TRIM - TRansformers for tracking Inertial Measurements

Daniel Eskandar, Florian Sprick, and Philipp Davydov

Eberhard Karls University of Tübingen

Abstract. Existing methods for inferring pose estimation from IMU measurements typically employ either multiple IMUs or integrate IMUs with additional sensors like cameras or LiDAR. Recent work involves data-driven approaches utilizing deep learning models such as CNNs and RNNs to predict position from a single IMU. However, these techniques often specialize in predicting either position or orientation or confine predictions to 2D space. In this paper, we aim to predict the 6-DoF pose of an object directly from 6D measurements of a single IMU. We present transformer-based architectures for direct orientation prediction and for predicting denoised velocities and accelerations. Our proposed method is evaluated on the TUM-VIE dataset and demonstrates superior performance compared to previous models. The code is made publicly available.[1]

Keywords: inertial navigation, deep learning, pose estimation, trajectory estimation, transformer, inertial measurement unit

1 Introduction

Inertial odometry, employing Inertial Measurement Units (IMUs), is vital across diverse applications like autonomous driving and human-object interaction, as it offers essential motion and positioning data. Additionally, IMUs are advantageous due to their low-power consumption, privacy-preserving nature, and robustness in various environments. However, low-cost inertial sensors, frequently experience high levels of noise and biases, causing unpredictable error escalation within the system [4]. While integrating additional sensor data or cameras presents a potential avenue for improving pose estimation accuracy, such an approach may necessitate compromising the inherent flexibility provided by IMUs. Recent methods have employed Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) to establish a mapping between IMU measurements and pose. Nonetheless, while CNNs fail to capture long-term dependencies, RNNs encounter issues with the non-parallelizability of sequential computation. In contrast, Transformers [10] are superior to RNNs and CNNs in handling sequential data because their self-attention mechanism allows for parallel processing and better capturing of long-range dependencies, and are yet to be adopted for this task.

[1] `https://github.com/Daniel-Eskandar/TRIM-TRanformers-for-tracking-IMus`

The main idea of this work is to leverage the transformer architecture to provide accurate pose predictions from IMU measurements. Our main contributions include: 1) Introducing three transformer-based architectures for learning corrected acceleration, velocity, and direct orientation from IMU data, 2) Extending pose estimation to encompass 6D poses (3D position and 3D orientation), and 3) Demonstrating superior results in predicting pose estimation. We evaluate our method on the TUM-VIE Dataset [7] that includes challenging sequences where state-of-the art visual intertial odometry fails or results in large drift.

2 Related Work

2.1 Classical Inertial Navigation

Strapdown Inertial Navigation System (SINS) is a classic model for position recovery using acceleration integration. Linear accelerations, $a_w(t)$, are obtained from IMUs in the world frame. Using the IMU's kinematic model, we express velocity v_w, position p_w in the world frame and rotation R_b^w from the body frame to the world frame as:

$$\begin{cases} v_w(t+1) = v_w(t) + a_w(t) \cdot dt \\ p_w(t+1) = p_w(t) + v_w(t) \cdot dt + \dfrac{1}{2} a_w(t) \cdot dt^2 \\ R_b^w(t+1) = R_b^w(t) \cdot R_{b_t+1}^{b_t} \end{cases} \tag{1}$$

While prone to drift, this method is commonly enhanced with deep learning models for predicting adjusted accelerations. Conversely, classical techniques like Pedestrian Dead Reckoning (PDR) [5] rely on domain-specific features such as the periodicity of walking for motion tracking. However, PDR approaches are restricted to pedestrian navigation and lack generalizability to other motion types.

2.2 Deep Learning Based Inertial Navigation

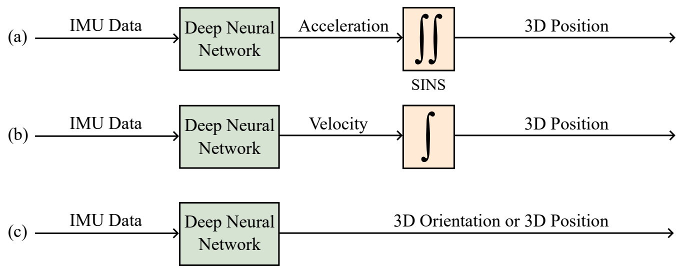

Fig. 1. Overview of deep learning based inertial navigation systems.

