CS230

Human Marker Augmentation with Deep Learning using Constraints

Aditya Agrawal
Department of Computer Science
Stanford University
adityaag@stanford.edu

Suguna Velury
Department of Electrical Engineering
Stanford University
sugunav@stanford.edu

Hermann Kumbong
Department of Computer Science
Stanford University
kumboh@stanford.edu

Abstract

A novel approach for measuring the dynamics of human motion presented is to use human markers generated from pose estimation and augment them using LSTMs to get a larger set of markers. Although this approach has produced good results, it still fails to generalize to other complex human motions like nordic exercises and jumping. In our work, we address this by incorporating knowledge of the physical constraints of human motion like invariant lengths, relative body part ratios, and bounded joint angles into the LSTM model. Our approach yields better qualitative results than the baseline for a subset of complex human motions.

1 Introduction

Measurement of joint angles and kinematics when analyzing human movement for biomechanical applications is traditionally done via marker-based motion capture. However, this is resource intensive, time-consuming, and requires expertise which has limited its usage to small-scale research studies. As an alternative to marker-based motion capture, video pose detection estimates joint positions from videos. Our work focuses on an existing solution called OpenCap [1]. In OpenCap, a pose detection algorithm like OpenPose is used to detect the pose from videos recorded from multiple views after which triangulation is used to reconstruct the 3D trajectories of the identified video keypoints, and then inverse kinematics is performed to estimate joint angles using a simulation tool like OpenSim [1]. Video pose estimation however suffers from a lack of accuracy due to the sparse nature of the video keypoints identified by most pose detection algorithms which leads to inaccurate inverse kinematics. In order to increase accuracy and robustness, human body marker augmentation (i.e. generating new body markers) from video keypoints using deep learning has been introduced in [1]. The idea of human body marker augmentation, as demonstrated in Figure 1, is to generate a quantitatively enhanced set of anatomical markers from a reduced set of video keypoints. This augmentation is needed to generate more accurate musculoskeletal simulations. One method of doing this is to use markers generated from a pose estimation model (OpenPose) as inputs to an LSTM and generate a more comprehensive set of markers for the simulation tool to work with [1].

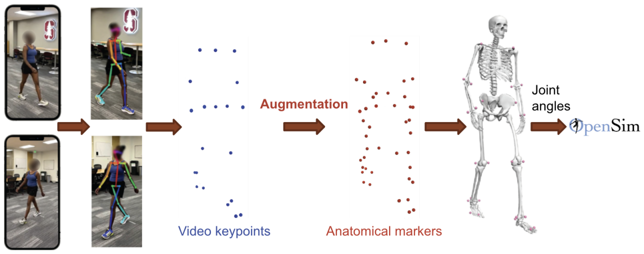


Figure 1: Using Anatomical Markers Generated From Video Keypoints to Estimate 3D kinematics

CS230: Deep Learning, Fall 2022, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

While this method of training LSTMs to generate augmented markers has generated reasonable results, there are several cases for which the model does not generalize well on out-of-distribution samples involving complex human motion [1]. Some examples of such activities involve jumping down from a raised platform and nordic exercises.

We seek to explore ways to remedy this issue by using deep learning integrated with human knowledge about the motion of the human body. In particular we will be exploiting the observation that the human body has rigid bone segments and can only exhibit a restricted range of motion [2].

2 Related work

Our work builds upon [1]. To understand more about the physical constraints of the human skeletal system and methods of incorporating it in human pose estimation we refer to [2].

We aim to incorporate the physical constraints of the human body. There are two ways to approach this: using hard constraints (constraints that "must" be satisfied at all times) or soft constraints (constraints that we "want" to be satisfied as much as possible). [3] talks about how hard constraints do not offer any significant benefits over soft constraints when trying to keep the operations computationally feasible. Hence we decided to move ahead with incorporating soft constraints through penalty terms in the loss function.

[4] is related to our work as it aims to incorporate bone length invariance constraints. However, this paper aims to impose these constraints to aid pose estimation and motion tracking of the markers, while we aim to generate better-augmented markers for musculoskeletal analysis. [5] is similar to our work in the sense that it incorporates soft constraints through penalties in the loss function in a different context and achieves reasonably positive results. However, the main objective is to enable label-free learning as opposed to our case, which involves generating more robust markers. [6] incorporates physics based constraints into the loss function to predict the trajectory of a pendulum. This paper uses an approach similar to our work, which is incorporating an energy-conservation constraint in the loss term which yields better results in terms of the predicted outputs.

