Predicting 3D Pose

A method to predict the 3D pose of Drosophila melanogaster from the 2D pose

Marco Pietro Abrate

Supervisor: Semih Gunel

Project type: semester project

15/08/2019

Contents

Abstract

DeepFly3D [3] is a computational pipeline that uses seven cameras for inferring the 3D pose of tethered adult Drosophila melanogaster. The cameras are used by the software to triangulate the 3D position of the flies' joints. However, they are positioned all around the platform where the fly stands, occupying the space that could be used for other tools (e.g. lateral lasers for optogenetic stimulation of neurons). A scheme of the setup is shown in Figure 1. The aim of this project is to reduce the number of cameras mounted on the platform that are used to predict 3D pose of D. melanogaster. The main approach to tackle the problem is to introduce a system that, given the 2D joints location of the limbs of the fly, predicts 3D position without the need of triangulation. This reduces the number of required cameras from seven to two. These two cameras would be located on either side of the insect, left and right. The scheme of the new setup is shown in Figure 2. To reach the goal, at first, a relatively simple deep feed forward network is implemented, adapting the one used by Matrinez et al. [10] to predict 3D human pose on the Human3.6M dataset. Secondly, the output data of DeepFly3D (i.e. 2D and 3D joints location) is modified to fit the requirements of the network. Lastly, different preprocessing approaches are carried out, in order to achieve a better performance. The implemented network, trained on 85,424 frames and tested on 18,881 frames, reaches an average test error of 0.11 millimeter per joint. In particular, the average error per joint is 0.04 mm on the x-axis, 0.02 mm on the y-axis and 0.09 mm on the z-axis. Since the error is higher than the one obtained by triangulation with seven cameras (which sits around 0.05 mm), the approach does not result as effective as triangulation. However, this model performs similarly to triangulation with five cameras. Hence, reducing the number of cameras from seven to five and apply triangulation is as successful as using the deep network with two cameras. There is plenty of space for improvements: the amount of data used in this project is small with respect to the Human3.6M dataset; a different network can be trained and used for different scenarios (forward/backward walking, grooming) as in [10] and for different flies (wildtype, MDN/aDN CsChrimson); ground truth can be used for training; data from DeepFly3D can be cleaned before usage; a different set of hyperparameters can be tried or a more complex model can be implemented.

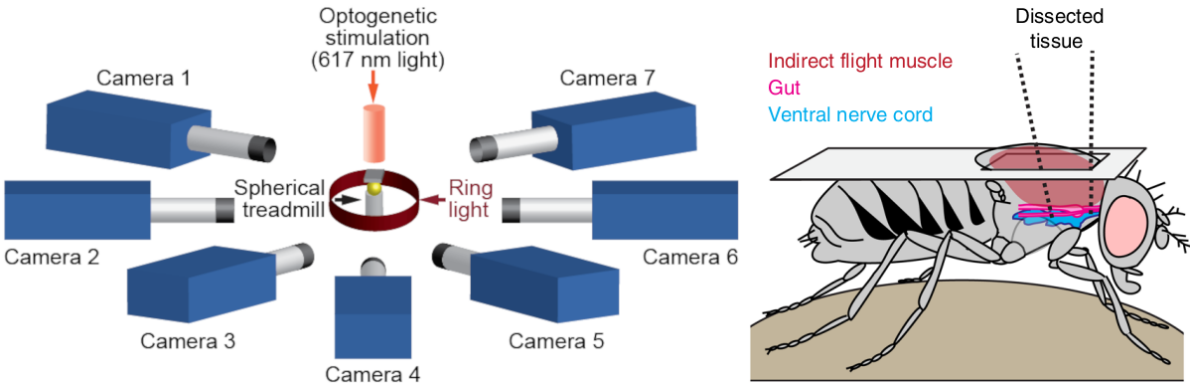


Figure 1: **Left**: The seven cameras setting used to monitor *Drosophila melanogaster* with the software DeepFly3D [3]. **Right**: Zoom of the spherical treadmill, where the fly is mounted and stimulated [2].

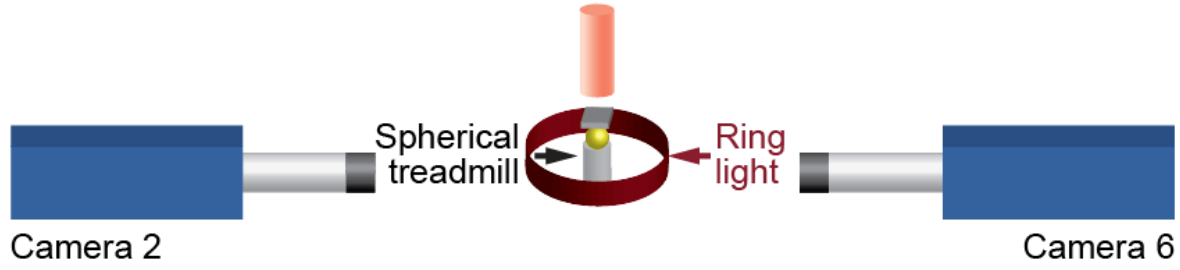


Figure 2: The two cameras setting used to monitor *Drosophila melanogaster* with the newly implemented network.

1. Introduction

The aim of *DeepFly3D* is to have a precise quantification of movements of *Drosophila melanogaster*, while mounted on the platform, where optogenetic stimulation from above is possible (Figure 1). This is essential for understanding how neurons, biomechanics, and the environment influence and give rise to animal behaviors. 3D joint and appendage tracking promises to accelerate the dissection of neural control principles, particularly in the genetically tractable and numerically simple nervous system of *D. melanogaster*. To obtain 3D joints position, the software first infers 2D locations from the seven different angles, using a stacked hourglass network [12]. Then triangulates the seven different 2D images to have a full 3D estimation of the six limbs, the

abdomen and the antenna of the fly. Triangulation is the solution of the optimization problem of finding the best 3D hypothesis, with respect to the Euclidean distance, given several 2D observations. It can be solved using least squares or singular value decomposition. The software, when triangulating, relies on pictorial structures and belief propagation message passing to detect and correct erroneous pose estimates. In this project, only the six limbs are taken into account, since they are the most important part to study *Drosophila* movements.

The input to *DeepFly3D* is video data from seven cameras. As seen in Figure 1, the seven cameras, standing on every side of the platform, occlude the possibility to stimulate the fly from a lateral position. Therefore, reducing the number of cameras would lower the budget needed to build a similar platform, ease the synchronization and speed up the image processing. Hence, the number of processable frames per second would increase considerably. This project brings improvements to the setup by predicting the 3D pose of the limbs using only two separate cameras, one for each side of the insect. In particular, the cameras used by the new method are camera numbers 2 and 6, with respect to Figure 1. This makes it possible to remove the two back cameras (numbers 1 and 7) and the three frontal ones (numbers 3, 4 and 5) from the initial setup. The prediction of 3D pose using only two cameras is achieved thanks to the power of separating pose approximation into two problems: first the estimation of 2D joints (as it is already done by the software *DeepFly3D*) and then the prediction of the 3D pose [10]. The contribution of this project is the design and analysis of a neural network that performs 2D-to-3D pose estimation. The newly implemented model is fast, understandable and easy to reproduce (see Section 3). Figure **??** shows all the tasks performed by *DeepFly3D*. The improvements brought by this project are marked in red.

