Deep Learning Precise 3D Human Pose from Sparse IMUs

Suparna Guha[1] and Hammad Tanveer Butt[2]

[1] guha@rhrk.uni-kl.de
[2] hammad.butt@dfki.de

Abstract. This paper represents our approach to estimate dense 3D human pose only from 6 IMU sensor orientation data using Deep Learning. There are many existing popular methods to solve inverse kinematic problem which mostly rely on iterative optimization to seek out an approximate solution. But their approach is infeasible for real time prediction. With deep learning real time prediction becomes possible. Our work is motivated from Deep Inertial Poser and we have used datasets published by them. Our approach can produce relatively convincing results from only orientation data from 6 IMUs. Instead of rotation matrix we used quaternion to represent orientation and pose which reduced the network input size.

Keywords: 3D Human Pose, IMU, Deep Learning

1 Introduction

3D Human pose estimation has been an active field of research among computer vision community for decades. Although with the help of deep learning recent camera based methods can generate satisfactory result, it becomes infeasible and fails in the wild or outdoor setting due to occlusion and other environmental factors. In contrast wearable sensors like IMUs are small, lightweight and low-cost and give measurement of orientation and acceleration. So they can facilitate capturing real time outdoor activities in a more natural way. However, to estimate full human pose, at least 17 sensors need to be attached at different body segments which makes this approach non-viable for complex motions. So our goal is to predict full human pose from sparse IMUs (i.e. IMUs only placed at body extremities) using deep learning. More specifically subject needs to wear only 6 IMUs at left & right lower legs, left & right arms, head and pelvis. This is an under-determined problem and so we impose anatomical body constraints (SMPL [3] body model) to discard impractical human poses. Besides, there exist a temporal dependency in human motion sequences like walking, jumping, eating or any particular limb movements. So Recurrent Neural Network can be an effective approach towards this. Our work is mainly inspired from Deep Inertial Poser(DIP)[2] and for this task we have used their published datasets. We have performed some experiments with different approaches and compared their qualitative and quantitative results.

2 Relative Work

Our work starts with the work represented by Deep Inertial poser[2] which tried to address this problem and came up with a novel deep neural network capable of reconstructing full human body pose from 6 IMU sensors at real time.

The end to end pipeline of DIP[2] can be described as follows: 6 IMUs are placed on the subject body at head, pelvis, left and right arms, left and right lower legs. They give raw measurements of orientation and acceleration. These raw data are first calibrated to SMPL[3] reference frame. Then all sensors are normalized with respect to the root sensor which results into 5 sensor readings (root sensor reading becomes identity). This calibration and normalization is done for all the frames in the sequence. The orientation is represented by 3 X 3 rotation matrices and acceleration comprises

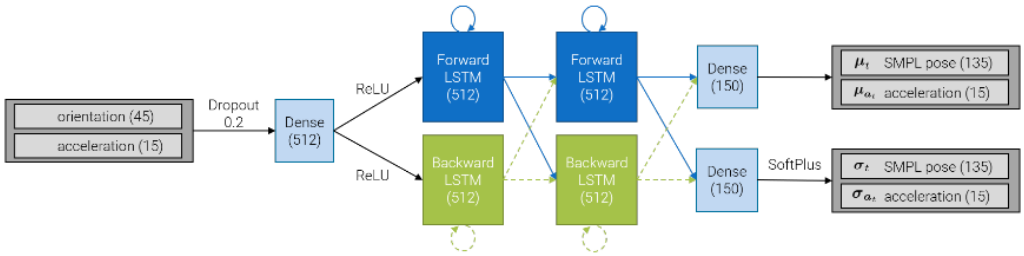


Fig. 1: Network architecture of Deep Inertial poser [2]: Bi directional stacked two layered LSTM with two fully connected layers at the two ends.

of three components (x,y,z). Orientation (5*9=45) and acceleration (5*3=15) from 5 sensors are concatenated resulting a single long input feature vector of size 60. This is fed into the network (fig 1) which can predict pose parameters for 15 joints. As reported in the DIP paper, during training their network had access to full sequence but we found that in the published DIP_IMU_nn training set, all activities were split into chunks of 300 time-steps which arises confusion if their network was trained over full sequence or in chunks. However, for evaluation they preferred window based approach with 20 past and 5 future frames. So to predict pose at time-step t, the input window contains total 26 frames starting from t-20 up to t+5 frames. DIP has provided their best performing two pre-trained models. One was trained on synthetic data and another one was further fine-tuned on real DIP_IMU data. We performed test on these two models with the activities from their DIP_IMU_nn test set and the quantitative result in terms of mean joint angle error is shown in fig 2. The result ascertains the effectiveness of fine-tuning on DIP_IMU data which mitigates the gap between synthetic and DIP_IMU real data distributions and difference in motions.