Some other approaches include enforcing constraints within the architecture of the model using a constrained optimization layer on top of the neural network [7] and adopting constraint-enforcing architecture design [8]. The approaches in these papers differ from our implementation in the sense that they modify the architecture of the model but not add any additional constraint terms to the loss function. [9] compares architecture constrained neural networks (ACNets) and loss constrained neural networks (LCNets) and concludes that both are similar in terms of their performance with ACNets performing only slightly better than LCNets. [10] talks about the benefits of incorporating constraints into the neural network and concludes that this technique helps in generalization and faster training.

3 Dataset and Features

The data set consists of 972 hours of motion capture data, which was extracted from 10 open-source data sets consisting of 336 subjects performing various activities like walking, running, squatting, cutting, and jumping. In the interest of time and efficiency, we will be using only one of the ten data sets, consisting of about 17 hrs of motion capture data. The data is stored as 121555-time sequences of length 0.5s. Each 0.5s sequence consists of 30 frames (i.e. 30 sets of input video key points and output anatomical markers). A set of input video key points consists of 20 body markers each represented as a set of normalized 3D coordinates relative to the hip bone of the subject (refer Figure 1). A set of output anatomical markers on the other hand consists of 43 body markers each represented as a set of normalized 3D coordinates as well (refer Figure 1). We intend to use 80% of the data for training, 10% for testing, and 10% for validation. Moreover, in order to encourage generalization, the data set will be augmented by scaling and rotating the input sequences. We also add slight gaussian noise into the input data to simulate real life measurement errors. There also exists a small set of examples where the existing baseline models fail which will be reserved for testing. Two such examples involve jumping from one platform to another and nordic exercises. Note that the smaller scale of our data set and the small sequence length of 0.5s can potentially limit the performance of our model.

4 Methods

Expanding on the observations from Section 2, we sought to incorporate two biological facts into our models using penalty terms: 1) The body consists of rigid bone segments whose shape is approximately preserved during motion, and 2) The body can only exhibit a fixed range of angular motion. We translated these two facts into three kinds of constraints: 1) Distances between output markers on the same segment should be conserved across time-steps, 2) Angles formed by various predicted segments of body should be within the natural human ranges, 3) Distances between input and output markers should be conserved across time-steps. In the following subsections we introduce the baseline loss and our proposed soft penalty terms for each of the constraints respectively.

4.1 Baseline

The baseline loss function consists of a mean squared error term that is optimized. This loss function does not contain any human knowledge regarding the physical constraints of the human body. The equation of the cost function for the baseline is as follows:

$$\frac{1}{3MN}\sum_{i=1}^{N}\sum_{j=1}^{M}(\|y^{(i,j)} - \hat{y}^{(i,j)}\|_2^2) \tag{1}$$

4.2 Output Length Constrained (OLC) Model

As part of our first iteration of improvement, we decided to incorporate the invariance of human bone lengths into the LSTM model [4]. This was done by modifying the original cost function to add a penalty term to incentivize the model to conserve the distance between pairs of markers lying on the same bone segment across various time steps. Formally, given the true coordinates of M output markers across N time steps (denoted $y^{(i,j)}$ ($i \in [N]$ and $j \in [M]$)), corresponding predicted co-ordinates $\hat{y}^{(i,j)}$ and a set of pairs of marker indices representing a pair of output markers to be constrained (denoted C, where $(j,k) \in C \iff$ output marker j and output marker k are on the same segment), our proposed OLC penalty term is as follows (where λ_1 is a scaling hyper-parameter that needs to be tuned):

$$\lambda_1 \sum_{i=1}^{N-1} \sum_{(j,k)\in C} \left(\|\hat{y}^{(i+1,j)} - \hat{y}^{(i+1,k)}\|_2^2 - \|\hat{y}^{(i,j)} - \hat{y}^{(i,k)}\|_2^2\right)^2 \tag{2}$$