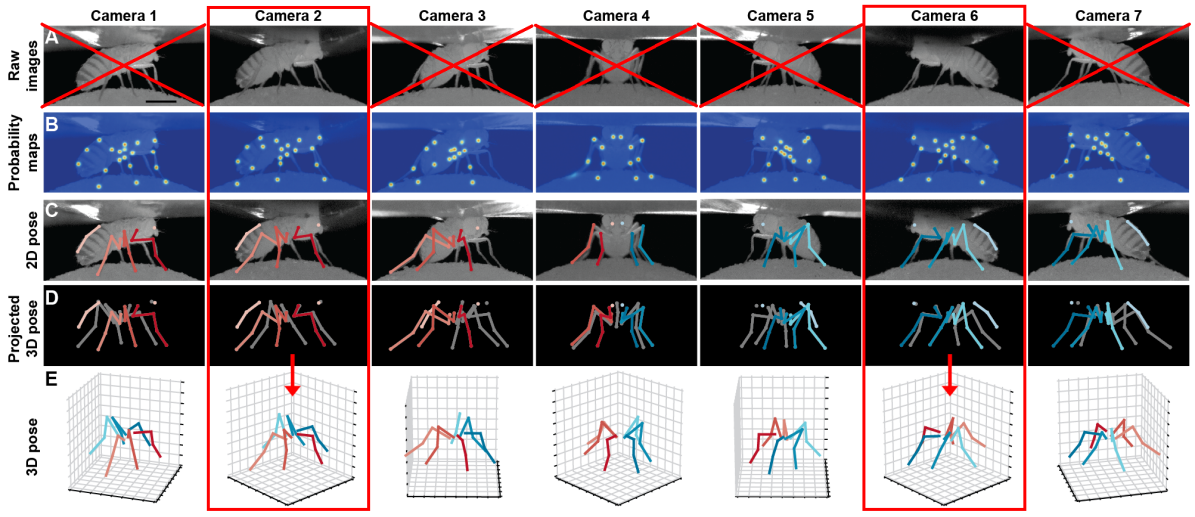


Figure 3: The tasks performed by *DeepFly3D*. Improvements brought by this project are marked in red.

2. State of the art

The CVPR 2019 state of the art in 3D pose estimation is the deep network implemented by Li et al. [8]. They first estimate the 2D joints from the input images using a stacked hourglass network, and then predict the 3D pose with a mixture density network [1], which consists of a feature extractor and a hypotheses generator. Due to time constraints and the complexity of this realization, an easier and faster path is preferred.

The starting point of this project is the deep neural network implemented by Martinez et al. [10], used to predict human 3D pose from 2D joints on the Human3.6M dataset. Their method considerably improves upon the previous best 2D-to-3D pose estimation results, while using a simpler architecture. This result was reached mostly because of an important design change, which is also the driving force of this project. The new method allows for the dramatic reduction of cameras needed in the setup of Figure 1. The idea is to use 2D and 3D points as inputs and outputs, in contrast to previous work that has used raw images [9, 13, 15, 16] or 2D probability distributions [16] as inputs, and 3D probabilities [13], 3D motion parameters [17] or basis pose coefficients and camera parameter estimation [16] as outputs. While 2D detections

carry less information, their low dimensionality makes them very appealing for training deep neural networks.

3. Methods

The goal is to estimate joint locations in 3-dimensional space given a 2-dimensional input. More formally, the input is a series of 2D points $\mathbf{x} \in \mathbb{R}^{2n}$, and the output is a series of points in 3D space $\mathbf{y} \in \mathbb{R}^{3n}$. The aim is to learn a function $f^* : \mathbb{R}^{2n} \rightarrow \mathbb{R}^{3n}$ that minimizes the prediction error over a dataset of N frames:

$$f^* = \min_f \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}(f(\mathbf{x}_i) - \mathbf{y}_i) \tag{1}$$

In practice, $\mathbf{x}_i$ is obtained from the output of the *DeepFly3D* stacked hourglass network, and is considered as ground truth. $\mathbf{y}_i$ is the triangulation result of the seven 2D poses, as pointed out above, n is the number of joints to predict and $\mathcal{L}$ is the Euclidean distance. The aim is to train a model able to estimate 3D poses $\mathbf{y}_i$ using 2D inputs $\mathbf{x}_i$. The network is trained to predict one side of the fly. Once the network is trained, it can be used also for the other side of the insect. To predict all six limbs, two cameras are necessary. The focus is on systems where f^* is a deep neural network.

3.1. Network design

The model is based on a simple, deep, multilayer neural network with batch normalization [6], dropout [14], Rectified Linear Units (RELUs) [11] and residual connections [4]. Figure 4 presents the scheme of the network. Two extra linear layers are not depicted in the figure: one applied to the input, which increases its dimensionality to 1024, and one applied before the final prediction, that produces outputs of size $3n$. In the experiments, 2 residual blocks are added. This means that there are 6 linear layers in total, and the model contains between 4 and 5 million trainable parameters. Hereafter, the building blocks of the implemented network are expanded.

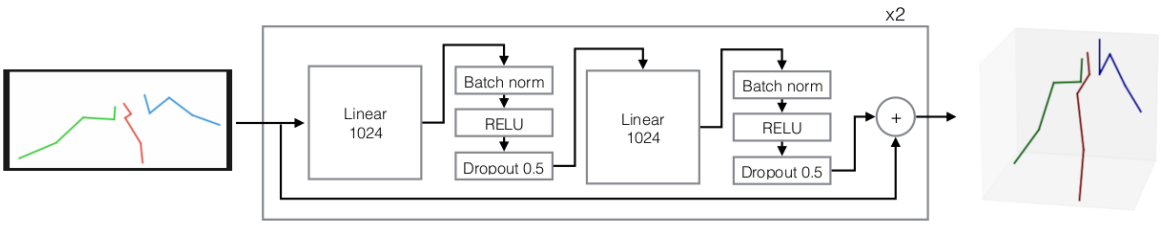


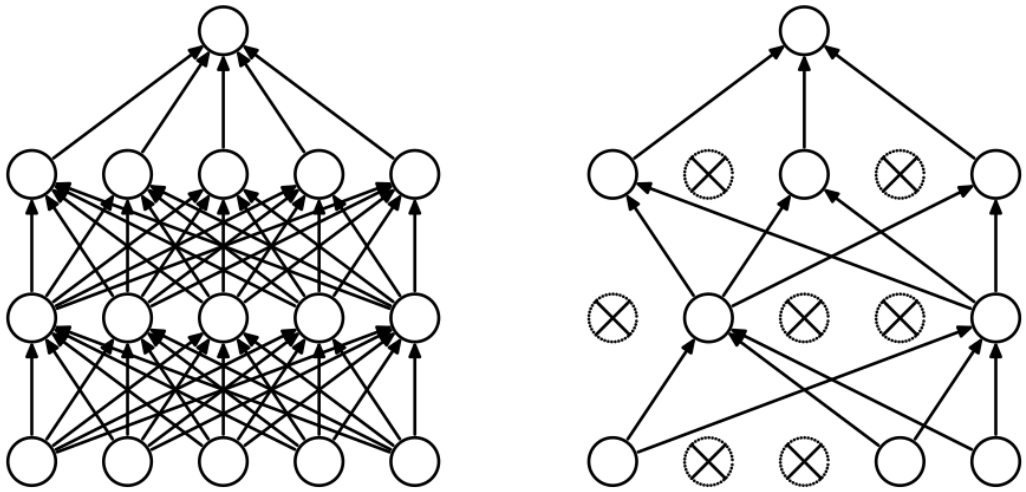
Figure 4: Scheme of the system used in the project. The building block is a linear layer, followed by batch normalization, drop out and a RELU activation. This is repeated twice, and the two blocks are wrapped in a residual connection. The outer block is repeated twice.

