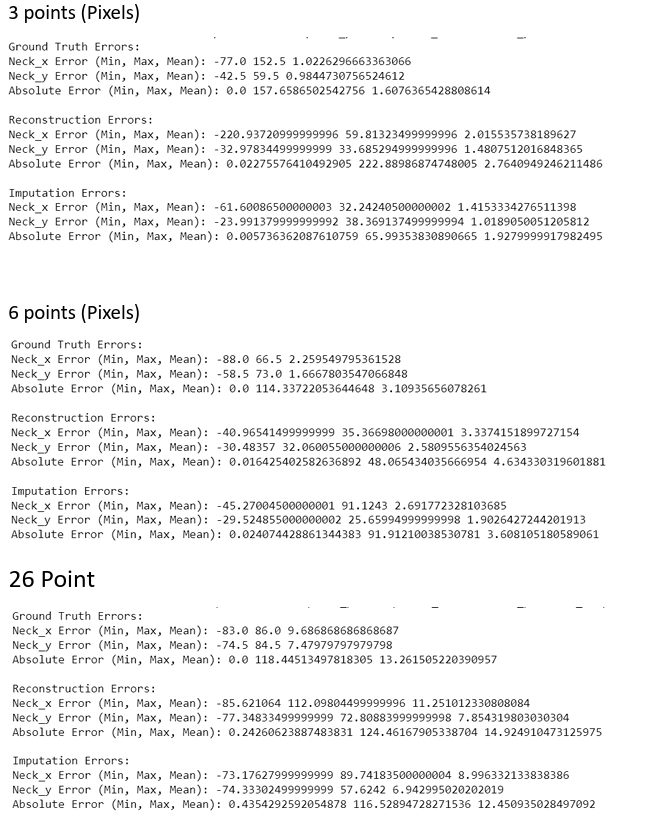
Description automatically generated**

**26 points (cm)**

**A graph of different colored bars

Description automatically generated with medium confidence**

**Values of Positional Noise in Pixels (Groundtruth Trajectroy)**

****

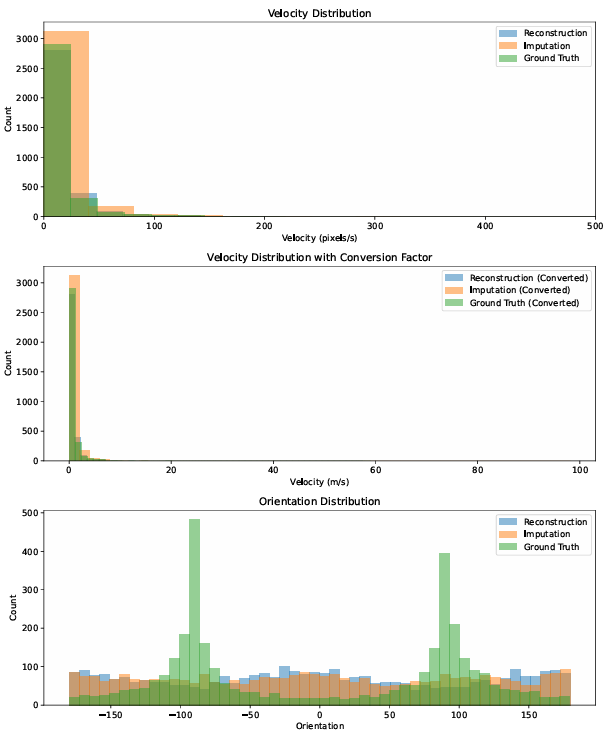
**Values of Positional Noise in cm (Groundtruth Trajectory)**