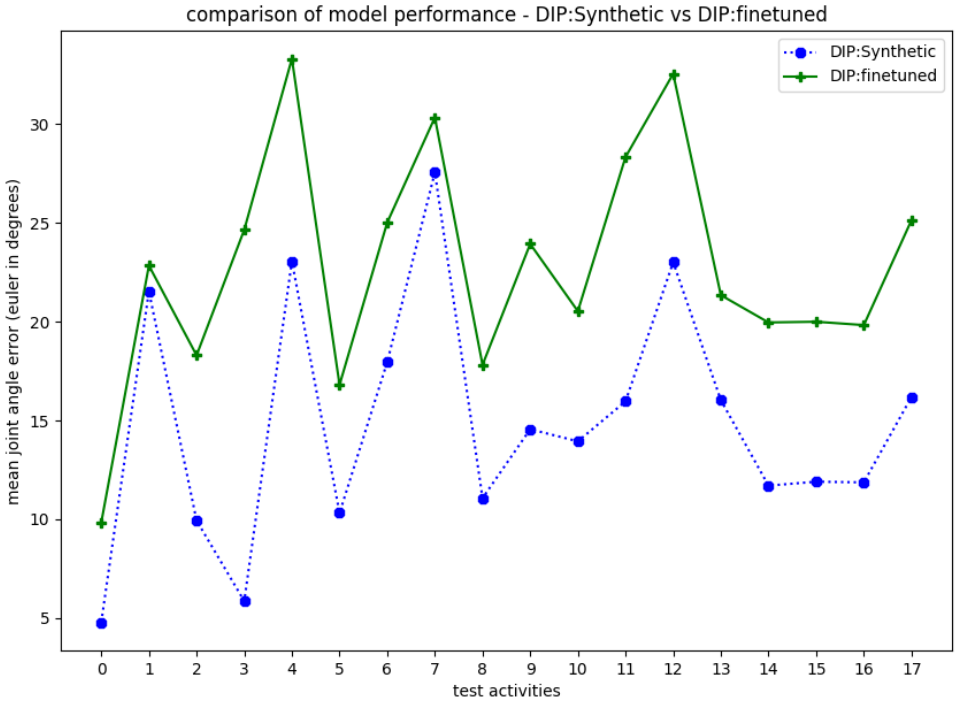


Fig. 2: Error of DIP models on DIP_IMU_nn test activities

2.1 SMPL body model

Skinned Multi-Person Linear model (SMPL[3]) is a skinned vertex-based model that accurately represents a wide variety of body shapes in natural human poses. It is parameterized by 72 pose, and 10 shape, parameters θ and β respectively, and returns a mesh with N = 6890 vertices. Here, in our case, we are interested in pose parameters which are represented as axis angle form (3 components) for total 24 joints (24 * 3 = 72). As aforementioned DIP model predicts 15 joints, to map into SMPL the remaining joints are assumed as equivalent rotation vector of identity rotation matrix. In fig 3 marked joints are predicted by the model.

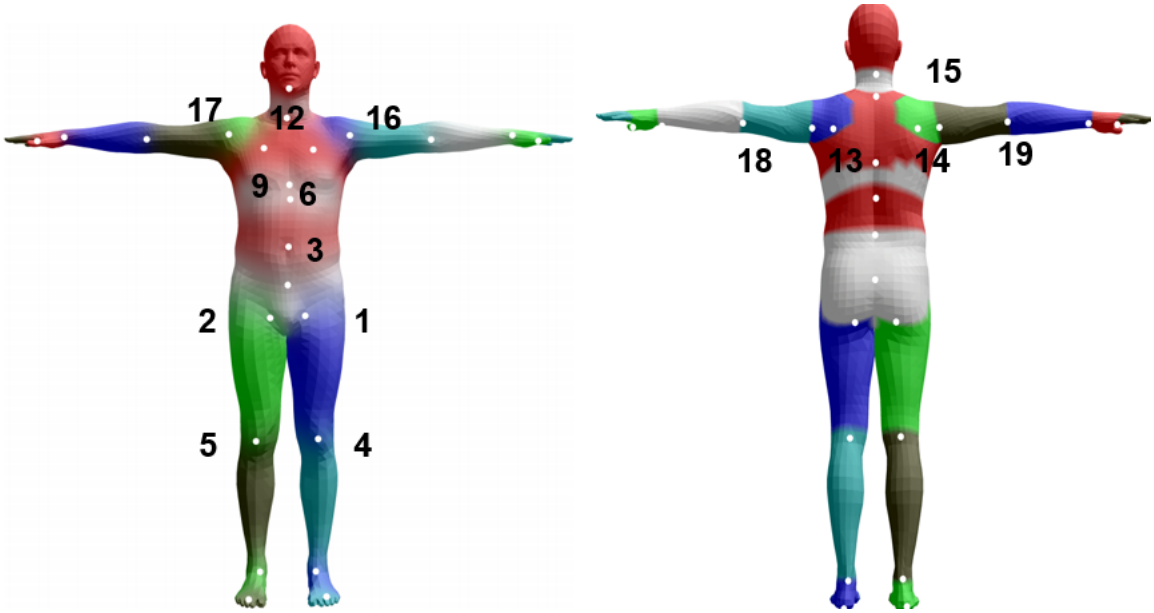


Fig. 3: SMPL needs 24 joints. The marked joints are predicted ones.

2.2 Orientation representation

There are several methods to represent orientation in three dimensions.

Euler angle: The first attempt to represent an orientation was owed to Leonhard Euler. Euler angles provide a way to represent the 3D orientation of an object using a combination of three rotations about different axes. In aerospace they are known as yaw-pitch-roll rotation ($z - y' - x''$). So this is parameterized by three variables. Euler angles are limited by a phenomenon called "gimbal lock," which prevents them from measuring orientation when the pitch angle approaches +/- 90 degrees.

Axis-angle representation: This representation describes a rotation or orientation using a unit vector aligned with the rotation axis, and an angle indicating the magnitude of rotation about the axis. This form is equivalent to more concise form known as rotation vector which is obtained by multiplying the angle with the unit rotation axis vector. It also has 3 parameters.