4.3 Output Angular Constrained (OAC) Model

In our second iteration of improvement, we decided to incorporate the limitations in the range of motion of human joints into the LSTM model. This was implemented by adding a constraint term that penalizes the loss function if the angle between two rigid segments in the human body goes beyond its physical limits. To implement these constraints we first compute the centroid of the augmented markers on each segment/joint. We use the centroid of the joint between two bones/segments as the reference to generate two segment vectors and calculate the cosine between these vectors. After computing their deviation from the expected min cosine cos_{min} and max cosine cos_{max} of the ranges allowed by the human body, we use a relu function to penalize the loss term if the cosine between the segments is out of range. Mathematically, let us define C_{s1}, C_{s2} and C_{ref} as the centroids of the output markers on segment1, segment2 and the reference respectively. Considering we have A angular constraints and N time steps, the angular constraints can be described as below (where λ_2 and λ_3 are scaling hyperparameters that need to be tuned)

$$\begin{aligned}&\lambda_2 \sum_{a=1}^{A}\sum_{i=1}^{N} relu(\cos((C_{s1,a,i} - C_{ref,a,i}),(C_{s2,a,i} - C_{ref,a,i})) - (\cos_{min,a})) + \\ &\lambda_3 \sum_{a=1}^{A}\sum_{i=1}^{N} relu((\cos_{max,a}) - (\cos((C_{s1,a,i} - C_{ref,a,i}),(C_{s2,a,i} - C_{ref,a,i}))))\end{aligned} \tag{3}$$

For a given angular constraint lets say segment K has $|K|$ markers (each marker containing 3 dimensions x,y,z). The centroid of this segment C_{sK} is calculated as

$$C_{sK} = \frac{1}{|K|}\sum_{i=1}^{|K|}\left(\hat{y}_x^{(i)}, \hat{y}_y^{(i)}, \hat{y}_z^{(i)}\right) \tag{4}$$

The cosine between the segment vectors V1 and V2 is calculated as

$$\cos(V1, V2) = \frac{V1 \cdot V2}{\|V1\|\|V2\|} \tag{5}$$

4.4 Input Output Length Constrained (IOC) Model

In the third iteration of our improvement, we decided to incorporate constraints between the input and output markers. The motivation behind this idea is that if the input markers generated by OpenPose are consistent with the actual motion, then we can enforce the notion of invariance of human bone lengths [4] by imposing constraints on distances between input and output markers on the same bone segment.

Similar to the OLC model, to implement input-output marker constraints, we constrained the output markers from each segments with the input markers from the same segment i.e. we incentivize the model to preserve distances between input and output markers on the same segment across time steps. Formally, given the coordinates of L input markers and the predicted coordinates of M output markers across N time steps (denoted $x^{(i,j)}$ and $\hat{y}^{(i,k)}$ respectively (where $i \in [N]$, $j \in [L]$, $k \in [M]$)), and a set of pairs of marker indices representing a pair of output markers to be constrained (denoted C, where $(j, k) \in C \iff$ input marker j and output marker k are on the same segment), our proposed IOC penalty term is as follows (where λ_4 is a scaling hyper-parameter which needs to be tuned):

$$\lambda_4 \sum_{i=1}^{N-1} \sum_{(j,k) \in C} \left(\|x^{(i+1,j)} - \hat{y}^{(i+1,k)}\|_2^2 - \|x^{(i,j)} - \hat{y}^{(i,k)}\|_2^2 \right)^2 \tag{6}$$

5 Experiments and Results

5.1 Baseline

For the baseline model, emulating [1], we built and trained a uni-directional LSTM [11] with 96 hidden units, 2 hidden layers, a learning rate of 5e-05, a batch size of 64 and using mean squared error between the true and predicted output markers as the loss. Additionally, to prevent overfitting, the training method utilized early stopping [12] with a patience of three on the validation loss. Note that the hyper-parameters were chosen as per [1].

5.2 Models, Hyperparameter Tuning & Evaluation

Besides the baseline we decided to train one best model for each type of penalty term, so that we could investigate their individual impacts independently. We performed hyper-parameter tuning on the learning rate and the scaling parameters λ_1, λ_2, λ_3, and λ_4 for the output length constraint term, max angular constraint term, min angular constraint term, and the input-output constraint term respectively. Note that we decided against tuning model parameters such as the number of hidden units, number of layers, etc. and set them equal to the baseline, to isolate the effect of the loss terms for fairer comparison between the different models. We used random sampling over the hyper-parameters (as opposed to a grid search) to maximize the number of values considered for each hyper-parameter. We eventually used a log-scale for sampling each hyper-parameter as each of the hyper-parameters have an exponential impact on the model. We picked the best hyper-parameters based on which yielded which model yielded the highest evaluation score. To evaluate the model we used the validation mean-squared-error, as we prefer models that have lower marker prediction error. Due to the low penalty of OLC and OAC we additionally decided to combine the penalty terms for output length constraints and output angular constraints to investigate their combined effect on the output marker performance. We ran hyperparameter tuning on this as well.

