IMU odometry using deep learning

[1]Hüseyin Kıvanç Oral, [1]Alperen Oğuz Demir, [1,*]Erkan Uslu
[1]Yildiz Technical University, Faculty of Electrical and Electronics, Department of Computer Engineering
*euslu@yildiz.edu.tr

Abstract: IMU is a measurement unit that usually consists of sensors such as accelerometers, gyroscopes and magnetometers. Odometry is the estimation of incremental or absolute position and orientation of the device by processing sensor data over time. This paper focuses on how deep learning, a subtopic of artificial intelligence, can be used for IMU odometry. IMU odometry is solely the use of data from IMU sensors for position and orientation estimation. IMU sensor measurements are subject to many errors and noise, and therefore the measurements obtained may not reflect the true values. Our study filters these measurements with deep learning algorithms to provide a more consistent estimate. Our study uses recurrent neural networks such as LSTM, GRU and a hybrid CNN-LSTM network to estimate the position of a micro aerial vehicle using IMU data and compares the results with the ground-truth data by means of defined metrics. EuRoC MAV dataset is used for this purpose. Our study also aims to predict orientation change in terms of rotation angle using only IMU data utilizing a deep learning model. While GRU model performed the best on position estimation by an R^2 value of 0,996, orientation estimation reached an R^2 value of 0,56. The promising results enable the use of the proposed method as an intermediary position estimation on high level SLAM algorithms as a future work.
Keywords: IMU Odometry, Deep Learning, LSTM, GRU, CNN-LSTM.

1. Introduction

Odometry or dead reckoning is used to estimate the position and movement of devices such as robots, autonomous vehicles, drones and virtual reality devices, often using data from different sensors such as accelerometers, gyroscopes and magnetometers.

IMU (Inertial Measurement Unit) odometry is used to accurately track the position and movement of devices in areas where GPS signals are insufficient or unavailable. For example, when a drone is flying outdoors and GPS signals are weak or completely lost, it can still be flown accurately using IMU odometry.

IMU odometry can also provide positioning and navigation systems with higher localization and orientation accuracy, improving the efficiency of the devices and helping them to perform their tasks more accurately.

Deep learning, a sub-branch of artificial intelligence, consists of artificial neural networks, similar to the way the human brain works. These neural networks pass input data through many layers, learning complex relationships between the data and ultimately producing an output. The processing steps between these layers are performed by performing mathematical operations with the inputs of each layer, using weights and bias values. As a result of these operations, the network produces an output that matches the given input.

Since the data from IMU sensors can have many errors and noise, it is difficult to obtain accurate output data and to facilitate this process, it is necessary to filter out these many errors and noise. Deep learning algorithms can be used for this filtering process and thus a more accurate result can be obtained.

Deep learning, as mentioned above, learns from data using a structure of artificial neural networks and can perform particularly well on large data sets. Therefore, using data from IMU sensors, deep learning algorithms can more accurately estimate the position and motion of the device by filtering out errors and noise in the sensor data.

Position and orientation estimation from IMU sensor is firstly defined in strapdown inertial navigation system (SINS) model (Titterton, et al., 2004) where double integration on linear accelerations and integration on angular velocities are carried out. However since accumulated error makes the estimation unstable, pedestrian dead reckoning (PDR) model is suggested with zero velocity constraint that enables eliminating accumulated error periodically (Lu, et al., 2019).

A local positioning system that uses 2D dead reckoning principle is proposed in (Zheng, et al., 2023). The system estimates real-time pose information using MTI Silicon Micro Inertial Measurement System. The proposed system uses Kalman filter on speed-acceleration model to reduce error of the odometer and uses a linear error model to minimize the accumulative error of the gyroscope. The cumulative error of the results obtained is within 2 meters in 100 meters distance interval.

An IMU dead reckoning method that filters noise data and determines the position of the vehicle accurately is proposed in (Toy, et al., 2022). The experimental results the article obtains show the proposed system is effective with rotational and translational errors of 1.03 degrees and 1.04 meters.

A hybrid visual-inertial odometry solution for autonomous navigation of unmanned vehicles in GPS-denied environments is proposed in (Mahmoud & Atia, 2019). The proposed system loosely couples integration between stereo cameras and IMU that is filtered by an Extended Kalman Filter (EKF) to estimate IMU biases. Then the system is downgraded into a single camera getting the motion scale from the calibrated IMU. The results the article obtained have less than 2m overall position estimation error.