Recent research investigates data-driven approaches employing deep learning to develop inertial positioning models, partially or entirely replacing conventional inertial navigation systems. Deep learning models can directly estimate quantities like 3D orientation or position. They can also infer velocity or corrected accelerations, which are then used with SINS to determine position. An overview of the deep learning based inertial navigation methods is shown in Figure 1.

Deep neural networks are capable of processing raw IMU measurements to produce calibrated outputs, accurately representing motion dynamics. In [11,3], researchers leverage RNNs and CNNs to mitigate accelerometer errors. Similarly, various methods employ deep learning models for velocity estimation, with IONet [2] utilizing Bi-LSTMs and RoNIN [6] employing ResNet. Finally, other approaches directly regress position from IMU data. For example, TLIO [8] employs ResNet to estimate 3D displacement, while [9] proposes a modular design integrating LSTMs and an Extended Kalman Filter (EKF) for orientation and position estimation. RIOT [1], a closely related method, adopts transformers to directly infer true positions and orientation, achieving pose-invariant deep inertial odometry. Nevertheless, the potential of transformers in calibrating acceleration measurements and inferring corrected accelerations remains unexplored.

3 Method

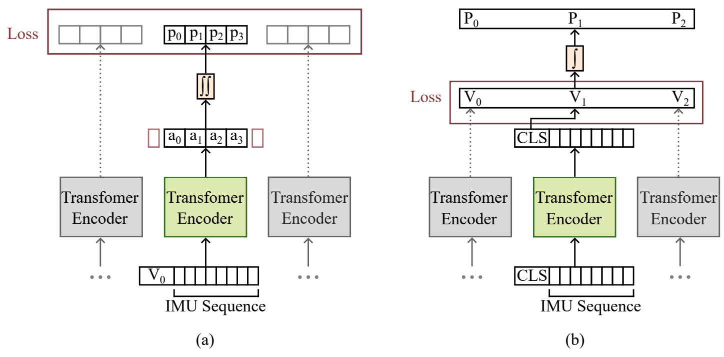

Fig. 2. Architectures for position estimation. **(a)** A transformer encoder is utilized to learn calibrated and denoised acceleration values from IMU sequences. The acceleration sequence is processed through double integration using SINS. **(b)** RoNIN-based model with transformer backbone. The IMU sequence is concatenated with a CLS token and passed through a transformer encoder, after which the CLS token is decoded into a velocity vector. The model is trained to predict the average velocity of the input window.

We present two novel methods for position estimation and one method for orientation estimation based on the transformer architecture [10]. During our experiments, mapping directly from IMU measurements to 3D positions (see Fig. 1c), effectively replacing the double integration, proved to be unstable and hard to train. Therefore, we explore the approaches from Fig. 1a and Fig. 1b, namely denoising the acceleration measurements or mapping from accelerations to ground truth velocity values, in more depth. The used architectures for position estimation are depicted in Fig. 2, the network for orientation estimation is shown in Fig. 3.

For (a), we use a window of IMU measurements together with its initial velocity as inputs to a transformer. We then learn cleaned and denoised acceleration values, which return a correct position sequence after SINS. To ensure that all predicted acceleration values have the necessary context, we drop the ten first and last output elements of our model. Consequently, at inference time, the window is moved along the IMU sequence (window_size − 20) elements at a time and the returned positions are concatenated. The model is trained to minimize the L1 loss between the output positions and ground truth positions.