- **Batch normalization:** This allows one to use much higher learning rates and be less careful about the initialization parameters. This method takes a step towards reducing internal covariate shift, and in doing so dramatically accelerates the training [6]. The technique consists in whitening each scalar feature independently, by making it have zero mean and variance of one. For a layer with d-dimensional input $\mathbf{x} = (x^{(1)}, ..., x^{(d)})$, the normalization is

$$\hat{x}^{(k)} = \frac{x^{(k)} - \mathbb{E}[x^{(k)}]}{\sqrt{Var[x^{(k)}]}}$$

- **Dropout:** This prevents overfitting and provides a way of approximately combine exponentially many neural network architectures efficiently. The term refers to temporarily dropping out units (hidden and visible) in a neural network, along with all their incoming and outgoing connections, as shown in Figure 5. In this project, each unit is retained with a fixed probability $p = 0.5$, which seems to be optimal for a wide range of tasks [14]. Training a neural network with dropout can be seen as training a collection of 2^n thinned networks with weight sharing. At test time, a single neural net is used, without dropout. The weights of this network are multiplied by p.

- **Linear RELU-layers:** Since the system is dealing with low-dimensional points as inputs and outputs, computationally cheap linear layers can be used. RELUs are a standard choice to add non-linearities in deep neural networks [11].



(a) Standard Neural Net (b) After applying dropout.

Figure 5: **Left**: A standard neural network with two hidden layers. **Right**: An example of a thinned net produced by applying dropout to the network on the left. Crossed units have been dropped.

- **Residual connection:** Since deeper neural networks are more difficult to train, a residual learning framework is used to ease the training, gain accuracy from increased depth and avoid degradation [4].

- **Max-norm constraint:** This is a constraint on the weights of each layer, so that their maximum norm is less than or equal to 1. This technique, coupled with batch normalization, stabilizes training and improves generalization when the distribution differs between training and test examples [10].

3.2. Data preprocessing

One important aspect of the work carried by Martinez et al. [10] is data preprocessing. Since the dataset used in this project treats insects instead of humans, different approaches must be taken in order to make frames more suitable for the network. Below, the list of preprocessing methods used is presented.

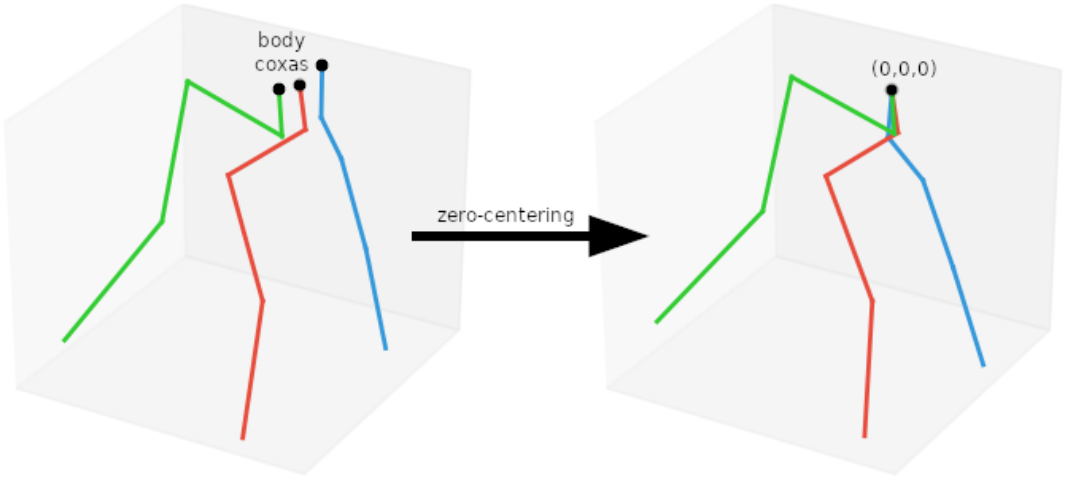


Figure 6: **Left**: Original 3D pose. **Right**: Same 3D pose after applying zero-centering around the three body coxas.

- **Standard normalization:** Subtraction of the mean and division by the standard deviation to the 2D inputs and 3D outputs.

- **Zero-centering:** The prediction of 3D global position is too complex. For humans, the problem is tackled by zero-centering the 3D pose around the hip joint. For *Drosophila melanogaster*, the idea is to zero-center the 3D location of all joints around the three body coxas. This is possible because all three of them are always approximately in the same position and are not important to the study of movements. An example of this method, applied on one frame, is shown in Figure 6. The same approach is taken for 2D images.

- **Camera coordinates:** Since it is unrealistic to expect the network to infer the 3D joint positions in an arbitrary coordinate space, a natural choice of global coordinate is the camera frame. This makes the 2D to 3D problem similar across different cameras, preventing overfitting to a particular global coordinate frame.

$$\mathbf{x}_i^{(proj)} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{x}_i \\ \mathbf{1} \end{bmatrix}$$

Where $\mathbf{x}_i^{(proj)}$ is the projected frame, **R** is the rotation matrix, **t** is the translation vector, $\mathbf{x}_i$ is the initial frame and **1** is a one-dimensional row vector of ones.

Presented below are the methods that were tried out but were not successful:

- **Data augmentation:** The dataset has, for each frame, information about all the seven cameras. Therefore, it was possible to train the network adding information from other cameras, besides the two already considered as a baseline. In particular, images from cameras numbers 1 and 3 (Figure 1) were added to the training dataset. Doing so, even with the above-mentioned camera projection, was not useful for predicting 3D position of the fly with input data coming from cameras number 2 and 6.

- **Procrustes analysis** The dataset is made of recordings from different experiments. Before implementing the camera coordinates, procrustes analysis, which finds optimal translation, rotation and scaling parameters, was used to superimpose body coxas location.

- **Low-pass filter** The movements of the fly are fast and the triangulated 3D joints location oscillate in a small space with high frequency, even when the insect stands still. A low-pass filter was used to smooth the dataset inputs and outputs.