A deep learning model based on separate input layers fed to sequential LSTM layers followed by a concatenation layer and several more sequential LSTM layers is proposed in (Abolfazli Esfahani, et al., 2020).

Another deep learning model based on CNN layers followed by two bidirectional LSTM layers and a fully connected last layer is proposed in (Silva do Monte Lima, et al., 2019).The network is trained in a multi-task learning framework where two distinct loss functions for position and orientation is fused together.

One of the most referenced 3D SLAM methods, LIO-SAM, uses IMU pre-integration step as an initial guess for LiDAR feature matching. It is reported that when the IMU pre-integration step fails to produce accurate position estimates, LIO-SAM cannot reach real time conditions or fails too (Shan, et al., 2020).

The rest of the study is organized as follows: Section 2 introduces the methods used in position and orientation estimation and gives details about the deep learning models used, Section 3 states the experimental setup, Section 4 gives the experimental results and discussions about these results, and Section 5 summarizes the study with results and discussion and gives future work remarks.

2. Materials and Methods

This section deals with the models proposed for position and orientation estimation based on IMU data. Position estimation is carried out by several RNN architectures whereas for orientation estimation a sequential fully connected model is utilized.

2.1. Position Estimation with Time Series Forecasting

A time series is a series of time-based values in an observation. Time series forecasting is scientific forecasting using historical data. In literature RNN networks such as LSTM and GRU are used for time series forecasting, in this study aforementioned networks are utilized for IMU odometry estimation.

2.1.1. Long Short-Term Memory

Long Short-Term Memory (LSTM) is a type of recurrent neural network (RNN) architecture designed to capture and retain long-term dependencies in sequential data. It overcomes the vanishing gradient problem of traditional RNNs by introducing a memory cell and gating mechanisms to control the flow of information. LSTMs are widely used in applications such as natural language processing and time series analysis (Van Houdt, et al., 2020).

A sequential model consisting of two LSTM layers with 50 units each, followed by a dense layer with 3 units is used. The model takes input sequences of shape (10, 9) and aims to minimize the mean squared error loss using the Adam optimizer. The model summary is given with Fig. 1.

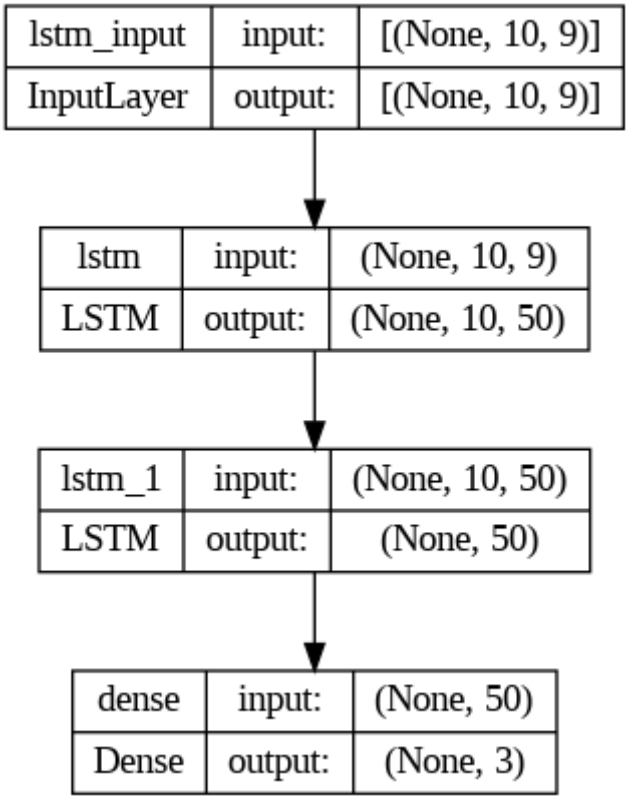
```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm (LSTM)                 (None, 10, 50)            12000

 lstm_1 (LSTM)               (None, 50)                20200

 dense (Dense)               (None, 3)                 153

=================================================================
Total params: 32,353
Trainable params: 32,353
Non-trainable params: 0
_________________________________________________________________
```

(a)



(b)

Figure 1. a) Summary, b) Visualization of LSTM Model

2.1.2. GRU

GRU (Gated Recurrent Unit) is a simplified type of recurrent neural network (RNN) architecture that addresses the vanishing gradient problem. It combines the input and forget gates of LSTM into a single update gate, making it more computationally efficient. GRUs capture long-term dependencies in sequential data and are commonly used in applications like natural language processing and speech recognition (Heck & Salem, 2017), (Niu, et al., 2023).