For (b), we propose to alter the RoNIN architecture by replacing the ResNet backbone by a transformer backbone. As in RoNIN, the aim is to learn the average velocity for every input window of IMU measurements. To that end, we add a (-1)-vector to the beginning of any IMU measurement sequence, which serves as a CLS token. The output value of the CLS token is decoded into a velocity vector using a feed forward network, forcing the model to compress a representation of the sequence into the CLS token during training. Again, L1 loss is used. At inference time, following RoNIN, we use a sliding window approach with low stride and mathematically integrate the obtained velocity sequence. To produce a smoothed trajectory which matches the dimensions of the ground truth, we perform interpolation on the point sequence. In addition to the altered RoNIN architecture, we also train ResNet-based RoNIN as a baseline method.

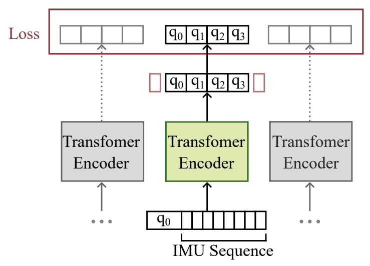

Fig. 3. Transformer-based model for predicting the orientation. The IMU sequence is concatenated with the initial orientation and passed through a transformer encoder model. The first and last ten values of the output are dropped to provide enough context for the remaining values. The architecture is trained using quaternion loss.

To track the orientation, we implement a model similar to (a), but provide the initial orientation in quaternion form for every input window. Again, the ten first and last output values are dropped to ensure that the remaining values have enough context. Since IMU's measure angular velocity, the model effectively aims to replace one integration layer, while also removing noise and converting to quaternions. We employ the quaternion loss function used by RIOT [1], which measures the angle between a predicted and a ground truth quaternion.

4 Evaluation

We trained and tested our method on different tracks from the TUM-VIE dataset [7], which encompasses a large variety of handheld and head-mounted sequences in indoor and outdoor environments. The dataset comprises 3-axis accelerometer and gyroscope data collected at 200 Hz by the IMU unit, accompanied by ground truth poses recorded from a motion capture system (MoCap) at 120 Hz, at the beginning and end of each sequence. For processing, we aligned MoCap and IMU sequences by computing nearest neighbors.

For the purpose of comparability, we adopt the same metrics used in [1] and [6], i.e., the Absolute Trajectory Error (ATE) and Relative Trajectory Error (RTE). First represents the global accuracy of the estimated position, while last one quantifies the location accuracy of the trajectory over a fixed time interval $\Delta t = 1s$.

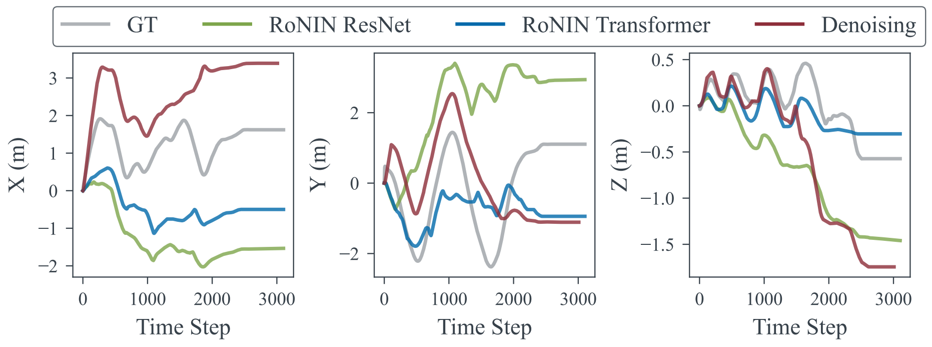

Fig. 4. Results of evaluated methods for position estimation task on the track "slide-vi-end". The ground truth is shown in gray.

Fig. 4 provides visualizations of the position estimates for each approach during which ground truth labels are available. Table 2 shows our evaluation results for three groups of tracks, namely "skate-hard", "slide" and "mocap", which contain the tracks "mocap-6dof-vi" and "mocap-desk-vi". SINS was not included, since it immediately exhibits heavy drift and is not comparable to the other methods. The transformer backbone in RoNIN consistently outperforms the ResNet backbone on all tracks. The transformer-based denoising network is more prone to drift, but achieves comparable values for some tracks.

