Deep Inverse Kinematics for Human Posture Synthesis and Estimation

Anonymous CVPR submission

Paper ID ****

Abstract

Inverse Kinematics (IK) is a long standing problem in the fields of robotics control and character animation. The main challenge lies in the redundancy, where an infinite number of body configurations may reach the same position of end-effector. Selecting the appropriate one from the large repertoire of candidates is an open question. It is particularly challenging to identify the natural posture for humanoid characters since we are most familiar with human motion and highly sensitive to subtle details. This paper addresses the problem of Inverse Kinematics with deep neural network, trained with so-far the largest human mocap database. We identify the critical temporal correlation between input and output motion frames. Given the challenge of multi-solution in the IK problem, the trained model selects the pose which is most consistent with the pose by the real performer. This consistency is validated by the comparison with the benchmark database. The trained model is adaptable to complex tasks, such as basketball dribbling and ballet dancing, and to characters of different geometrical lengths. We do not assume the knowledge of the accurate limb lengths and skip the requirement of manual setup of joint limits. This allows our method to be directly used in the problems of posture estimation.

1. Introduction

Inverse Kinematics (IK) refers to the challenge of solving the body degrees of freedom, given the location of the end-effectors in the world coordinate. It is a long standing problem in the fields of robotics control and character animation. Robotic arm solves the most common task of point-reach-grab with this technique and the virtual character is visually demonstrated to interact with the environment by constraining the positions of its hands and feet.

However, solving the task of inverse kinematics is non-trivial. The major challenge is the *spatial redundancy* [1], which indicates that there may exist multiple solutions (body configurations) for the same position of end-effector. Although all solutions are numerically correct, some postures may not appear visually natural. This challenge critically affects the naturalness of the synthetic character animation in graphics applications, such as games and films, and increasingly important for social robots in the mixed environment with real humans.

We address the IK problem by modelling the posture distribution with deep neural network. The network model is trained with >2 million postures of real human and produces the mapping between the end-effector position and joint orientation. Existing data-driven methods address the IK problem by utilizing the prior information embedded in latent models, including Gaussian Process Latent Variable Model (GPLVM) [2], Principal Components Analysis (PCA) [3], Multi-variate Gaussian distribution model (MGDMs) [4]. However, these models can only handle databases of moderate sizes. Robotics researchers have proposed the use of deep neural network to solve the IK problem for robot manipulator with training data synthesized from computer simulation [5, 6, 7]. Although there are a few works using deep neural networks for character control [8, 9, 10], we did not find the exact application of neural work for IK problem as our work. This work is, to our best knowledge, the first to train the IK solver with database of the comparable size. More specifically, the contributions of this work are:

- We prioritize the goal of synthesizing natural movements and prefer the posture which is most likely to be performed by real human. This is, to our best knowledge, the first work to utilize such a large-scale database for solving the IK problem. Existing methods use neural network to solve the IK problem (in particularly in the case of robotic arm).
- We identify the critical temporal correlation between input and output motion frames. A short segment of motion sequence is used as network input and improves the prediction accuracy. The motion is stabilized by applying 1D mean-filter to the output joint channel.
- We take advantage of the assumption of constant limb ratio and the use of relative coordinates. This is based

on the observation that the length ratios of femur : tibia and humerus : ulna were remarkably similar (1.21 and 1.22, respectively) and varied little (<7%) between individuals [11]. This removes the effect of limb length, and thus allows the adaptation of various body lengths.

Compared with traditional analytic solutions, our method is capable of avoiding kinematic singularities and impossible postures as can result from an ill-conditioned matrix inversion. We achieve the comparable runtime cost as the analytic solutions and out-perform the iterative solver.

Hierarchical structure, this is based on the observations from two proposed applications: animators normally attach the IK solver separately for arms and legs, and in posture estimation problems, body parts may be occluded and not feasible to predict, using separate solvers offers flexibility.

2. Related Works

Inverse kinematics is a long-standing and important problem which is the basis for resolving the technical challenges such as controlling the end-effector of a humanoid robot in real environment [12] and animating the virtual characters in simulated environment [13]. However, as the complexity of robot or virtual character increases, solving the problem of inverse kinematics is difficult and computationally expensive. This section presents a concise summary of three categories of existing IK techniques. For a comprehensive review, including some meta solutions (Sequential Monte Carlo [14] and Particle-based IK [15] solvers), we kindly refer our readers to a recent survey [16] on existing IK techniques in computer graphics.