Rotation matrices: The product of two rotation matrices is the composition of rotations. Therefore, the orientation can be given as the rotation from the initial frame to achieve the frame that we want to describe. The rotation matrices are 3 X 3 orthogonal matrices and it has 9 parameters.

Quaternion: Another way to describe orientation in 3D coordinate system is unit quaternions which is a four element vector and is composed of one real element and three complex elements. Compared to Euler angles they are simpler to compose and avoid the problem of gimbal lock. Compared to rotation matrices they are more compact, more numerically stable, and more efficient.

3 Our Methodology

3.1 Data preparation

For our work we have used the datasets published by DIP_IMU. The dataset can be downloaded from `http://dip.is.tuebingen.mpg.de/pre_download`. At the time of writing this paper, three main folders were available. **DIP_IMU :** This was their recorded uncalibrated raw IMU data from 17 Xsens sensors worn by 10 subjects. **DIP_IMU_nn :** This was calibrated and normalized version of the activities from DIP_IMU. This was further split into training set consisting of 40 activities, test set consisting of 18 activities and validation set consisting of 3 activities. **Synthetic60FPS :** This was uncalibrated IMU data synthesized at 60 frames/sec mainly from CMU, H36 and AMASS dataset.

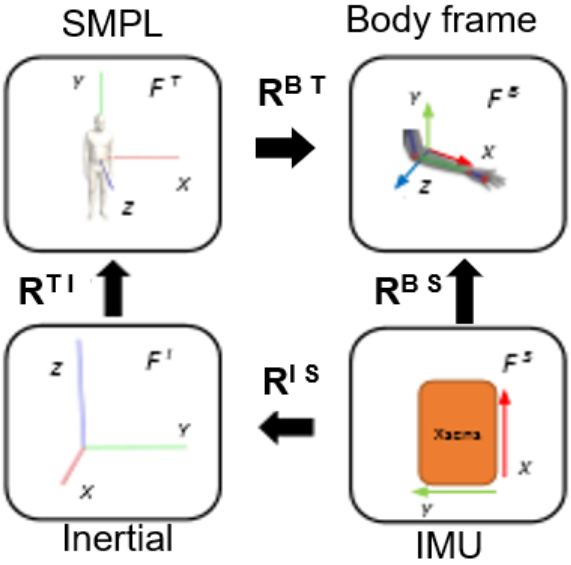


Fig. 4: Calibration steps from sensor coordinate to SMPL coordinate

Calibration Before feeding into the model raw IMU data must be transformed to our reference target SMPL frame F^T. For the detailed calibration procedure, we contacted with authors of DIP and tried to implement it following their direction which is summarized below.

All rotation matrices given below should be understood as F^{AB}: rotation from frame A to frame B. We will talk about four frames denoted as follows T: SMPL frame, B: bone coordinate, I: inertial frame, S: sensor local coordinate. Fig 4 gives the visual understanding of the coordinate frames involved in the entire calibration process.

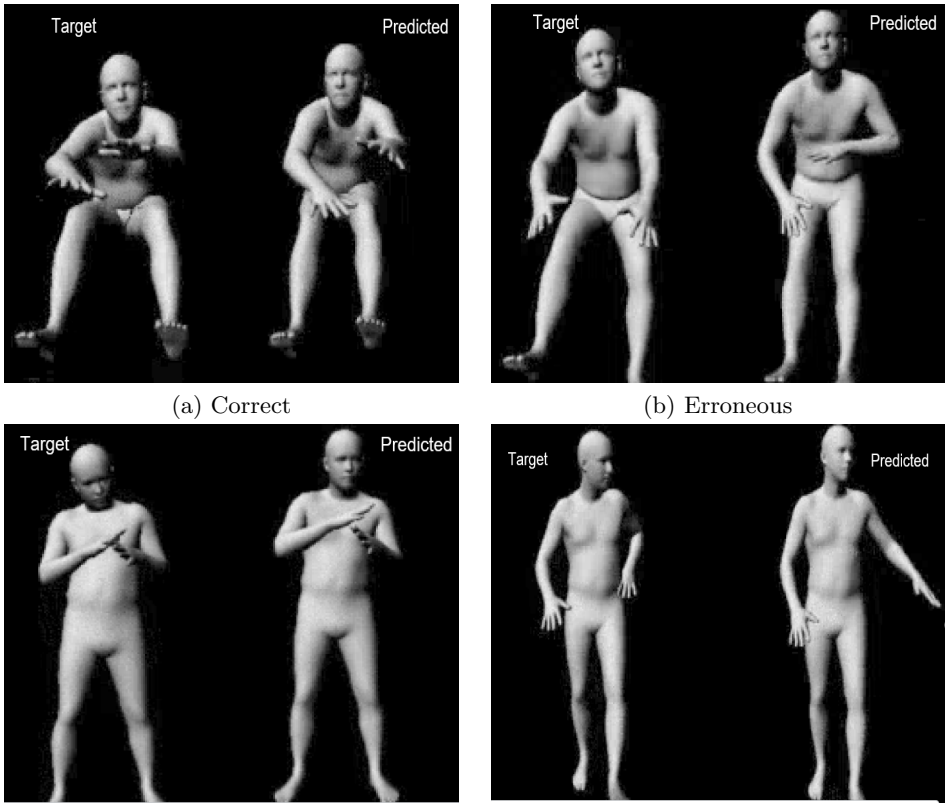
1. IMU provides absolute orientation of each sensor relative to a global inertial frame F^I in terms of $R^{IS} : F^S \rightarrow F^I$. So our goal is to find the mapping $R^{TI} : F^I \rightarrow F^T$, relating the inertial frame to the SMPL body frame.
2. All IMU readings can then be expressed in the SMPL body frame by the following equation $R_t^{TS} = R^{TI} R_t^{IS}$.
3. Here R^{TI} is constant over all frames and calculated as $R^{TI} = inv(R^{IS} R^{SB} R^{BT})$ where R^{IS} is considered as the head sensor readings at 1st frame assuming head sensor is aligned with SMPL axes and value of R^{SB} and R^{BT} needs to be known. Here $inv(\cdot)$ denotes matrix inverse.
4. In the first frame of each sequence, each subject stands in a known straight pose with known bone orientation R_0^{BT}.
5. We compute the per-sensor bone offset as $R^{BS} = R_0^{BT} R_0^{TS}$.
6. The last step is to calculate $R^{TB} = R^{TS} inv(R^{BS})$ for every frame. The output R^{TB} can be interpreted as the bone orientations measured by the IMU in SMPL frame. Here $inv(\cdot)$ denotes matrix inverse.