A sequential model with two GRU layers followed by a dense layer is used. The input shape is (10, 9), representing a sequence length of 10 time steps and 9 features per step. The first GRU layer has 50 units and returns the full sequence. The second GRU layer also has 50 units but only returns the last output. The final dense layer has 3 units. The model summary is given with Fig. 2. The model is trained using the mean squared error loss function and the Adam optimizer.

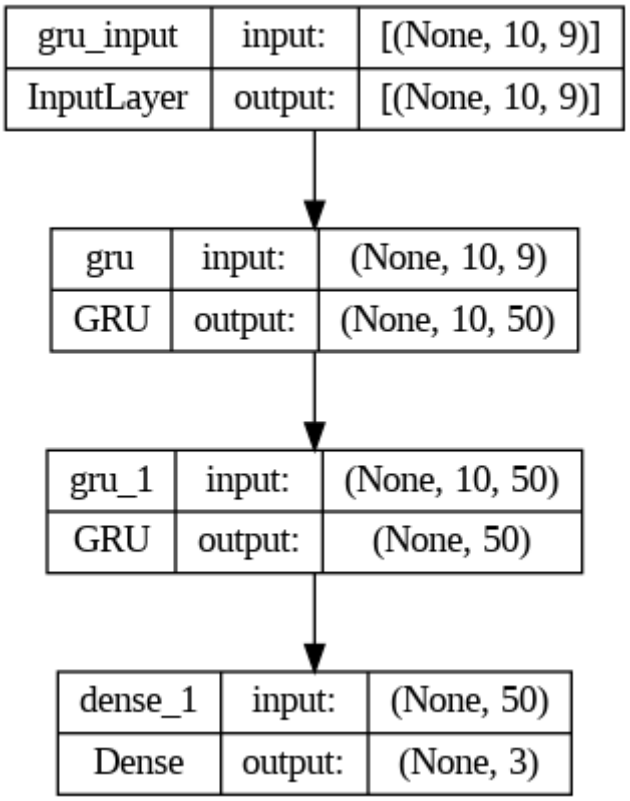
```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 gru (GRU)                   (None, 10, 50)            9150

 gru_1 (GRU)                 (None, 50)                15300

 dense_1 (Dense)             (None, 3)                 153

=================================================================
Total params: 24,603
Trainable params: 24,603
Non-trainable params: 0
_________________________________________________________________
```

(a)



(b)

Figure 2. a) Summary, b) Visualization of GRU Model

2.1.3. CNN-LSTM

CNN-LSTM is a hybrid neural network that combines Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks. It can process sequential data with spatial information, like images or videos. CNN extracts spatial features, while LSTM models temporal dependencies. This architecture is used in tasks such as video analysis and action recognition (Donahue, et al., 2016).

A sequential model that combines CNN and LSTM layers is used. It takes input sequences of shape (10, 9, 1) and applies a 1D Convolutional layer with 64 filters and

a kernel size of 3 to each time step independently. MaxPooling1D with a pool size of 2 is then applied to reduce spatial dimensionality. The output is flattened and processed by an LSTM layer with 50 units to capture temporal dependencies. Finally, a Dense layer with 3 units produces the output. The model summary is given with Fig. 3. The model is trained using mean squared error loss and the Adam optimizer.

2.2. Orientation Change Estimation with Quaternion Data

Quaternions are units that represent the spatial orientation and rotation of objects in three-dimensional space. In our dataset, we have quaternion data for w, x, y and z dimensions for each time instant. Using this data, the formula for finding the change in orientation of the object in terms of rotation angle at each time interval is given in (1) where q_t is the quaternion representation of orientation at time step t with respect to the orientation at time step 0, $\otimes$ is the Hamilton product and $\Delta\phi$ is the orientation change between time step t-1 and t.

$$\Delta\phi = 2 \cdot \cos^{-1}\left(\frac{|q_{t-1} \otimes q_t|}{\|q_{t-1}\| \cdot \|q_t\|}\right) \quad (1)$$

By applying this formula, the rotation differences between each pairwise rows were found. A deep learning network was trained with IMU data as input and orientation differences as output.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 time_distributed (TimeDistr  (None, 10, 7, 64)        256
 ibuted)

 time_distributed_1 (TimeDis  (None, 10, 3, 64)        0
 tributed)