Analytic methods attempt to find all possible solutions, given the link lengths, target positions and potential constraints, by solving the trigonometric-related functions [17, 18]. Analytic methods are fast and provide accurate solutions to simple cases, such as 2D plannar with limited number of DOFs. However, it suffers major challenges when extending the domain from 2D to 3D, since the number of possible solutions which satisfy the trigonometric requirements is theoretically unlimited. This is particularly true when solving the body configuration of 3D virtual characters. In contrast to the priorities of stability and accuracy for robotics, visually-perceived naturalness of synthetic motion is the critical concern for graphical applications and it is a nontrivial task.

Numeric methods approach the solution via iteratively minimizing a cost function, which in general indicates the deviation between the current and target postures. Standard techniques include Jacobian, Newton and Heuristic methods. Jacobian-based methods computes the partial derivatives between the global positions of the joint with respect to the angle parameters [19]. It is further divided into subcategories, depending on the specific implementation of Jacobian matrix, including Jacobian transpose [20], Jacobian pseudo-inverse [21], Damped Least Squares [22], Singular value decomposition [23] etc. Newton-based methods find the solution via a second-order approximation of the function using quasi-Newton methods [24, 25]. Heuristic methods include Cyclic Coordinate Descent (CCD) [26] and Forward and Backward Reaching IK (FABRIK) [27]. The numeric methods can be time-consuming before the iterative process converges to an optimal solution so that it may not be suitable for real-time applications. There is no guarantee that the posture is visually natural, or consistent with real human performers although it is numerically correct.

Data-driven solutions take advantage of the large database and use the pre-learned models to infer the most probable posture for a given end-effector position. The database can either be captured from real performers [4] or generated via simulation (particularly popular for robot manipulation) [28]. Researchers interpolate example motions and positions using radial basis function, offering IK control alongside descriptive features such as emotion, difficulty, health [29]. Multi-variate Gaussian distribution model (MGDMs) has been proposed to precisely specify the soft joint constraints of a kinematic skeleton and produce higher accuracy and stability [4]. The IK is formulated as a constrained optimization problem and solved the latent space constructed with the techniques of Principal Components Analysis (PCA) or Probabilistic PCA (PPCA) [30, 31, 32, 3]. Impressive results were achieved with a learned model of human poses based real-time inverse kinematics system for different styles of IK [2]. This system is capable of generating any pose but preferring poses most similar to the space of poses in the training data. It can be applied to interactive character posing, trajectory keyframing, real-time motion capture with missing markers, posing from a 2D image, etc.

Deep neural network arise as a promising solution for the IK problem for constructing the underlying mapping from the global coordinates to the local joint DOFs. For instance, researchers [33] explored the application of a feedforward neural network to solve the IK problem for a three-link planar manipulator. Furthermore, an approach utilizing multi-layered perceptron (MLP) with a back-propagation training algorithm proves to effective in reducing the computing complexity when using the Inverse Geometric Models implementation (IGM) in robotics [6]. However, the gradient-based learning algorithms used by previous researchers can cause a very slow training process, especially for a complex configuration, or a large set of training data. To solve this problem, researchers proposed a learning algorithm called Extreme Learning Machines (ELMs) [7] which randomly choose the input weights and analytically determines the output weights of the single hidden layer feedforward neural networks. Based on this achievement, a