Normalization The calibrated IMU data are further normalized with respect to root sensor to make the motions invariant to any directions. Below equation calculates the normalized orientation $R_s^{TB}(t) = (R_{root}(t))^{-1} R_s^{TB}(t)$, while $R_{root}(t)$ denoting the orientation of the root at time step t,

and $R_s^{TB}(t)$, the orientation of the bone corresponding to sensor s at time step t. Thus the root sensor readings becomes identity matrix and so discarded and remaining 5 sensor data are used for training.

3.2 Our approaches

1st approach As mentioned earlier, due to more numerical stability, we have chosen quartenion over rotation matrices. A rotation matrix is 3 X 3 square matrix and has 9 parameters while a quaternion has 4 parameters (w,x,y,z). So now orientation of 5 sensors is given as (5*4=20) 20 dimensional vector. So after concatenating orientation and acceleration our new input size reduces from 60 to (4+3)*5 = 35. Similarly prediction of 15 joints was represented by (15*4=60) 60 dimensional vector. Now we trained a similar Bidirectional 2 layered LSTM as in DIP with the required modification in input and output layer on synthetic H36 dataset. Some results on the unseen test data are shown in fig 5.



(a) Correct (b) Erroneous

(c) Correct (d) Erroneous

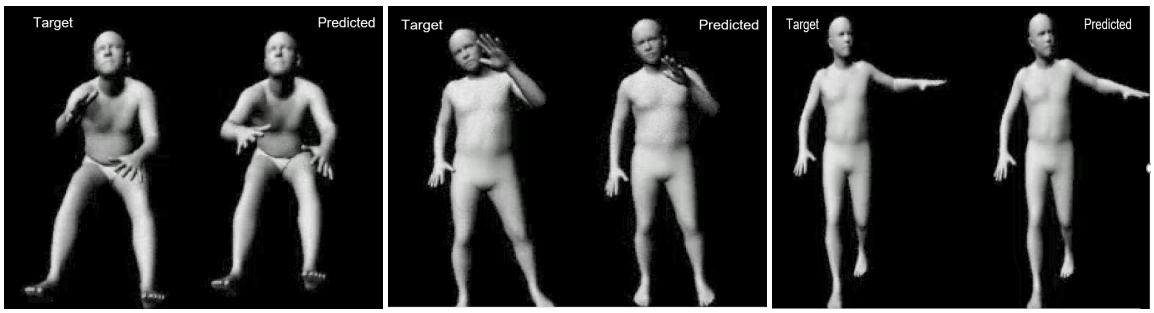
Fig. 5: Target and predicted frames from 1st approach. Fig 5a & fig 5b represent correct & erroneous frame respectively of same activity. Same applies for fig 5c & 5d.

2nd approach When we visualized the prediction from 1st approach, we found that the predicted poses are near to ground truth pose but there was a lot of jittering. Next,we further eliminated acceleration from our input and predicted pose only from orientation of 5 IMUs. Now with 20 dimensional input we trained a model and found a very smooth satisfactory result. Acceleration data from IMUs are generally very noisy and so, if it is added in input, it needs to be handled in the loss function.

Fig. 6: Simple Multi-layer Perceptron

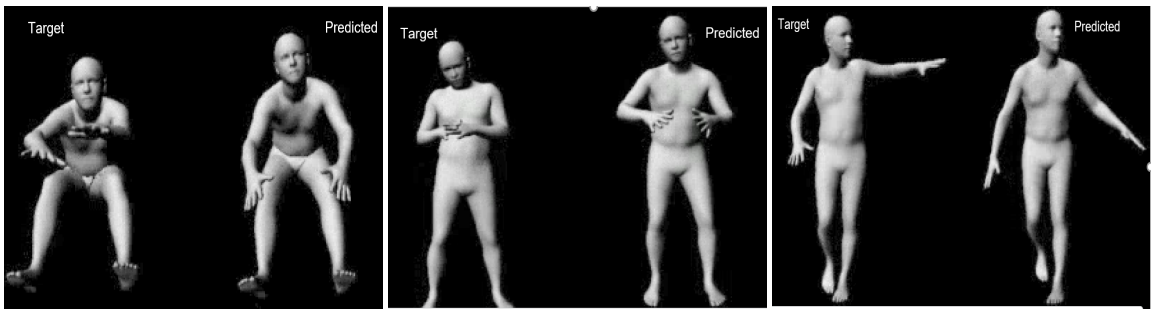
So rather than making it complex, we first tried a simpler approach of not considering acceleration as determining factor. With this we achieved far better results and also it eased the task of the network by reducing the number of parameters to learn. Some example results from this model are presented in the fig 7.