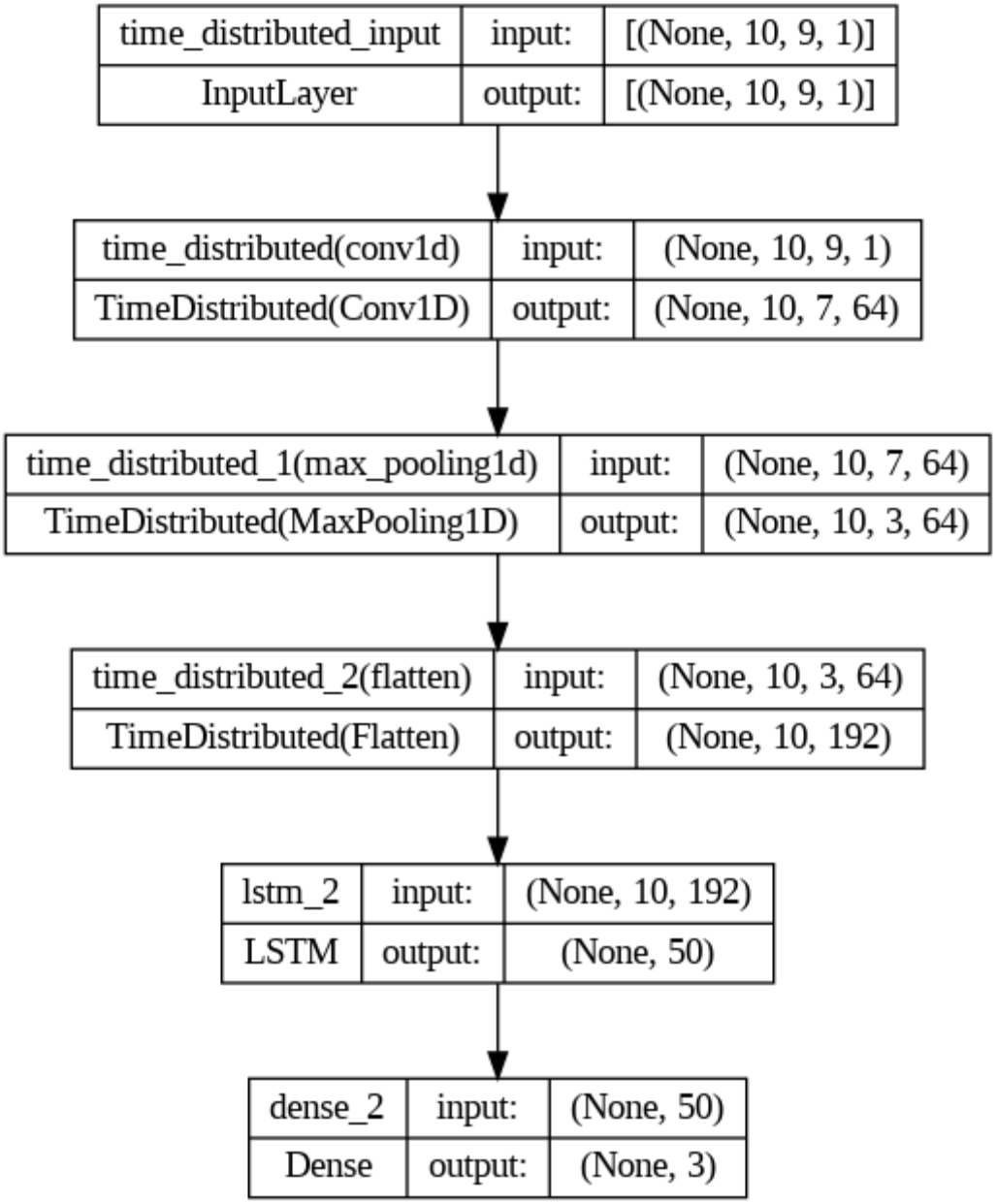
 time_distributed_2 (TimeDis  (None, 10, 192)          0
 tributed)