**A screenshot of a computer code

Description automatically generated**

**Interpolated Region**

**Velocity and orientation distribution for Interpolated Region**

****

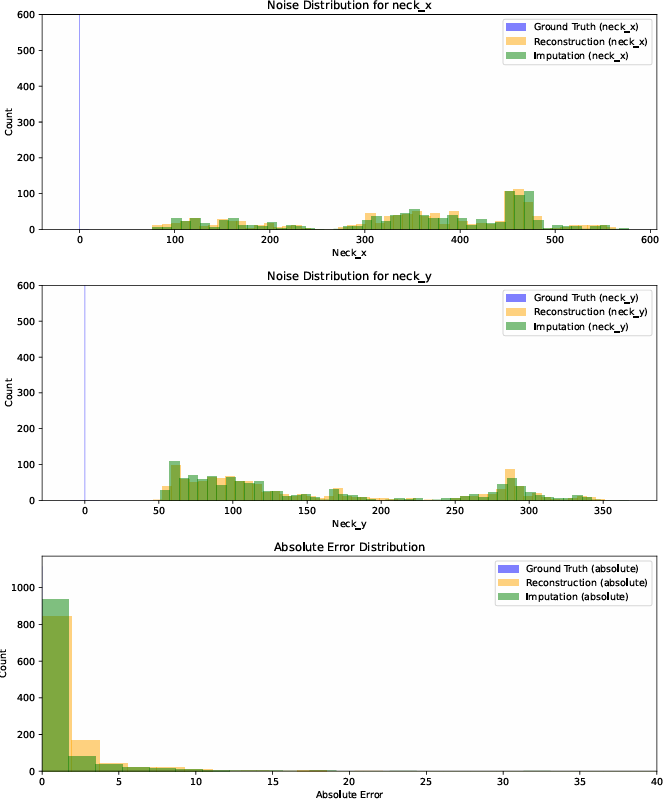
**Values of Velocity and orientation for interpolated Region**