3rd approach Next, we have performed an experiment with similar input and output on a very simple 3 layered perceptron. The network architecture is given in fig 6. We wanted to compare the result from MLP with LSTM. A MLP is able to learn a non-linear mapping from input variables to output variable while LSTM network is mainly used to learn the temporal dependency for sequence data. The result from MLP was worth to consider and some glimpses of that can be visualized in fig 8.



(a) (b) (c)

Fig. 7: Target and predicted frames from 2nd approach. Compared to fig 5 error is much reduced and prediction became smoother without jittering.



(a) (b) (c)

Fig. 8: Target and predicted frames from 3rd approach. There were no jittering like fig 7 but predicted poses are less accurate.

Comparison of approaches Fig 9 shows the quantitative comparison of our aforementioned three approaches on left out test data (H36-S11) containing total 30 activities. It is clear that our 2nd approach outperforms others for all the activities. So BiLSTM network with only orientation input in quaternion form gave the best results so far to predict 15 joints in quaternion. As desired, MLP performs poor than BILSTM network for this problem because human motion has temporal pattern between the subsequent frames which can be better learned by long short term memory network.

The performance of models are compared in terms of mean joint error across all frames for all activities.

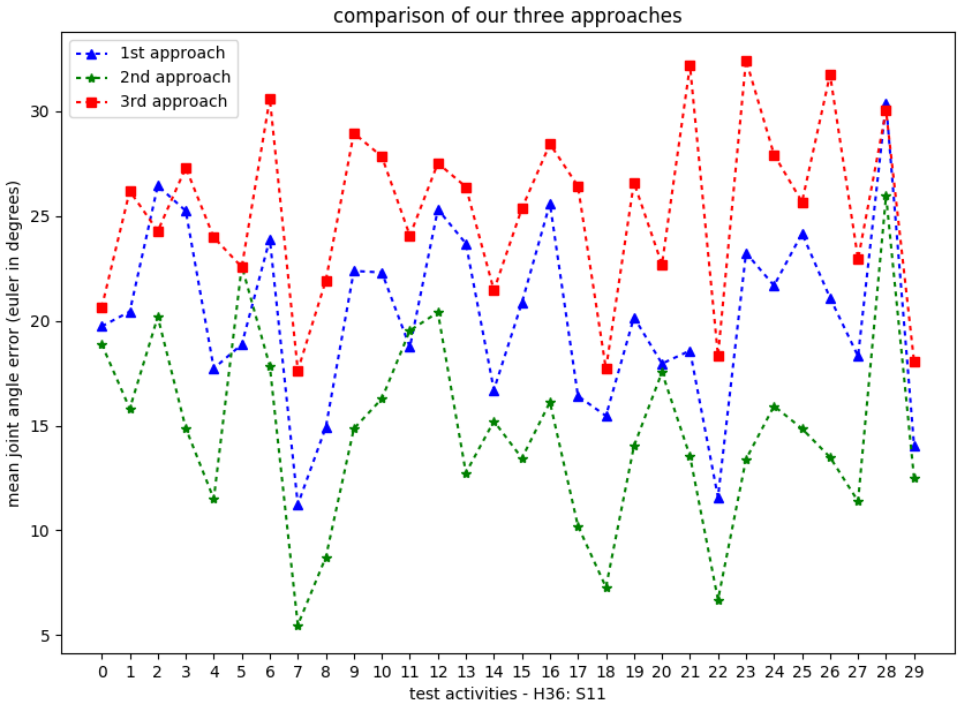


Fig. 9: Comparison of our approaches on left out test set H36-S11 containing 30 activities in terms of mean joint error in euler angle(degree).

3.3 SMPL Model Visualization

For the qualitative understanding of the prediction of our model the visualization of activities is very important. For that we adapted SMPL python packages (`http://smpl.is.tue.mpg.de/downloads`) to our problem. To generate human poses at each frame we only change the pose parameters as per the prediction from our model and other parameters remain constant for any activity. We have used openCV[1] for visualization of frames and further merge into video.

4 Miscellaneous Experiments

4.1 Single vs all dataset

As seen in the previous section our 2nd approach produced the best results, so we proceeded with it and trained a model across all synthetic datasets. With larger dataset, our model generalizes overall but the performance drops for any specific dataset. To illustrate, in fig 10, we compared model a: trained on H36 dataset with model b: trained across all datasets. Model a outperformed model b when tested on H36 test set but it performs worse when tested on different dataset, e.g AMASS Transition. This gives us an intuition of multimodal distribution across our synthetic datasets.