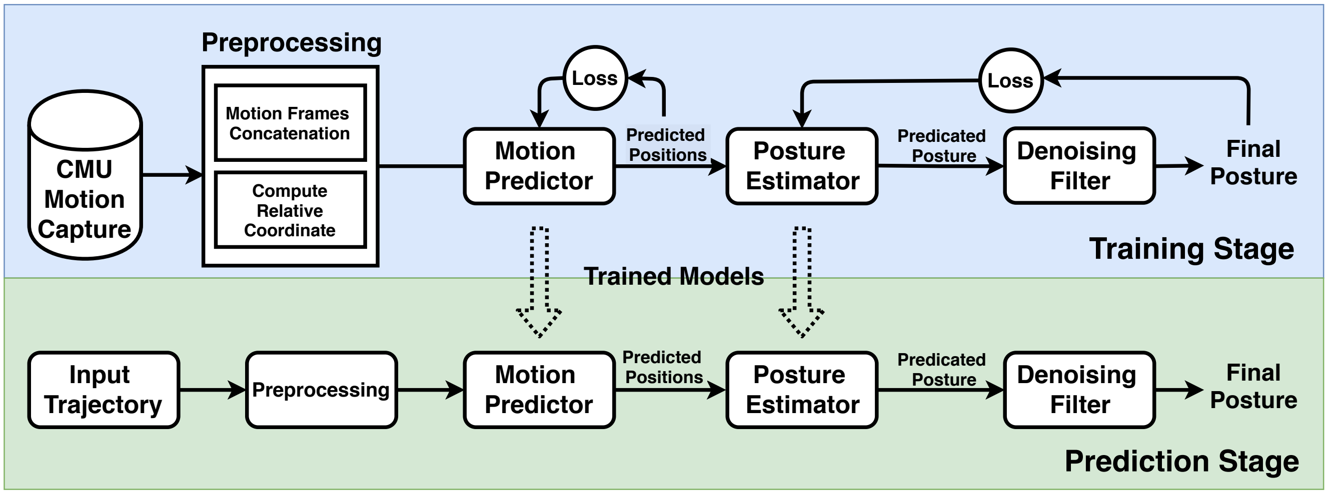


Figure 1: Flowchart of our method.

novel recurrent neural network controller also has been exploited for learning and maintaining multiple solutions of the inverse kinematics [34]. Their work combines ideas of Reservoir Computing [35] and Extreme Learning Machines (ELMs) [36] which alleviate the error in the process of back-propagation of errors and thus improve the precision. Another work used the deep neural network to produce new sequences of movements according to examples by approximating target trajectories [37]. Nevertheless, these methods are based on high-level constraints so that they are not flexible enough to create new poses or satisfy new constraints. Yet, the above study only considers an ideal model assuming the exact configuration of the virtual and real robots. A supervised learning based method is recently proposed to address the manufacturing and assembly errors [38]. By comparing the error of the inverse kinematics function with and without joint misalignments, it can be observed that for neural networks, misalignment does not produce a difference from the result point of view.

3. Methodology

In this section, we present our proposed method for supervised learning of the IK solver. Figure 1 illustrates the pipeline of our method. The key components in our method are Motion Predictor (MP) and Posture Estimator (PE). The former predicts the future positions of end-effectors given the current trajectory, while the latter estimates the body posture based on a sequence of end-effector positions.

We first define the terms to be used in this paper:

- **Skeleton** The skeleton $\mathbf{S}$ defines the hierarchy of the character, with a root joint and its inter-connected joints. The joint distance is specified to construct characters with geometry variations.
- **End-effector** The end-effector $\mathbf{EE}$ is normally referred to the foot and hand of the character, for the purpose of environmental interaction.
- **Posture** A posture $\mathbf{P}$ is vector describing the character body configuration, composing of the global transformation of the root joint and the angle values of joint orientations.
- **Motion** The motion $\mathbf{M}$ is a matrix describing a series of character postures in the temporal domain.
- **Position** The position $\mathbf{PE}$ describes the global position of the $\mathbf{EE}$.
- **Forward Kinematics** refers to the process of computing the positions of end-effectors, in 3D space given the specific character posture.
- **Inverse Kinematics** refers to the process of computing the joint orientations along the body link, when the end-effector is placed at the desired target position.

3.1. Offline Network Learning

3.1.1 Datasets and Preprocessing

We here use the CMU Mocap database to construct the training and testing sets in this problem. The training dataset contains ∼2000 sequences (∼2 million postures) while the testing dataset contains ∼200 sequences (∼200k postures). The positions of the skeleton joints are computed with the technique of forward kinematics. We propose a hierarchical structure which is composed of four separate IK solvers designed specifically for: left/right arms, and left/right legs.

We identified two effective techniques which significantly affect the learning performance:

Temporal Correlation Between Consecutive Frames Instead of using current position of the end-effector as the network input, concatenating multiple frames $[\cdots, X_{t-2*k}, X_{t-k}, X_t, X_{t+k}, X_{t+2*k}, \cdots]$ as network input proves effective in improving the prediction accuracy. This is confirmed in our experiments (see Figure 7 and relevant discussions).

Relative Coordinate for Shoulder-Hand/Hip-Foot We compute the relative vector of shoulder-hand and hip-foot and use these vectors to represent the current position of the end-effector. This is based on the assumption that the ratios of femur : tibia and humerus : ulna were remarkably similar between individuals. This allows the ik solver to adapt to characters with different geometry.

By doing so, we can construct the mapping between the $\mathbf{X}$ and $\mathbf{Y}$:

- $\mathbf{X}$: a vector of N_X to represent the global positions of end-effectors in a sequence of frames. These are computed from the hierarchical skeleton structure and the joint channels.

- $\mathbf{Y}$: a vector of N_Y to represent the angle values of joint orientation. These are directly copied from the corresponding channels of the motion files.

3.1.2 Motion Predictor and Posture Estimator

MP and PE are two main components of our IK solver. We use an identical network structure to model both components. The network is Recurrent Neural Network (RNN), with three layers of LSTM (size of 512), followed with a fully-connected layer.