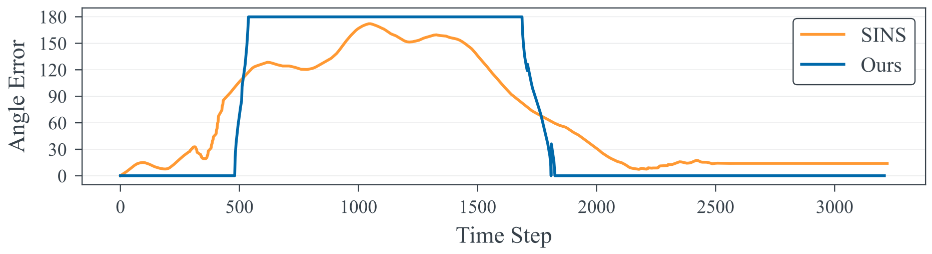

Fig. 5. Comparison of SINS and our method on the "skate-hard" track. Our method tracks the orientation correctly for most of the time, but experiences a mirror effect between time steps 500 and 1750, where instead of the target orientation, the orientation opposite to that is tracked. SINS exhibits drift and shows inaccurate orientation predictions for most of the track.

Table 1. Heading estimation, compared between the SINS baseline and the transformer based network using the mean angle error (MAE). For the performance per window, we pre-process the dataset similar to the training setting, by splitting it into windows of size 120 and providing the initial ground truth orientation for every window. For the performance per track, we "roll out" the orientation model by using the last predicted orientation of one window as the initial input for the next window.

Track	**Metric**	**SINS**	**Ours**
Skate-Hard	MAE (Window)	79.18°	**0.18°**
	MAE (Track)	**73.43°**	79.79°
Slide	MAE (Window)	78.99°	**0.08°**
	MAE (Track)	84.12°	**81.3°**
Mocap	MAE (Window)	176.43°	**0.08°**
	MAE (Track)	174.41°	**0.08°**

To evaluate the performance on pose estimation, we use the mean angle error (MAE) between the predicted and ground truth quaternions as a metric. We differentiate between evaluating on input sequences with provided initial ground truth orientation, and evaluating by using a sliding window approach, providing the last predicted orientation of one window as the initial orientation for the next window. Table 1 shows that our approach outperforms naive SINS for most cases. Further analysis of the track-wise evaluation is performed by analyzing the MAE curve, shown in Fig. 5. Even though our model achieves lower MAE scores for "skate-hard", the curve showcases that it tracks the orientation more consistently than SINS. The high MAE arises from a "mirror effect", where the orientation is flipped accidentally, after which the model continues to track the orientation by predicting the exact opposite of every ground truth heading, i.e., in relation to the mistakenly flipped orientation, the motion is tracked correctly. In contrast to that, SINS smoothly accumulates drift which leads to various amounts of heading error throughout the track.

Table 2. Position evaluation. We compare three competing methods: Denoising, and RoNIN (2 variants) on the dataset. The top three results are highlighted in red, green, and blue colors per row.

Track	**Metric**	**Denoising**	**RoNIN**	
			ResNet	**Transformer**
Skate-Hard	ATE (m)	2.116	2.156	**1.281**
	RTE (m)	8.516	8.397	**7.356**
Slide	ATE (m)	1.514	1.588	**1.041**
	RTE (m)	7.892	6.914	**5.810**
Mocap	ATE (m)	1.173	1.334	**0.737**
	RTE (m)	7.295	**5.396**	5.419

5 Conclusions

In this paper, we presented two novel methods for transformer-based position estimation and one method for orientation estimation based on IMU sequences. Using transformers shows promising results on our dataset, outperforming other approaches such as RoNIN. To conclude, we discuss limitations of our approach as well as perspectives for future work.