The hyper-parameters obtained for our different experiements are summarized in section 5.3.

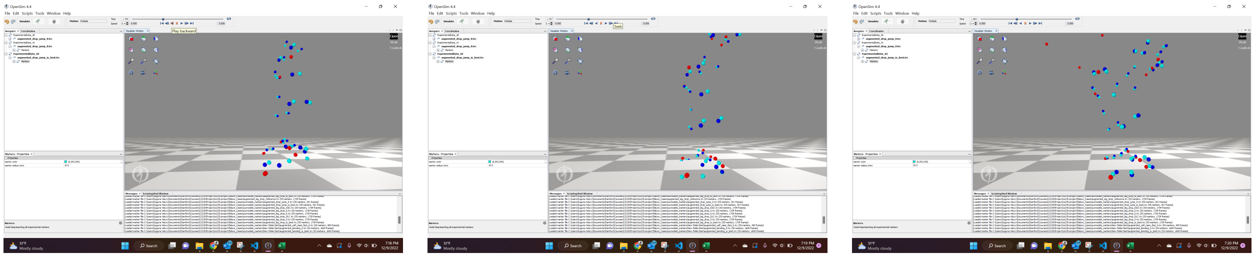
5.3 Quantitative Results

Model	LR	$\lambda 1$	$\lambda 2$	$\lambda 3$	$\lambda 4$	train_mse	val_mse	loss	val loss
Baseline	5.00e-05	NA	NA	NA	NA	2.78e-05	3.78e-05	2.78e-05	3.78e-05
OLC	4.22e-05	1.1187	NA	NA	NA	3.27e-05	3.90e-05	3.27e-05	3.91e-05
OAC	1.71e-05	NA	0.0527	0.0117	NA	6.92e-05	6.68e-05	7.03e-05	6.68e-05
OLC + OAC	4.38e-05	1.2492	0.9198	0.0438	NA	9.58e-05	7.15e-05	1.02e-04	7.56e-05
IOC	1.80e-05	NA	NA	NA	0.01	0.003	0.0027	0.0223	0.0202

Table 1: Summary of optimal hyperparameters, loss and evaluation metrics (mse) for each trained model

As expected for each model the validation loss is greater than the validation mean-squared error. This is due to the extra non-negative penalty terms in the loss function. One observation from the hyper-parameters is that the total loss has different proportions of contributions from mean-squared error loss term and the penalty terms. For instance the penalty term contributes to 1% of the total loss in the OLC case while the penalty term contributes to 85% of the total loss in the IOC case. This indicates that certain constraint violations (like IOC) have greater influence on the overall performance of the optimal model. Note that the training curves were excluded as they did not carry any relevant insights (refer to the repository for some samples).