 lstm (LSTM)                 (None, 50)                48600

 dense (Dense)               (None, 3)                 153

=================================================================
Total params: 49,009
Trainable params: 49,009
Non-trainable params: 0
_________________________________________________________________
```

(a)



(b)

Figure 3. a) Summary, b) Visualization of CNN-LSTM Model

A sequential model with six dense layers is used to estimate the orientation difference. It starts with a layer of 128 units, followed by layers of 256, 512, 256, and 128 units, all using the ReLU activation function. The final layer has a single unit and serves as the output layer. The sequential fully connected model is given with Fig. 4. The model is trained using the Adam optimizer and the mean squared error loss function. Its goal is to minimize the MSE during training.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 128)               896

 dense_1 (Dense)             (None, 256)               33024

 dense_2 (Dense)             (None, 512)               131584

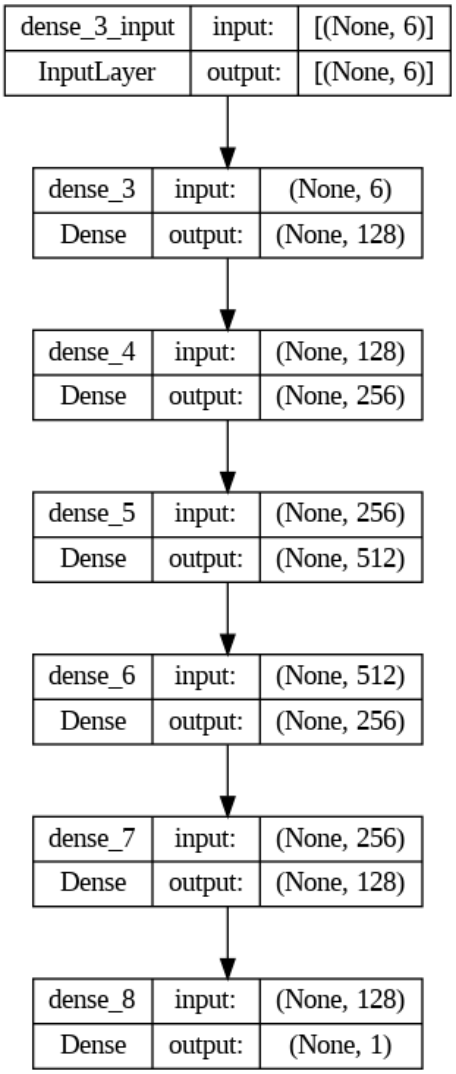
 dense_3 (Dense)             (None, 256)               131328

 dense_4 (Dense)             (None, 128)               32896

 dense_5 (Dense)             (None, 1)                 129

=================================================================
Total params: 329,857
Trainable params: 329,857
Non-trainable params: 0
_________________________________________________________________
```

(a)



(b)

Figure 4. a) Summary, b) Visualization of the Fully Connected Sequential Model

3. Experimental Setup

In this section the dataset used is introduced and the experimental setup is explained and the metrics used for comparison is defined.

3.1. Dataset

The dataset used is named "The EuRoC MAV Dataset". It presents visual-inertial datasets collected on a Micro Air Vehicle (MAV). These datasets include stereo images, synchronized 6 DoF IMU measurements (accelerometer, gyroscope), high fidelity motion and true position information. Our study uses only IMU measurements and true position information of the current frame and tries to estimate true position and orientation of the next frame from these IMU measurements. For this purpose, we use the dataset named "Machine Hall 01" from the site for supervised learning of our artificial neural network algorithm and test our network on the dataset named "Vicon Room 1 01" (Burri, et al., 2016).

3.2. Data Preprocessing

There are two different csv files in the dataset, one with ground-truth location information and one with IMU data. These two files were combined into a single table by merging the rows with common timestamps. The table has 28377 rows and 10 columns. The data was scaled with the MinMaxScaler. When the data is prepared for time series prediction, first the data is added to the X array as timesteps value. Then timesteps + 1 is added to y array as output value. The window is scrolled until the end of the data set and the data is added to the X and y arrays. Data is used to train 3 deep learning models that are described in Section 2.1, namely LSTM, GRU, CNN-LSTM models.

Quaternions are units that represent the spatial orientation and rotation of objects in three-dimensional space. In our dataset, we have quaternion data for w, x, y and z dimensions for each time instant. The rotation angle obtained from quaternion data is used as output and IMU data is used as input. Data is used to train deep learning model described in Section 2.2, namely fully connected sequential model.

3.3. Metric Definitions

This section provides definitions of the metrics used in performance evaluations. Parameters used in the metric is defined as y as the actual value, $\hat{y}$ as the predicted value, whereas n is the total number of estimated values.