A screenshot of a computer

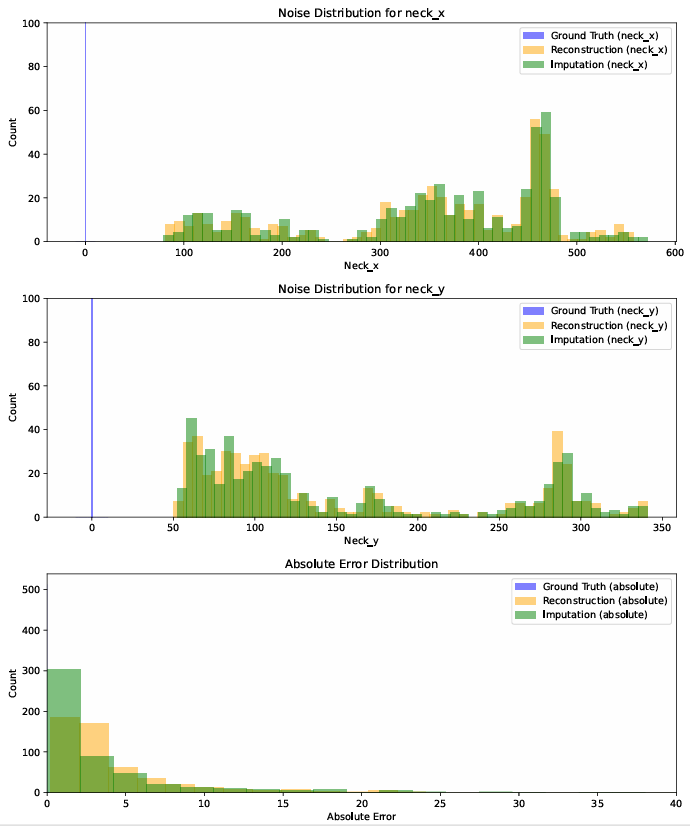
Description automatically generated

**Positional Noise for Interpolated Region**

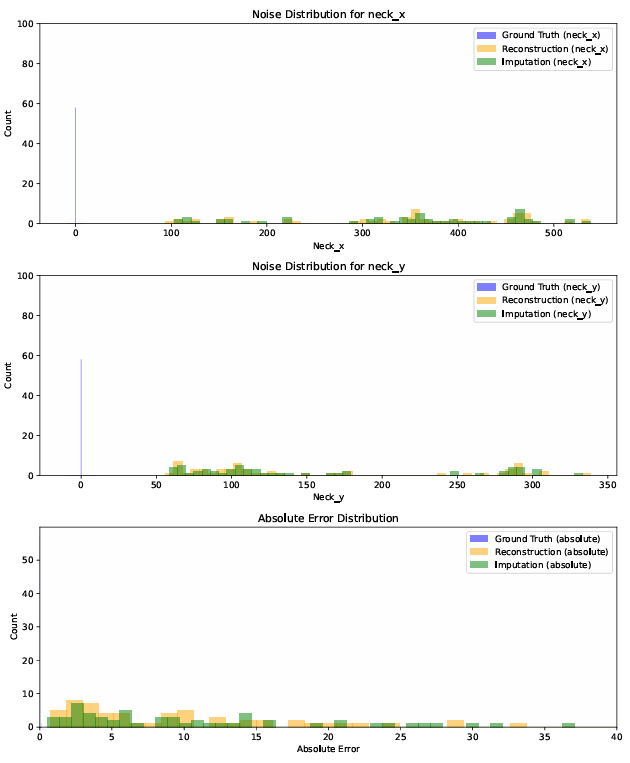
**3 points (Pixels)**

****

**6 points(Pixels)**

****

**26 points(Pixels)**

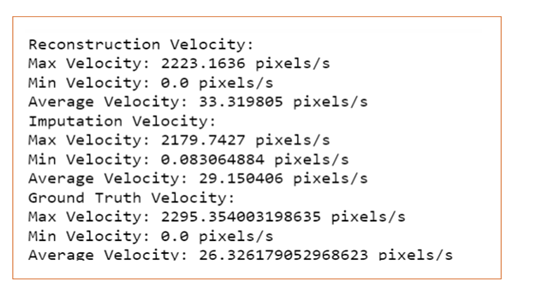
****

**Values of Positional Noise for Interpolated Region**

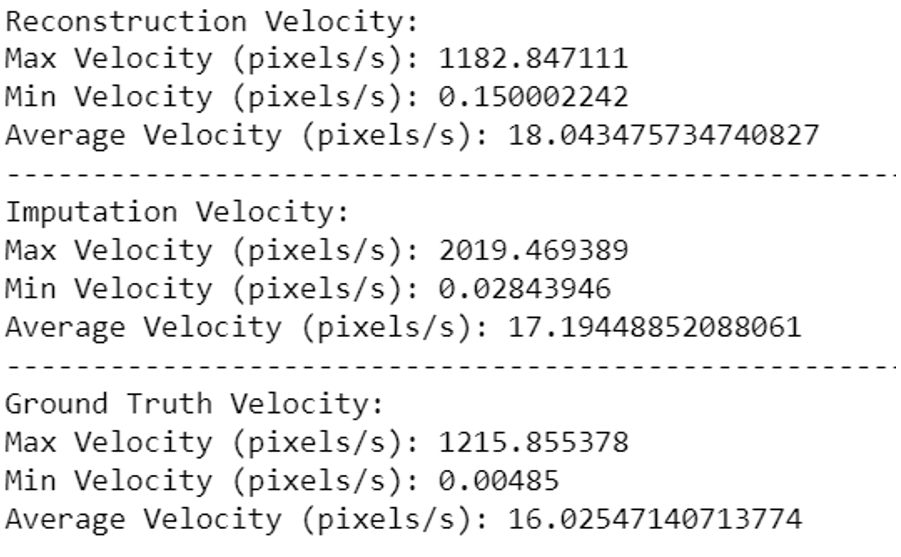
****

**Comparison of Velocity(Pixels) for three sections**

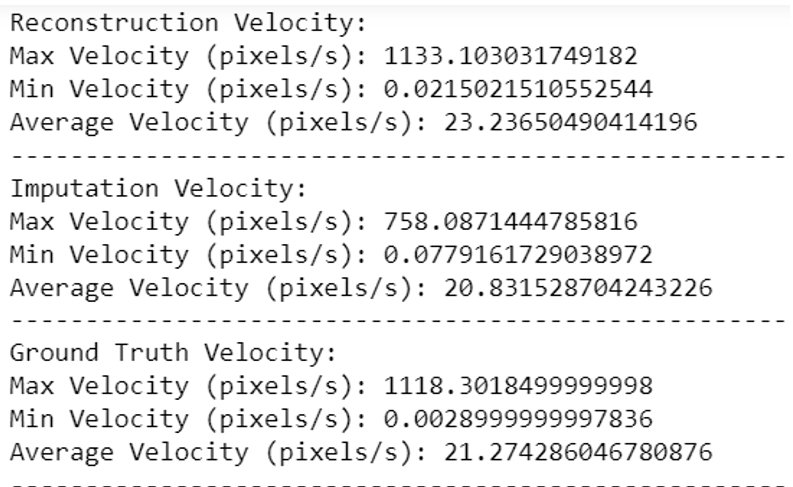
**Complete Trajectory**



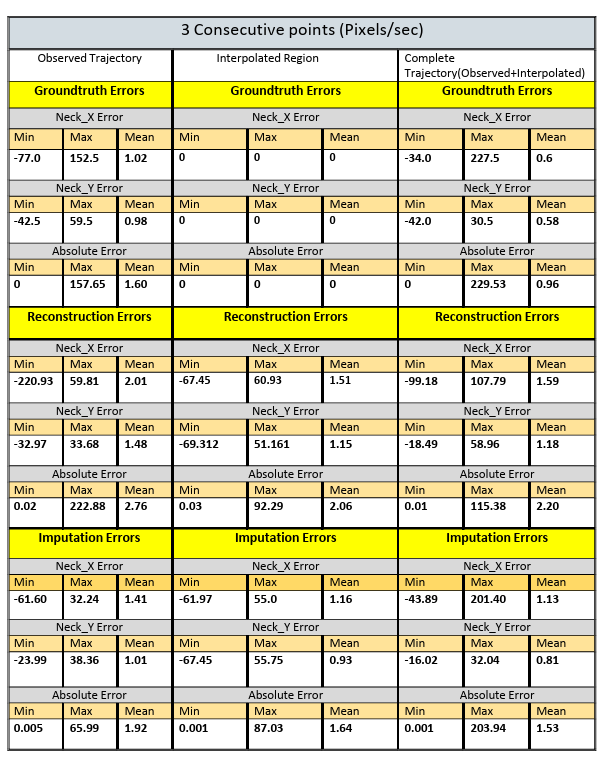
**Interpolated Region**

****

**Ground-truth Trajectory**

****

**Comparison of Positional Noise for Three sections**

****

**6 points (Pixels)**

**A table with numbers and symbols

Description automatically generated with medium confidence**

**26 Points**

**A yellow and white table with numbers and text

Description automatically generated**

**Comparison of Positional Noise in centimeters**

**A table of error codes

Description automatically generated with medium confidence**

**6 Points**

**A table with text and numbers

Description automatically generated**

**26 Points**

**A table of error codes

Description automatically generated**

**Conclusion :**

In conclusion, the LSTM with attention model has demonstrated outstanding performance in both trajectory reconstruction and imputation tasks, as evidenced by the position noise error analysis for Ground Truth, Interpolated, and Whole Trajectory data.

After analyzing the noise and velocity ranges in the reconstructed and imputed whole trajectories, we noticed that they were slightly higher compared to the ground truth. This happened because in our ground truth data, we used linear interpolation to fill in the missing parts. This interpolation process made the ground truth data smoother, which in turn resulted in slightly lower noise and velocity variations. These findings shows that the reconstructed and imputed results are good

**References:**

* **Introduction to LSTM -** [**https://www.pluralsight.com/guides/introduction-to-lstm-units-in-rnn**](https://www.pluralsight.com/guides/introduction-to-lstm-units-in-rnn)
* **Self-Attention in NLP-** [**https://www.geeksforgeeks.org/self-attention-in-nlp/**](https://www.geeksforgeeks.org/self-attention-in-nlp/)
* **Auto Encoder Decoder LSTM:** [**https://github.com/pdacorn/animal-trajectory-analysis**](https://github.com/pdacorn/animal-trajectory-analysis)
* [**https://peerj.com/articles/cs-656/**](https://peerj.com/articles/cs-656/)