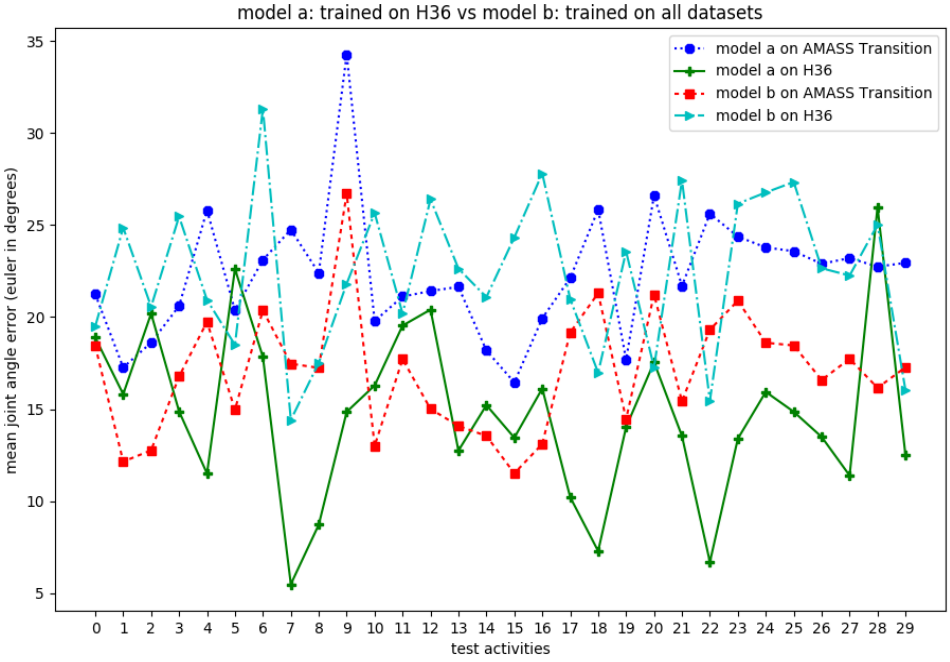


Fig. 10: Model performance on two different test datasets - H36 & AMASS_Transition while model a is trained on H36 & model b is trained on all 12 synthetic datasets.

4.2 LSTM training strategy

Fig. 11: short BiLSTM Network

Throughout the implementation of our experiments we followed different training strategies for better learning of the model. Our synthetic datasets contain activities of varied sequence length in the range from 200 to 5500 almost. From some failed attempts, we found that the longer sequences are very much prone to destroy LSTM hidden and cell state learning. Empirically it was found that 200 sequence length was optimal to learn in our problem. So, we divided all the longer activities into chunks of smaller sequence length(200 in our case). We created batch of 10 activities of 200 time steps each randomly with replacement. We also tried stateful batch learning where for each new activity hidden and cell state are initialized by 0 and the intermediate state is maintained throughout the whole activity by initializing cell and hidden state from last timestep of previous batch. We observed that maintaining state did not improve our result due to comparatively long sequences for LSTM .

5 Challenges

This section explains the main challenges encountered in our task. While we tested the DIP_IMU dataset on the model trained on synthetic dataset, it failed. We could not get any interpretable output for DIP_IMU data from the same model that worked before for synthetic data.

5.1 Dataset distribution

The root cause lies in the different distribution of initial poses for all activities across all datasets. For correct calibration measurement, subject should start any activity either with T-pose or I-pose with known bone orientation in that pose. Fig 12a shows the output pose distribution at the first frame for all activities of different datasets. It is clearly visible that the first pose is not same for different activities in a single dataset and further the whole distribution of one dataset is far different than another. This effects our calibration procedure.

5.2 Concept shift

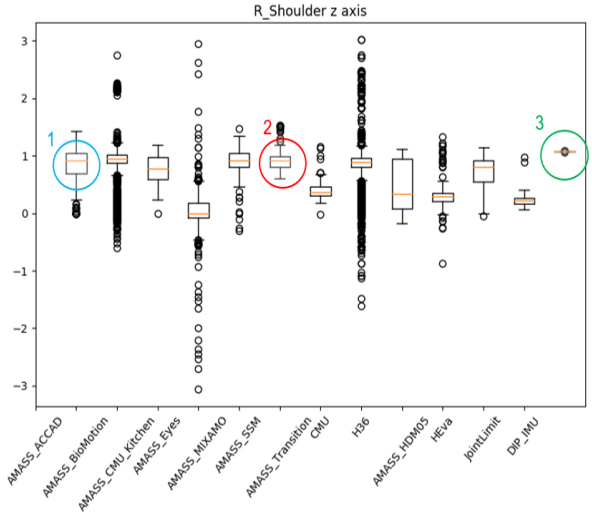
Concept shift is one type of dataset shift problem in machine learning. Dataset shift refers to the change in the distribution of data contained in train and test set that will lead to unreliable predictions. In our case, train set was synthetic data and test set was DIP_IMU real recorded data. Now in fig 12, we have highlighted three datasets, DIP_IMU,AMASS_ACCAD, AMASS_SSM.

Fig 12a shows they have similar output pose distribution but fig 12b shows they have different input orientation distribution. So the relationship between input and target variable differs which is known as concept shift problem. In ideal case after accurate calibration, dataset having similar pose should have similar input orientation. But in our case the problem was not solved even after calibration as visible in fig 12c.