3.3.1. Mean Absolute Error (MAE)

The average magnitude of errors in a series of predictions or forecasts is measured using the metric known as Mean Absolute Error (MAE). Prior to averaging these discrepancies, it determines the absolute difference between the expected and actual values. Without taking into account the direction of the mistakes, MAE offers a simple and simple-to-interpret measurement of how well a model performs in terms of absolute prediction accuracy as given in (2). Usually, it is represented in the same units as the predicted variable.

$$\frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \qquad (2)$$

3.3.2. Mean Squared Error (MSE)

Mean Squared Error (MSE) measures the average of the squared differences between predicted and actual values as given in (3), giving more weight to larger errors. It is commonly used in regression analysis and machine learning.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \quad (3)$$

3.3.3. Root Mean Squared Error (RMSE)

Root Mean Squared Error (RMSE) is the square root of the average of the squared differences between predicted and actual values as given in (4), providing a measure of prediction accuracy in the original units.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \quad (4)$$

3.3.4. R^2 Score

The R^2 score, also known as the coefficient of determination, is a statistical metric that evaluates the goodness of fit of a regression model. It measures the proportion of the total variance in the dependent variable that can be explained by the independent variables in the model as given in (5), where residual is the difference between the predicted value and the actual value and total sum of squares is defined as the variance of the actual values. Ranging from 0 to 1, a higher R^2 score indicates a better fit, with 1 representing a perfect fit.

$$R^2 = 1 - \frac{\sum \text{squares of residuals}}{\text{total sum of squares}} \quad (5)$$

4. Results and Discussion

The trained deep learning models are evaluated both qualitatively and quantitatively by means of accumulated odometry estimation graphs and metric calculations.

4.1. Qualitative Evaluation on Position Estimation

Birds eye view and 3D accumulative position estimation graphs are given as qualitative evaluation of LSTM, GRU and CNN-LSTM models respectively in Fig. 5. It can be easily seen that GRU has the best accumulated position estimation.

4.2. Quantitative Evaluation on Position Estimation

Metric results for LSTM, GRU and CNN-LSTM models are given respectively in Table 1. As seen in the experimental results, LSTM and RNN models performed similarly with high accuracy in route prediction with time series. The GRU model performed better by a small margin with lower error values and higher R^2 Score. The CNN-LSTM model has a high loss value. As seen in the experimental results section, the error values are quite high. The R^2 Score is as low as 0.25.