5.1 Limitations

Each of the approaches used comes with certain downsides. The main limitation of the acceleration denoising approach is that it is still susceptible to error accumulation, as it uses SINS after predicting the "cleaned" acceleration data. Even if trained to a very low error margin, the estimated position will drift if enough time passes and no intermediate points are known to regularize the prediction. On the other hand, RoNIN-based models suffer from smoothing, because they only estimate one velocity per window. Therefore, fine-grained movements cannot be captured as accurately as with the denoising approach. The window size may be tuned as a hyperparameter to counteract this. The orientation estimation is prone to drift and the mirror effect as discussed in section 4.

5.2 Future Work

Our work opens diverse perspectives for future work. Since we reduced the IMU data to the frequence of mocap, a future approach is to utilize the full frequence of IMU to obtain even more accurate results. Moreover, larger transformer models with better optimized hyperparameters may be trained, allowing for larger window sizes, more context and better predictions. Future work may also focus on the combination of transformer-based architectures and regularization of position estimates if further information is known, such as the type of activity performed or the exact mounting position of the IMU on an object or body. With our paper, we aim to contribute to the exploration of human-object interaction.

References

1. Brotchie, J., Li, W., Greentree, A.D., Kealy, A.: Riot: Recursive inertial odometry transformer for localisation from low-cost imu measurements. Sensors **23**(6) (2023). https://doi.org/10.3390/s23063217, `https://www.mdpi.com/1424-8220/23/6/3217`
2. Chen, C., Lu, X., Markham, A., Trigoni, N.: Ionet: Learning to cure the curse of drift in inertial odometry. In: McIlraith, S.A., Weinberger, K.Q. (eds.) Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018. pp. 6468–6476. AAAI Press (2018). https://doi.org/10.1609/AAAI.V32I1.12102, `https://doi.org/10.1609/aaai.v32i1.12102`
3. Chen, H., Aggarwal, P., Taha, T.M., Chodavarapu, V.P.: Improving inertial sensor by reducing errors using deep learning methodology. In: NAECON 2018 - IEEE National Aerospace and Electronics Conference. pp. 197–202 (2018). https://doi.org/10.1109/NAECON.2018.8556718
4. El-Sheimy, N., Hou, H., Niu, X.: Analysis and modeling of inertial sensors using allan variance. IEEE Transactions on Instrumentation and Measurement **57**(1), 140–149 (2008). https://doi.org/10.1109/TIM.2007.908635
5. Harle, R.: A survey of indoor inertial positioning systems for pedestrians. IEEE Communications Surveys Tutorials **15**(3), 1281–1293 (2013). https://doi.org/10.1109/SURV.2012.121912.00075
6. Herath, S., Yan, H., Furukawa, Y.: Ronin: Robust neural inertial navigation in the wild: Benchmark, evaluations, new methods. In: IEEE International Conference on Robotics and Automation (ICRA). pp. 3146–3152. IEEE (2020)
7. Klenk, S., Chui, J., Demmel, N., Cremers, D.: Tum-vie: The tum stereo visual-inertial event dataset. In: International Conference on Intelligent Robots and Systems (IROS) (2021)
8. Liu, W., Caruso, D., Ilg, E., Dong, J., Mourikis, A.I., Daniilidis, K., Kumar, V., Engel, J.: Tlio: Tight learned inertial odometry. IEEE Robotics and Automation Letters **5**(4), 5653–5660 (Oct 2020). https://doi.org/10.1109/lra.2020.3007421, `http://dx.doi.org/10.1109/LRA.2020.3007421`
9. Sun, S., Melamed, D., Kitani, K.: Idol: Inertial deep orientation-estimation and localization (2021)
10. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. In: Proceedings of the 31st International Conference on Neural Information Processing Systems. p. 6000–6010. NIPS'17, Curran Associates Inc., Red Hook, NY, USA (2017)
11. Wang, Y., Cheng, H., Wang, C., Meng, M.Q.H.: Pose-invariant inertial odometry for pedestrian localization. IEEE Transactions on Instrumentation and Measurement **70**, 1–12 (2021). https://doi.org/10.1109/TIM.2021.3093922