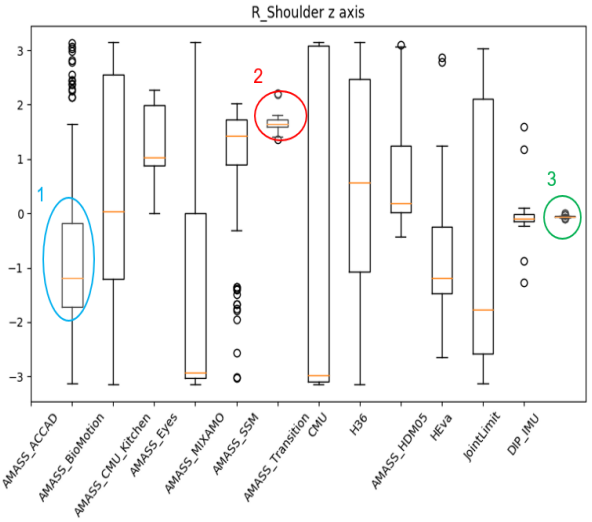
6 Analysis of our approach with DIP_IMU data

6.1 Training

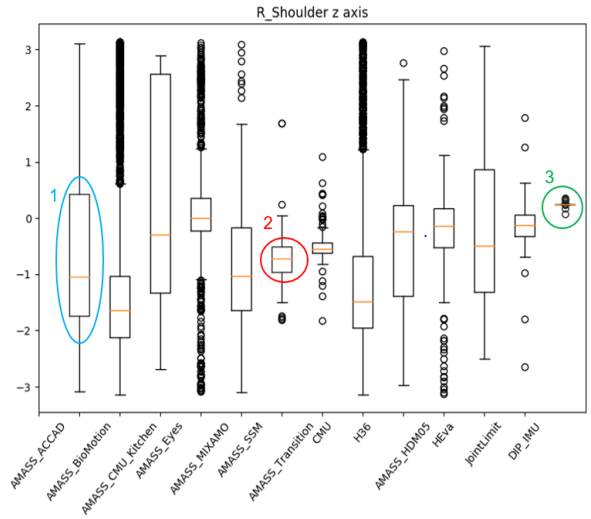
Further we verified our network and approaches on provided calibrated DIP_IMU_nn data which is already split into train set(40 activities), test set(18 activities) and validation set(3 activities). For that we trained two models - short BiLSTM (Fig 11) and MLP(fig 6) on this data. The results ascertain that the problem discussed above being the reason of not getting any satisfactory results from our previous models for DIP_IMU data.



(a) Pose distribution at 1st frame

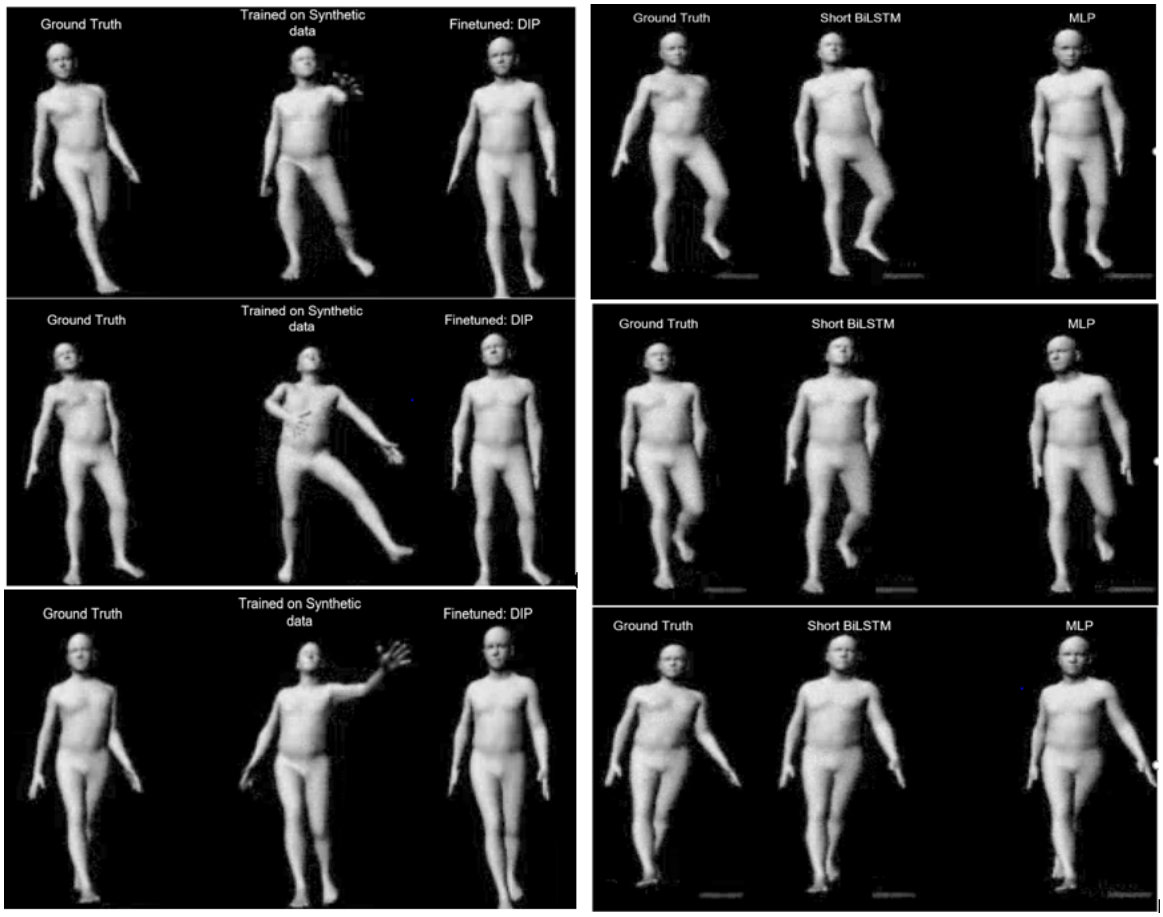


(b) Raw orientation distribution at 1st frame



(c) Calibrated orientation distribution at 1st frame

Fig. 12: Insights of dataset distribution



(a) (b)

Fig. 13: Some target and prediction visualization on DIP_IMU_nn testset. Fig 13a represents result from DIP state-of-the-art models. Fig 13b represents results from our own trained models.