5.4 Qualitative Results

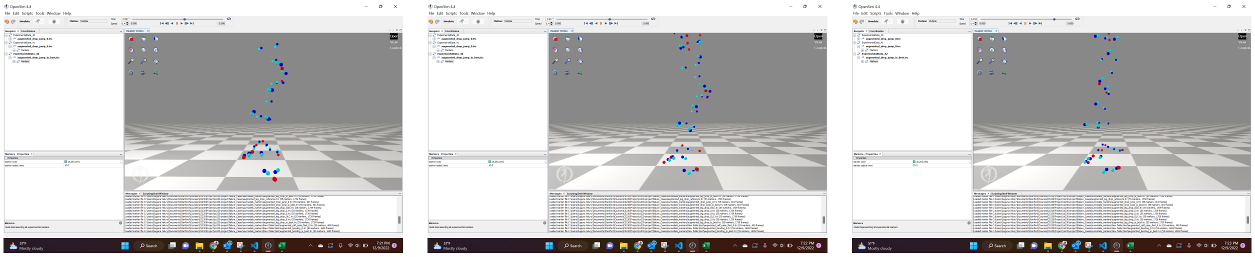


Before point of contact

During point of contact

After point of contact

Figure 2: Jump 1



Before point of contact

During point of contact

After point of contact

Figure 3: Jump 2

Our qualitative analysis involved visualizing the output markers by each model on the small set of failure cases as referred to in section 3 (a sample can be seen in Figures 2 and 3, other samples not included due to space constraints). The visualizations brought out particular insights about the failure cases. On observing the failure cases, we noticed that most failure cases in which the predicted markers overshoot the true output, consisted of prolonged high-velocity motion in one direction. We hypothesize that this is due to the fact that LSTMs have memory, i.e. the LSTM model tracks the marker motion and at the point of contact, LSTM predicts marker trajectory if it had not encountered a halt. In a sense, the LSTM models intertia in physical systems. Our penalty terms in the loss function serve to counteract this inertia by placing a bound on the overshooting of the markers. In the case of the failure case analyzed in 5.4, the first jump is off a ledge, i.e. there are two components of velocity. There is a lesser component of velocity in the vertical direction. Hence our penalty terms were enough to overcome the LSTMs memory implications and successfully resolve the failure. However, the second jump had a larger velocity component in the vertical direction had hence our penalty terms were not enough to completely resolve the issue, which is an area of future exploration.

6 Conclusion & Future Work

In summary, introducing penalty terms in the loss function is an effective and computationally feasible way of incorporating human-knowledge into a neural network. The penalty terms in our loss functions served to counter the overshooting of the predicted markers seen in the failure cases by bounding the markers to obey constraints of the human body. We speculate that this subdues the effect of LSTM memory on the output. It is also interesting to note that certain type of constraints are prioritized by the optimal models than others. One direction we wanted to explore was incorporating constraint-enforcing layers into the neural network [8] [7] and compare its performance against penalty terms in the loss function. We would have also liked to explore motion-specific data augmentations to help train the model against failure cases. We were also interested in exploring the use of output markers and physics principles to determine contact surfaces. Further, it is also worth investigating how exactly the penalty terms counteract LSTMs memory (if at all) and if there is a trade-off between the two. It is also worth exploring to improve the penalty terms to successfully resolve cases which have larger magnitudes of velocity/force in one direction.

7 Contributions

We split the coding (https://github.com/Kumbong/cs230-project) and writing equally. All three members worked on conceptualizing the loss terms and verifying their validity. All members worked equally on the OLC model. Aditya

worked on incorporating the IOC terms, Suguna worked on the OAC terms and Hermann worked on the hyper parameter tuning. All members contributed equally on testing and resolving bugs in our code. We would like to thank Dr. Antoine Falisse for his assistance.

References

[1] Scott D Uhlrich, Antoine Falisse, Łukasz Kidziński, Julie Muccini, Michael Ko, Akshay S Chaudhari, Jennifer L Hicks, and Scott L Delp. Opencap: 3d human movement dynamics from smartphone videos. *bioRxiv*, 2022.

[2] Joumana Medlej. Human anatomy fundamentals: Flexibility and joint limitations. *Design & Illustration Tutorials*, 2014.

[3] Pablo Márquez-Neila, Mathieu Salzmann, and Pascal Fua. Imposing hard constraints on deep networks: Promises and limitations. *arXiv preprint arXiv:1706.02025*, 2017.

[4] Ruotong Li, Weixin Si, Michael Weinmann, and Reinhard Klein. Constraint-based optimized human skeleton extraction from single-depth camera. *Sensors*, 19(11):2604, 2019.

[5] Russell Stewart and Stefano Ermon. Label-free supervision of neural networks with physics and domain knowledge. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[6] Samuel J Raymond and David B Camarillo. Applying physics-based loss functions to neural networks for improved generalizability in mechanics problems. *arXiv preprint arXiv:2105.00075*, 2021.

[7] Charles Sutton, Andrew McCallum, et al. An introduction to conditional random fields. *Foundations and Trends® in Machine Learning*, 4(4):267–373, 2012.

[8] Tao Li and Vivek Srikumar. Augmenting neural networks with first-order logic. *arXiv preprint arXiv:1906.06298*, 2019.

[9] Tom Beucler, Michael Pritchard, Stephan Rasp, Jordan Ott, Pierre Baldi, and Pierre Gentine. Enforcing analytic constraints in neural networks emulating physical systems. *Physical Review Letters*, 126(9):098302, 2021.

[10] Yatin Nandwani, Abhishek Pathak, and Parag Singla. A primal-dual formulation for deep learning with constraints. *Advances in Neural Information Processing Systems*, 32, 2019.

[11] Benjamin Lindemann, Timo Müller, Hannes Vietz, Nasser Jazdi, and Michael Weyrich. A survey on long short-term memory networks for time series prediction. *Procedia CIRP*, 99:650–655, 2021.

[12] Yingbin Bai, Erkun Yang, Bo Han, Yanhua Yang, Jiatong Li, Yinian Mao, Gang Niu, and Tongliang Liu. Understanding and improving early stopping for learning with noisy labels, 2021.