3.3. Training details

The dataset is made of 104,305 frames, coming from the recording of 116 experiments performed on 25 different flies. The data is then separated into 85,424 frames (around 82% of the dataset) for training and 18,881 frames (around 18% of the dataset) for testing. The list of files used for training and testing can be found in Appendix A. The frames in the test set are coming from experiments performed on flies which are not part of the training set, in order to test the model on a more general sample and avoid overfitting.

The network is trained for 200 epochs using Adam [7], a starting learning rate of

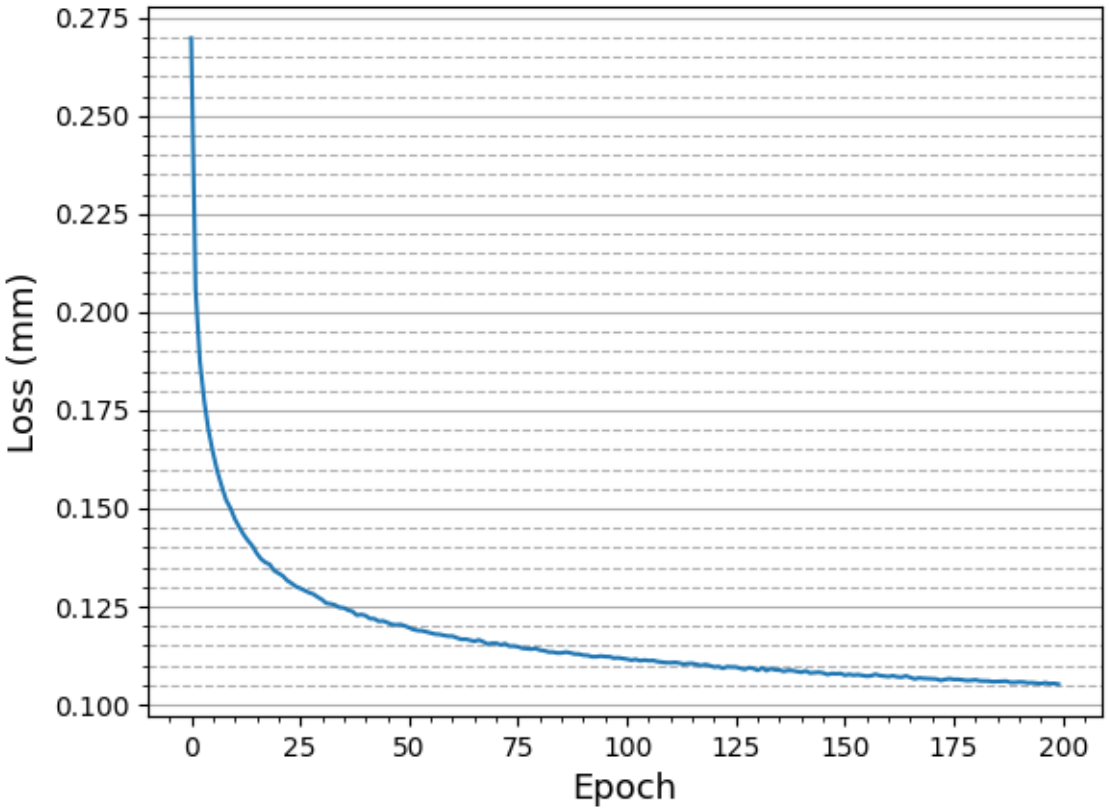


Figure 7: Trend of the loss function, which measures the average distance between the predicted joints and the ground truth, with respect to the epoch.

0.001 and exponential decay, the mini-batches size is set to 64. Initially, the size of the linear layers are calculated using Kaiming initialization [5]. The code, written in Python, uses the Tensorflow library. Around ??? milliseconds are needed to perform a forward+backward pass and around ??? ms for a forward pass on a ??? GPU.

4. Results

After training for 200 epochs, the loss, which measures the average distance between the predicted joints and the ground truth, reaches a value of 0.11 mm. Figure 7 shows the trend of the function for the entire training process. The training results effective, as suggested by the decreasing behavior of the function. However, the outcome is not as good as expected. The triangulation error using seven cameras is around 0.04

Errors limb 1 (mm)	Joint 1	Joint 2	Joint 3	Joint 4	Average
Triangulation 7 cameras	0.03	0.03	0.04	0.07	0.04
Triangulation 5 cameras	0.07	0.08	0.09	0.18	0.11
Deep neural network	0.05	0.07	0.11	0.18	0.10

Table 1: Prediction errors for limb number 1 (frontal limb) for triangulation methods and the deep network.

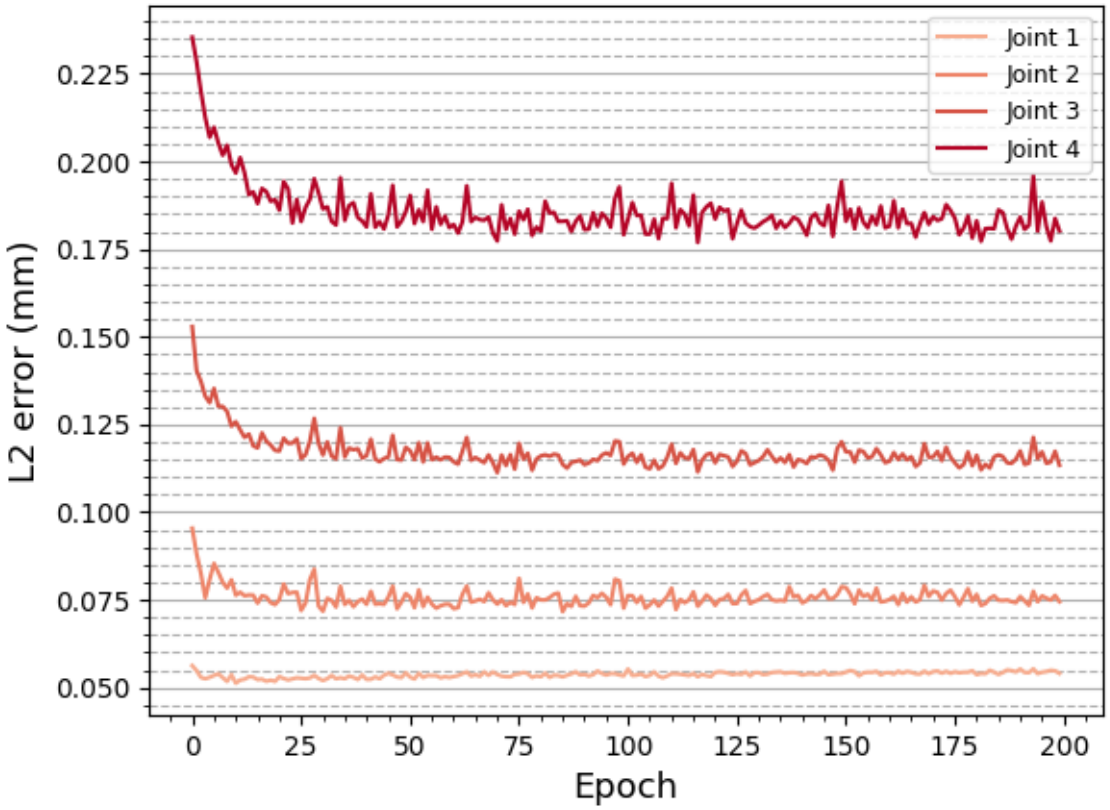


Table 2: Trend of the prediction errors for limb number 1 (frontal limb), for each joint, with respect to the epoch.

mm, which is considerably lower than the prediction error of this network. On the other hand, the triangulation error using five cameras is close (in the order of 10^{-3} mm) to the one of this model. This means that reducing the number of cameras from seven to five and perform triangulation is as effective as reducing the cameras from seven to two and use this deep neural network.