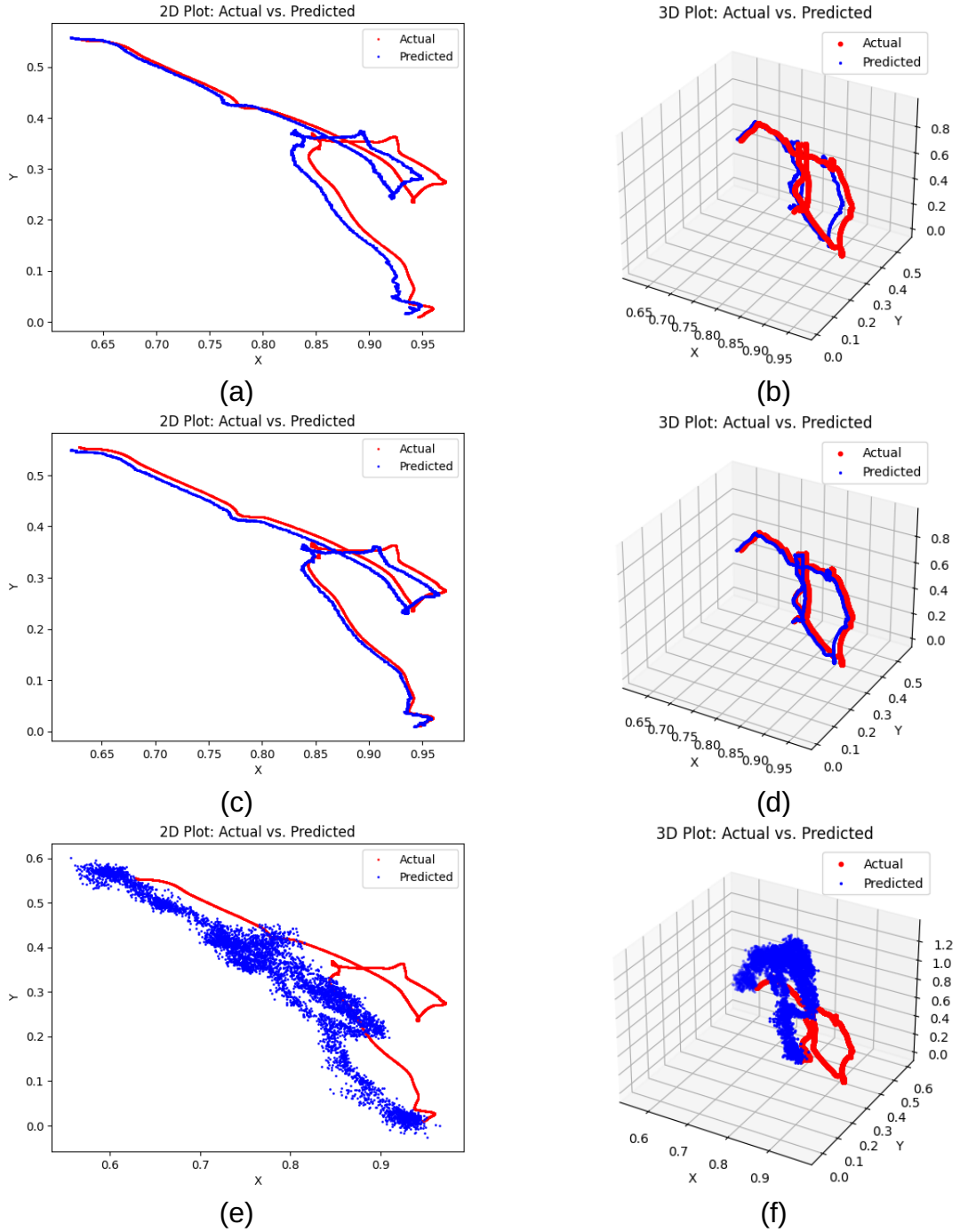


Figure 5. Position Estimation Results for a-b) LSTM, c-d) GRU, e-f) CNN-LSTM

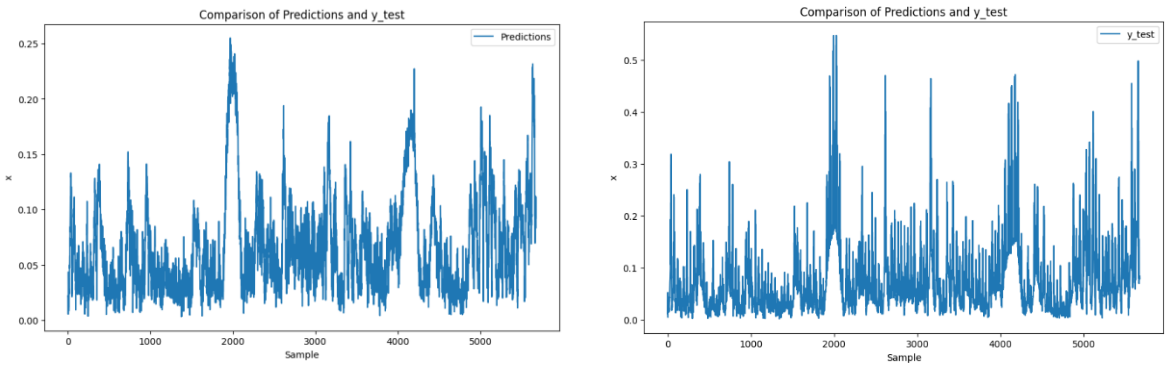
Table 1. Metric Results for Position Estimation

Metric	LSTM	GRU	CNN-LSTM
MAE	0,01	0,005	0,10
MSE	0,00015	0,00004	0,02
RMSE	0,011	0,006	0,12
R^2	0,98	0,996	0,25

In the literature for position estimation, SINS model is reported to have a MAE value of 0,0172 and RMSE value of 0,0198 whereas AbelDeepIO model has a MAE value of 0,0119 and RMSE value of 0,0146 with the same test data. The proposed method in this study has better MAE (0,01) and RMSE (0,011) results compared to those previous two methods. The method proposed by Silva do Monte Lima, however, has a MAE value of 0.0075 and RMSE value of 0.0099 which are better compared to our results.

4.3. Qualitative Evaluation on Orientation Estimation

Orientation change estimation is qualitatively evaluated with estimation and ground truth graphs provided respectively.



(a) (b)

Figure 6. a) Orientation Change Estimation Results for the Fully Connected Model, b) Ground-truth Data

As can be seen in the graphs, the predictions show a remarkable correlation with the actual values, but the accuracy should be further improved.

4.4. Quantitative Evaluation on Orientation Estimation

Metric results for the fully connected model for orientation change estimation are given in Table 2.