6.2 Comparison with benchmark

These two models were tested on the DIP_IMU_ nn test set containing 18 activities. We have compared the performance of following four models, i.e provided best two models of DIP, one trained on synthetic data, another finetuned on DIP_IMU_nn data, our trained short BiLSTM and MLP models trained only on DIP_IMU_nn data. The test was performed on 18 activities of DIP_IMU_nn test set. Some qualitative results are presented in fig 13. Out of these 4 models, state of the art model trained on synthetic data performs worst on this test set. We also analyzed the quantitative results and presented in fig 14. Clearly, their model trained on synthetic data gave highest error for almost all activities, and their finetuned model performs best. Our trained MLP comes in 3rd rank. Our short BILSTM model can produce good results considering the limitation of dataset. Also it performed better in some activities than their finetuned model.

7 Conclusion

Towards the conclusion of this project, we have found that orientation and pose parameters presented in quaternion can reduce input parameters and cost computation. Further it simplifies the training

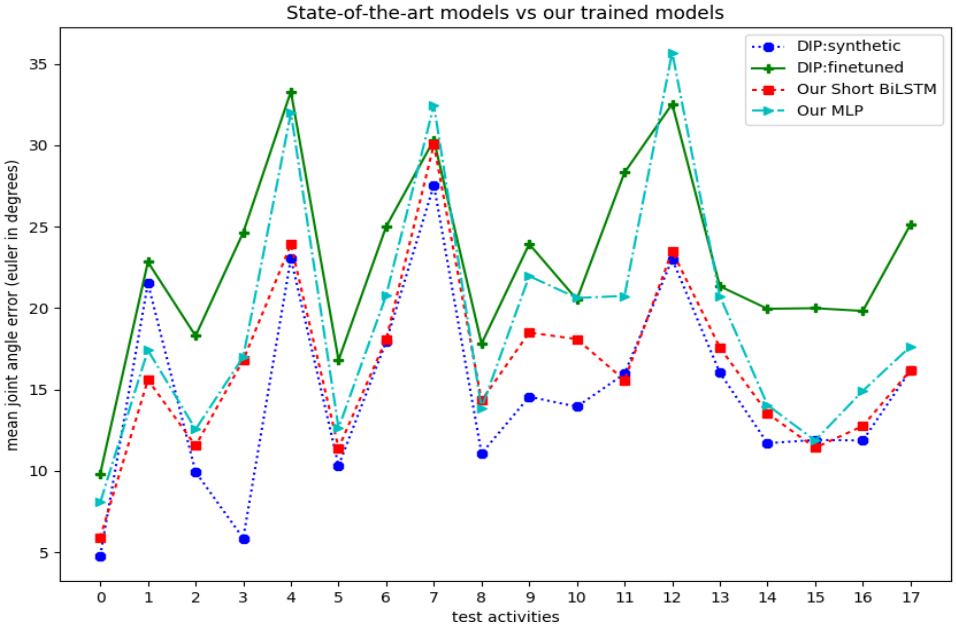


Fig. 14: Comparison of quantitative error on DIP_IMU_nn test activities between state-of-the-art models and our trained models.

of model and stabilizes the weight parameters fast. We trained our model for maximum 30 epochs and it started converging. After that, eliminating acceleration improved our result from jittering. Due to temporal dependency in human motions, LSTM performs better than MLP. Considering synthetic data for training and DIP_IMU data for test we observed that dataset suffers from concept shift problem even after calibration. The limitations of calibration procedure were not mitigated properly due to varied starting pose, unknown bone offset and different alignment of calibration sensor between synthetic and DIP_IMU dataset. Our proposed model still performs better than DIP:synthetic model mostly for all activities and in some cases than DIP:finetuned model, when using their calibrated data of DIP IMUs.

References

1. Gary Bradski and Adrian Kaehler. *Learning OpenCV: Computer vision with the OpenCV library.* " O'Reilly Media, Inc.", 2008.
2. Yinghao Huang, Manuel Kaufmann, Emre Aksan, Michael J Black, Otmar Hilliges, and Gerard Pons-Moll. Deep inertial poser: learning to reconstruct human pose from sparse inertial measurements in real time. In *SIGGRAPH Asia 2018 Technical Papers*, page 185. ACM, 2018.
3. Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J Black. Smpl: A skinned multi-person linear model. *ACM transactions on graphics (TOG)*, 34(6):248, 2015.
4. Dario Pavllo, David Grangier, and Michael Auli. Quaternet: A quaternion-based recurrent model for human motion. *arXiv preprint arXiv:1805.06485*, 2018.
5. Gerard Pons-Moll, Andreas Baak, Juergen Gall, Laura Leal-Taixe, Meinard Mueller, Hans-Peter Seidel, and Bodo Rosenhahn. Outdoor human motion capture using inverse kinematics and von mises-fisher sampling. In *2011 International Conference on Computer Vision*, pages 1243–1250. IEEE, 2011.
6. Timo von Marcard, Bodo Rosenhahn, Michael J Black, and Gerard Pons-Moll. Sparse inertial poser: Automatic 3d human pose estimation from sparse imus. In *Computer Graphics Forum*, volume 36, pages 349–360. Wiley Online Library, 2017.