In Table 5, a more detailed description of the prediction errors and a comparison with the triangulation method is presented, with respect to the first limb (frontal limb).

The training trend of the same errors is shown in Figure 2. The limbs are numbered starting from the frontal one. The joints are numbered starting from the body coxa (which is set to 0). The table shows that the further away the joint is from the body coxa, the bigger the error in the predicted joint. This is an expected behavior because, for further joints, the problem becomes more complex, since more factors are involved in the movement of the final part of the limb. In Appendix B, the errors for all joints are presented. As expected, the network easily predicts x and y coordinates. The challenging part for the model (and for a possible human observer) is to predict the depth of the joints, as shown in Figure 8 (Appendix B).

5. Discussion

The model is far from being perfect. First of all, comparing the training performed in this project and the one carried out by Martinez et al. [10] with a similar network on the Human3.6M database, it is clear that the amount of data greatly differs. They used 1,559,744 frames, which is roughly 20 times more frames than what was used in this project. Increasing the quantity of data would help the network to generalize the model for predicting 3D pose. In addition, referencing the same paper, it is useful sometimes to separate the data with respect to different scenarios. In the case of flies, the grooming scenario is much different from a forward/backward walking scenario. Thus, it might be useful to train distinct networks. The same procedure can be applied for different type of flies (wildtype, MDN/aDN CsChrimson).

The input data of the deep model is coming from the 2D predictions of the stacked hourglass network implemented in *DeepFly3D*, which is a source of error. In [10], training on ground truth, instead of using a state of the art stacked hourglass network, decreases the error by 33%. The output data (3D pose) of the network, considered as ground truth during training, is also source of error, because it is generated via triangulation, which is not perfect. In particular, the average error is around 0.04 mm. Theoretically, having access to the ground truth for both input and output, the network

would reach an average error of around 0.05 mm, much closer to the triangulation (with seven cameras) error. If annotation is not possible, a more exhaustive preprocessing of the data is necessary. Each training frame can be discarded if the output of the stacked hourglass is clearly wrong (this would require a human annotator or a sophisticated outlier detector) or the 3D pose mismatch even only one of the 2D images coming from the seven cameras. Unfortunately, this would reduce even more the number of frames available for training.

In relation to the network architecture, an interesting direction to follow would be to use multiple samples from the 2D stacked hourglass heatmaps to estimate an expected gradient. Finally, it is possible that further research into network design could lead to better results on 2D-to-3D systems.

This project can be considered as a first step towards 3D pose prediction from 2D pose within the *Drosophila melanogaster* area of research. The results show that it is possible to think that in future work the method might perform as well as triangulation with seven cameras. Furthermore, the same network could potentially be adjusted to work with different type of insects and animals.

References

[1] Christopher M. Bishop. *Mixture density networks*. Tech. rep. 1994.

[2] Chin-Lin Chen et al. “Imaging neural activity in the ventral nerve cord of behaving adult Drosophila”. In: *Nature communications* 9.1 (2018), p. 4390.

[3] Semih Günel et al. “DeepFly3D: A deep learning-based approach for 3D limb and appendage tracking in tethered, adult Drosophila”. In: *bioRxiv* (2019), p. 640375.

[4] Kaiming He et al. “Deep residual learning for image recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[5] Kaiming He et al. “Delving deep into rectifiers: Surpassing human-level performance on imagenet classification”. In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1026–1034.

[6] Sergey Ioffe and Christian Szegedy. “Batch normalization: Accelerating deep network training by reducing internal covariate shift”. In: *arXiv preprint arXiv:1502.03167* (2015).

[7] Diederik P Kingma and Jimmy Ba. “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980* (2014).

[8] Chen Li and Gim Hee Lee. “Generating Multiple Hypotheses for 3D Human Pose Estimation with Mixture Density Network”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 9887–9895.

[9] Sijin Li, Weichen Zhang, and Antoni B Chan. “Maximum-margin structured learning with deep networks for 3d human pose estimation”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2015, pp. 2848–2856.

[10] Julieta Martinez et al. “A Simple yet Effective Baseline for 3D Human Pose Estimation”. In: *The IEEE International Conference on Computer Vision (ICCV)*. Oct. 2017.

[11] Vinod Nair and Geoffrey E Hinton. “Rectified linear units improve restricted boltzmann machines”. In: *Proceedings of the 27th international conference on machine learning (ICML-10)*. 2010, pp. 807–814.

[12] Alejandro Newell, Kaiyu Yang, and Jia Deng. “Stacked hourglass networks for human pose estimation”. In: *European conference on computer vision*. Springer. 2016, pp. 483–499.

[13] Georgios Pavlakos et al. “Coarse-To-Fine Volumetric Prediction for Single-Image 3D Human Pose”. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. July 2017.

[14] Nitish Srivastava et al. “Dropout: a simple way to prevent neural networks from overfitting”. In: *The journal of machine learning research* 15.1 (2014), pp. 1929–1958.

[15] Xiaowei Zhou et al. “Sparse representation for 3D shape estimation: A convex relaxation approach”. In: *IEEE transactions on pattern analysis and machine intelligence* 39.8 (2016), pp. 1648–1661.

[16] Xiaowei Zhou et al. “Sparseness meets deepness: 3D human pose estimation from monocular video”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 4966–4975.

[17] Xingyi Zhou et al. “Deep kinematic pose regression”. In: *European Conference on Computer Vision*. Springer. 2016, pp. 186–201.