Table 2. Metric Results for Position Estimation

Metric	Fully Connected Model
MAE	0,02
MSE	0,001
RMSE	0,040
R^2	0,56

An R^2 Score of 0.56 was obtained for the orientation change with quaternion data.

In the literature for orientation change, SINS model is reported to have a MAE value of 0,0174 and RMSE value of 0,1723 whereas AbelDeepIO model has a MAE value of 0,0134 and RMSE value of 0,1724 with the same test data. Even though the proposed method in this study has a worse MAE (0,02) value compared to previous two methods, RMSE (0,040) value is better those previous two methods.

4. Conclusion

In the study, odometry prediction is carried out using IMU data. The experiments are divided into two main subjects int the first part, route prediction is performed with time series prediction using IMU and ground-truth data. LSTM, GRU and CNN-LSTM models are used for time series prediction. In this study, LSTM and GRU models gave good results while CNN-LSTM model was inadequate with an R^2 score of 0,25.

In the second part, the orientation difference at two time-intervals is estimated as rotation angle using IMU data. The rotation angle obtained from quaternion data is trained in a deep network model with IMU inputs. An R^2 Score of 0,56 was obtained.

The results are promising by means of robustness. Therefore it can concluded that feeding deep learning based IMU odometry prediction into SLAM algorithms as an initial guess for each optimization step can increase the robustness of the SLAM algorithm.

For future work given the results this article obtains, combining IMU odometry and wheel odometry using deep neural networks may provide better results. It is possible that wheel sensor and IMU data complement each other. As the proposed work only relies on IMU and wheel sensors data it can be used in rough environments which are visually challenging and GPS unavailable or unreliable.

References

- Abolfazli Esfahani, M., Wang, H., Wu, K. & Yuan, S., (2020). AbolDeepIO: A Novel Deep Inertial Odometry Network for Autonomous Vehicles. IEEE Transactions on Intelligent Transportation Systems, Volume 21, pp. 1941-1950.
- Burri, M., Nikolic, J., Gohl, P., Schneider, T., Rehder, J., Omari, S., Achtelik, M. W. & Siegwart, R., (2016). The EuRoC micro aerial vehicle datasets. The International Journal of Robotics Research.
- Donahue, J., Hendricks, L. A., Rohrbach, M., Venugopalan, S., Guadarrama, S., Saenko, K. & Darrell, T., (2016). Long-term Recurrent Convolutional Networks for Visual Recognition and Description. IEEE Transactions on Pattern Analysis and Machine Intelligence, 39(4), pp. 677-691.
- Heck, J. C. & Salem, F. M., (2017). Simplified minimal gated unit variations for recurrent neural networks. IEEE 60th International Midwest Symposium on Circuits and Systems. Boston, MA, USA, pp. 1593-1596.
- Lu, C., Uchiyama, H., Thomas, D., Shimada A. & Taniguchi, R.-i., (2019). Indoor Positioning System Based on Chest-Mounted IMU. Sensors, Volume 19.
- Mahmoud, A. & Atia, M. M., (2019). Hybrid IMU-Aided Approach for Optimized Visual Odometry. IEEE Global Conference on Signal and Information Processing.
- Niu, Z., Zhong, G., Yue, G., Wang, L.-N., Yu, H., Ling, X. and Dong, J., (2023). Recurrent attention unit: A new gated recurrent unit for long-term memory of important parts in sequential data. Neurocomputing, Volume 517, pp. 1-9.
- Shan, T. et al., (2020). LIO-SAM: Tightly-coupled Lidar Inertial Odometry via Smoothing and Mapping. IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 5135-5142.
- Silva do Monte Lima, J. P., Uchiyama, H. & Taniguchi, R.-i., (2019). End-to-End Learning Framework for IMU-Based 6-DOF Odometry. Sensors, Volume 19.

- Titterton, D., Weston, J. L., (2004). Strapdown Inertial Navigation Technology. Institution of Engineering and Technology.
- Toy, I., Durdu, A. & Yusefi, A., (2022). Improved Dead Reckoning Localization using IMU Sensor. International Symposium on Electronics and Telecommunications.
- Van Houdt, G., Mosquera, C. & Nápoles, G., (2020). A review on the long short-term memory model. Artificial Intelligence Review, Volume 53, p. 5929–5955.
- Zheng, S., Wang, J., Rizos, C., Ding W., & El-Mowafy, A., (2023). Simultaneous Localization and Mapping (SLAM) for Autonomous Driving: Concept and Analysis. Remote Sensing, February, Volume 15, p. 1156.