A. Dataset files

Training files:

pose_result__data_paper_180919_MDN_CsCh_Fly4_001_SG1_behData_images.pkl
pose_result__data_paper_180919_MDN_CsCh_Fly4_002_SG1_behData_images.pkl
pose_result__data_paper_180919_MDN_CsCh_Fly4_003_SG1_behData_images.pkl
pose_result__data_paper_180919_MDN_CsCh_Fly4_004_SG1_behData_images.pkl
pose_result__data_paper_180919_MDN_CsCh_Fly4_005_SG1_behData_images.pkl
pose_result__data_paper_180919_MDN_CsCh_Fly5_001_SG1_behData_images.pkl
pose_result__data_paper_180919_MDN_CsCh_Fly5_002_SG1_behData_images.pkl
pose_result__data_paper_180919_MDN_CsCh_Fly5_003_SG1_behData_images.pkl
pose_result__data_paper_180919_MDN_CsCh_Fly5_004_SG1_behData_images.pkl
pose_result__data_paper_180919_MDN_CsCh_Fly5_005_SG1_behData_images.pkl
pose_result__data_paper_180919_MDN_CsCh_Fly6_001_SG1_behData_images.pkl
pose_result__data_paper_180919_MDN_CsCh_Fly6_002_SG1_behData_images.pkl
pose_result__data_paper_180919_MDN_CsCh_Fly6_003_SG1_behData_images.pkl
pose_result__data_paper_180919_MDN_CsCh_Fly6_004_SG1_behData_images.pkl
pose_result__data_paper_180919_MDN_CsCh_Fly6_005_SG1_behData_images.pkl
pose_result__data_paper_180919_MDN_CsCh_Fly7_001_SG1_behData_images.pkl
pose_result__data_paper_180919_MDN_PR_Fly5_001_SG1_behData_images.pkl
pose_result__data_paper_180919_MDN_PR_Fly5_003_SG1_behData_images.pkl
pose_result__data_paper_180919_MDN_PR_Fly5_004_SG1_behData_images.pkl
pose_result__data_paper_180919_aDN_CsCh_Fly1_001_SG1_behData_images.pkl
pose_result__data_paper_180919_aDN_CsCh_Fly1_002_SG1_behData_images.pkl
pose_result__data_paper_180919_aDN_CsCh_Fly1_003_SG1_behData_images.pkl
pose_result__data_paper_180919_aDN_CsCh_Fly1_004_SG1_behData_images.pkl
pose_result__data_paper_180919_aDN_CsCh_Fly1_005_SG1_behData_images.pkl
pose_result__data_paper_180920_MDN_CsCh_Fly10_001_SG1_behData_images.pkl
pose_result__data_paper_180920_MDN_CsCh_Fly8_001_SG1_behData_images.pkl
pose_result__data_paper_180920_MDN_CsCh_Fly8_002_SG1_behData_images.pkl
pose_result__data_paper_180920_MDN_CsCh_Fly8_003_SG1_behData_images.pkl
pose_result__data_paper_180920_MDN_CsCh_Fly8_004_SG1_behData_images.pkl
pose_result__data_paper_180920_MDN_CsCh_Fly8_005_SG1_behData_images.pkl
pose_result__data_paper_180920_MDN_CsCh_Fly9_001_SG1_behData_images.pkl
pose_result__data_paper_180920_MDN_CsCh_Fly9_002_SG1_behData_images.pkl
pose_result__data_paper_180920_MDN_CsCh_Fly9_003_SG1_behData_images.pkl
pose_result__data_paper_180920_MDN_CsCh_Fly9_004_SG1_behData_images.pkl
pose_result__data_paper_180920_MDN_CsCh_Fly9_005_SG1_behData_images.pkl
pose_result__data_paper_180920_MDN_PR_Fly9_001_SG1_behData_images.pkl
pose_result__data_paper_180920_MDN_PR_Fly9_002_SG1_behData_images.pkl
pose_result__data_paper_180920_MDN_PR_Fly9_003_SG1_behData_images.pkl
pose_result__data_paper_180920_MDN_PR_Fly9_004_SG1_behData_images.pkl
pose_result__data_paper_180920_MDN_PR_Fly9_005_SG1_behData_images.pkl

pose_result__data_paper_180920_aDN_CsCh_Fly2_001_SG1_behData_images.pkl
pose_result__data_paper_180920_aDN_CsCh_Fly2_002_SG1_behData_images.pkl
pose_result__data_paper_180920_aDN_CsCh_Fly2_003_SG1_behData_images.pkl
pose_result__data_paper_180920_aDN_CsCh_Fly2_004_SG1_behData_images.pkl
pose_result__data_paper_180920_aDN_CsCh_Fly2_005_SG1_behData_images.pkl
pose_result__data_paper_180920_aDN_PR_Fly2_001_SG1_behData_images.pkl
pose_result__data_paper_180920_aDN_PR_Fly2_002_SG1_behData_images.pkl
pose_result__data_paper_180920_aDN_PR_Fly2_003_SG1_behData_images.pkl
pose_result__data_paper_180920_aDN_PR_Fly2_004_SG1_behData_images.pkl
pose_result__data_paper_180920_aDN_PR_Fly2_005_SG1_behData_images.pkl
pose_result__data_paper_180921_aDN_CsCh_Fly3_001_SG1_behData_images.pkl
pose_result__data_paper_180921_aDN_CsCh_Fly4_001_SG1_behData_images.pkl
pose_result__data_paper_180921_aDN_CsCh_Fly4_002_SG1_behData_images.pkl
pose_result__data_paper_180921_aDN_CsCh_Fly4_003_SG1_behData_images.pkl
pose_result__data_paper_180921_aDN_CsCh_Fly4_005_SG1_behData_images.pkl
pose_result__data_paper_180921_aDN_CsCh_Fly5_001_SG1_behData_images.pkl
pose_result__data_paper_180921_aDN_CsCh_Fly5_002_SG1_behData_images.pkl
pose_result__data_paper_180921_aDN_CsCh_Fly5_003_SG1_behData_images.pkl
pose_result__data_paper_180921_aDN_CsCh_Fly5_004_SG1_behData_images.pkl
pose_result__data_paper_180921_aDN_CsCh_Fly5_005_SG1_behData_images.pkl
pose_result__data_paper_180921_aDN_CsCh_Fly6_001_SG1_behData_images.pkl
pose_result__data_paper_180921_aDN_CsCh_Fly6_002_SG1_behData_images.pkl
pose_result__data_paper_180921_aDN_CsCh_Fly6_003_SG1_behData_images.pkl
pose_result__data_paper_180921_aDN_CsCh_Fly6_004_SG1_behData_images.pkl
pose_result__data_paper_180921_aDN_CsCh_Fly6_005_SG1_behData_images.pkl
pose_result__data_paper_180921_aDN_CsCh_Fly7_001_SG1_behData_images.pkl
pose_result__data_paper_180921_aDN_CsCh_Fly7_002_SG1_behData_images.pkl
pose_result__data_paper_180921_aDN_CsCh_Fly7_003_SG1_behData_images.pkl
pose_result__data_paper_180921_aDN_CsCh_Fly7_004_SG1_behData_images.pkl
pose_result__data_paper_180921_aDN_CsCh_Fly7_005_SG1_behData_images.pkl
pose_result__data_paper_180921_aDN_PR_Fly4_001_SG1_behData_images.pkl
pose_result__data_paper_180921_aDN_PR_Fly4_002_SG1_behData_images.pkl
pose_result__data_paper_180921_aDN_PR_Fly4_003_SG1_behData_images.pkl
pose_result__data_paper_180921_aDN_PR_Fly4_004_SG1_behData_images.pkl
pose_result__data_paper_180921_aDN_PR_Fly4_005_SG1_behData_images.pkl
pose_result__data_paper_180921_aDN_PR_Fly5_001_SG1_behData_images.pkl
pose_result__data_paper_180921_aDN_PR_Fly5_002_SG1_behData_images.pkl
pose_result__data_paper_180921_aDN_PR_Fly5_003_SG1_behData_images.pkl
pose_result__data_paper_180921_aDN_PR_Fly5_004_SG1_behData_images.pkl
pose_result__data_paper_180921_aDN_PR_Fly5_005_SG1_behData_images.pkl
pose_result__data_paper_180921_aDN_PR_Fly7_001_SG1_behData_images.pkl
pose_result__data_paper_180921_aDN_PR_Fly7_002_SG1_behData_images.pkl
pose_result__data_paper_180921_aDN_PR_Fly7_003_SG1_behData_images.pkl
pose_result__data_paper_180921_aDN_PR_Fly7_004_SG1_behData_images.pkl

pose_result__data_paper_180921_aDN_PR_Fly7_005_SG1_behData_images.pkl
pose_result__data_paper_180921_aDN_PR_Fly8_001_SG1_behData_images.pkl
pose_result__data_paper_180921_aDN_PR_Fly8_002_SG1_behData_images.pkl
pose_result__data_paper_180921_aDN_PR_Fly8_003_SG1_behData_images.pkl
pose_result__data_paper_180921_aDN_PR_Fly8_004_SG1_behData_images.pkl
pose_result__data_paper_180921_aDN_PR_Fly8_005_SG1_behData_images.pkl
pose_result__data_paper_180921_aDN_PR_Fly9_001_SG1_behData_images.pkl
pose_result__data_paper_180921_aDN_PR_Fly9_002_SG1_behData_images.pkl
pose_result__data_paper_180921_aDN_PR_Fly9_003_SG1_behData_images.pkl
pose_result__data_paper_180921_aDN_PR_Fly9_004_SG1_behData_images.pkl
pose_result__data_paper_180921_aDN_PR_Fly9_005_SG1_behData_images.pkl

Testing files:

pose_result__data_paper_180918_MDN_CsCh_Fly1_001_SG1_behData_images.pkl
pose_result__data_paper_180918_MDN_CsCh_Fly1_002_SG1_behData_images.pkl
pose_result__data_paper_180918_MDN_CsCh_Fly1_003_SG1_behData_images.pkl
pose_result__data_paper_180918_MDN_CsCh_Fly1_004_SG1_behData_images.pkl
pose_result__data_paper_180918_MDN_CsCh_Fly1_005_SG1_behData_images.pkl
pose_result__data_paper_180918_MDN_CsCh_Fly2_001_SG1_behData_images.pkl
pose_result__data_paper_180918_MDN_CsCh_Fly2_002_SG1_behData_images.pkl
pose_result__data_paper_180918_MDN_CsCh_Fly2_003_SG1_behData_images.pkl
pose_result__data_paper_180918_MDN_CsCh_Fly2_004_SG1_behData_images.pkl
pose_result__data_paper_180918_MDN_CsCh_Fly2_005_SG1_behData_images.pkl
pose_result__data_paper_180918_MDN_CsCh_Fly3_001_SG1_behData_images.pkl
pose_result__data_paper_180918_MDN_CsCh_Fly3_002_SG1_behData_images.pkl
pose_result__data_paper_180918_MDN_PR_Fly1_002_SG1_behData_images.pkl
pose_result__data_paper_180918_MDN_PR_Fly1_003_SG1_behData_images.pkl
pose_result__data_paper_180918_MDN_PR_Fly1_004_SG1_behData_images.pkl
pose_result__data_paper_180918_MDN_PR_Fly1_005_SG1_behData_images.pkl
pose_result__data_paper_180918_MDN_PR_Fly2_001_SG1_behData_images.pkl
pose_result__data_paper_180918_MDN_PR_Fly2_002_SG1_behData_images.pkl
pose_result__data_paper_180918_MDN_PR_Fly2_003_SG1_behData_images.pkl
pose_result__data_paper_180918_MDN_PR_Fly2_004_SG1_behData_images.pkl
pose_result__data_paper_180918_MDN_PR_Fly2_005_SG1_behData_images.pkl

B. Prediction errors

Figure 8 shows the trand of the average prediction error coordinate wise. Table **??** and Figure 4 present, respectively, the final value and the trend of the prediction error per joint for limb number 2. Table **??** and Figure 6 present, respectively, the final value and the trend of the prediction error per joint for limb number 3.

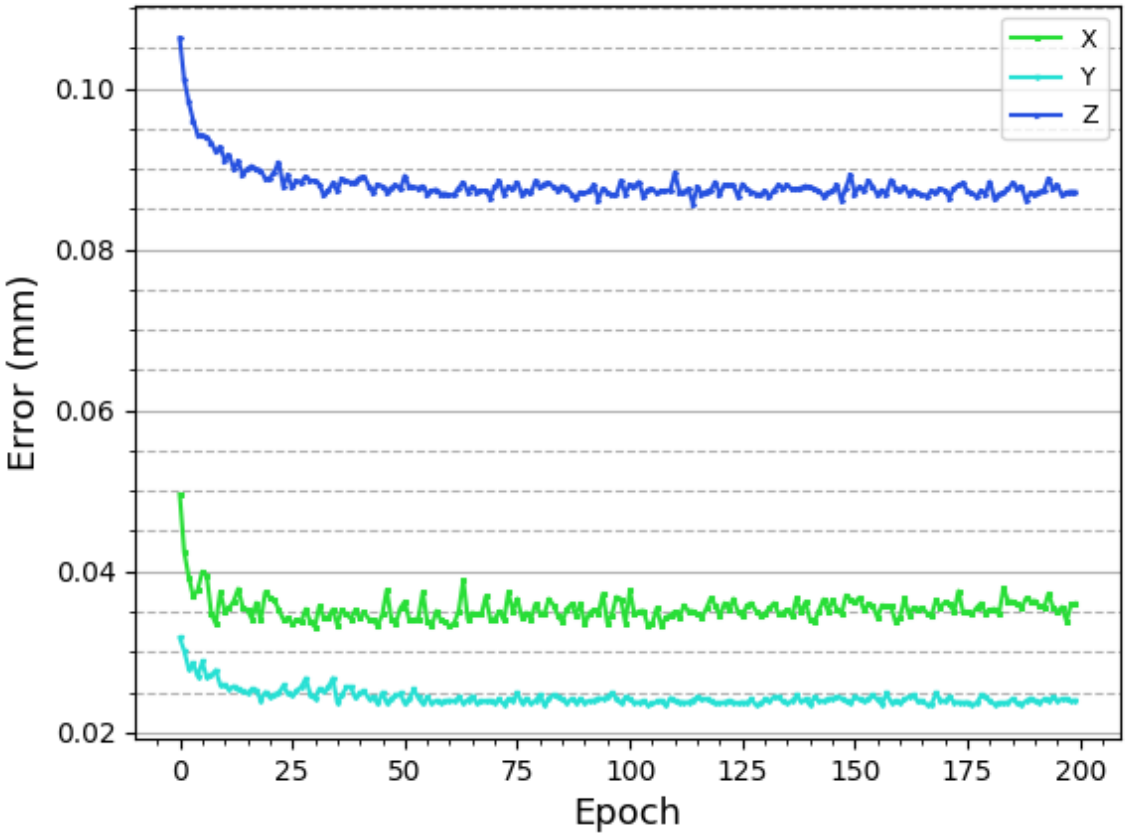


Figure 8: Function of the average error coordinate wise, with respect to the epoch.

Errors limb 2 (mm)	Joint 1	Joint 2	Joint 3	Joint 4	Average
Triangulation 7 cameras	0.03	0.03	0.04	0.07	0.04
Triangulation 5 cameras	0.07	0.08	0.09	0.18	0.11
Deep neural network	0.04	0.06	0.13	0.19	0.11

Table 3: Prediction errors for limb number 2 (central limb) for triangulation methods and the deep network.

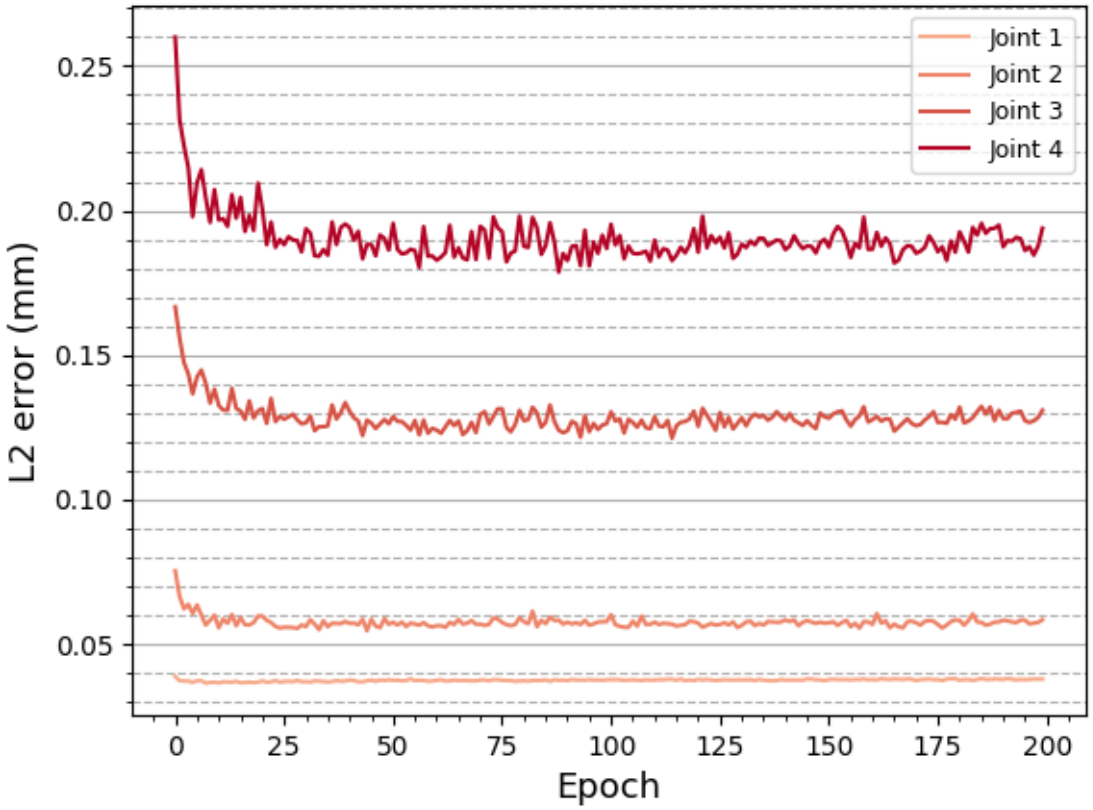


Table 4: Trend of the prediction errors for limb number 2 (central limb), for each joint, with respect to the epoch.

Errors limb 3 (mm)	Joint 1	Joint 2	Joint 3	Joint 4	Average
Triangulation 7 cameras	0.03	0.03	0.04	0.07	0.04
Triangulation 5 cameras	0.07	0.08	0.09	0.18	0.11
Deep neural network	0.04	0.07	0.11	0.22	0.11

Table 5: Prediction errors for limb number 3 (back limb) for triangulation methods and the deep network.

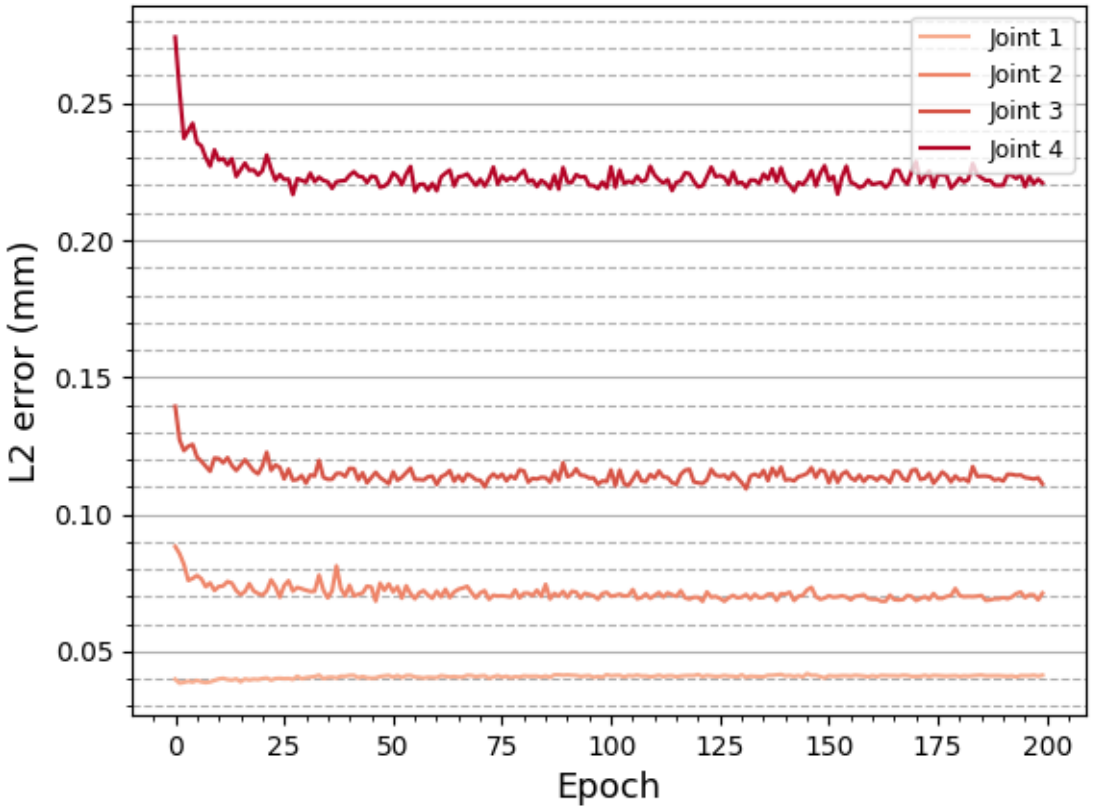


Table 6: Trend of the prediction errors for limb number 3 (back limb), for each joint, with respect to the epoch.