The input to the MP network is a sequence of **EE** positions in the past K frames. The output of the MP network is a sequence of **EE** positions in the next K frames. The loss function is defined as:

$$L_m = \sum_{i=0}^{K_m} (\mathbf{PE}_i - \mathbf{PE}_i^G) \tag{1}$$

where K_m is the number of concatenated frames, $\mathbf{PE}_i^G$ is the ground truth from the motion capture database.

The input to the PE network is a sequence of **EE** positions including the past K frames and the next K frames. The output of the PE network is the predicted postures including the past K frames and the next K frames. The loss function is defined as:

$$L_p = \sum_{i=0}^{K_p} (\mathbf{P}_i - \mathbf{P}_i^G) \tag{2}$$

where K_p is the number of concatenated frames, $\mathbf{P}_i^G$ is the ground truth from the motion capture database.

Learning Algorithm and Hyperparameters The learning is conducted with the Adam solver, with the hyperparameters as: the batch size = 64, the learning rate = 0.001 and ...

Denoising Filter The output from the PE network is further processed with a mean filter for the purpose of denoising:

$$\mathbf{P} = \frac{1}{N} \sum_{i}^{k} \mathbf{P}_i \tag{3}$$

The improves the smoothness of the predicted posture.

3.2. Posture Synthesis and Estimation

We apply the proposed IK solver to two applications: posture synthesis and estimation.

For the task of online posture synthesis, we automatically generate the full body posture after the user specifies the trajectories of the end-effectors. The trajectory is sampled with a uniform interval of 0.5 second. The trajectory is preprocessed by the two steps: motion frame concatenation and relative coordinate computation. The prediction procedure is shown in Figure 1.

For the task of posture estimation, we focus on specifically using 2D images to estimate 3D human postures. This is challenging problem due to its ill-posed condition. In real-world applications, it is not feasible to have the knowledge of the body geometry information. We only use the 3D positions of the **EE** and their corresponding root joints, and apply our IK solver to estimate the optimal posture for the current body link.

4. Results and Discussions

4.1. Implementation Details

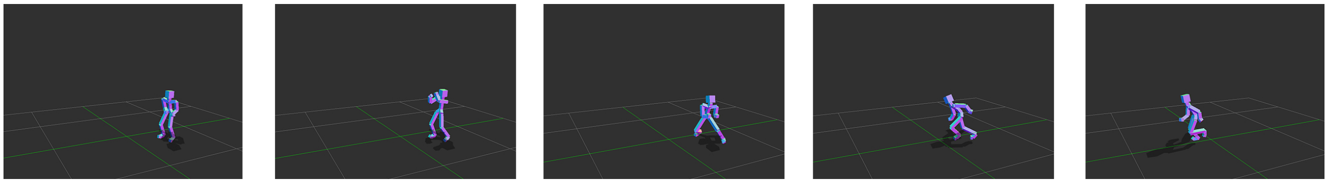
The network is trained and tested on a standard PC with OS: Ubuntu 16.04, CPU: Intel i7 ??, GPU: NVidia Geforce 1080Ti, Memory: 16G. The materials, including source code, training and testing samples, are published on the github repository [1]. The training timecost is ? hours.

4.2. Results on Training and Testing Dataset

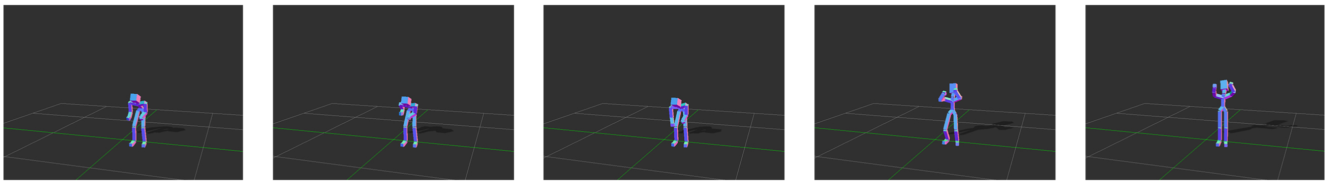
Figure 2 demonstrates the synthetic motion (basketball dribbling, shooting and ballet dancing) from the training data. Note that these examples require dexterous manipulation of object (such as basketball handling) and rich contact with the environment (such as foot support on the ground). Such movements all require high accuracy of the ik solver. In general, the average loss of the training samples is ?.

Figure 3 demonstrates the synthetic motion (getting-up from the ground, basketball dribbling and rolling on the

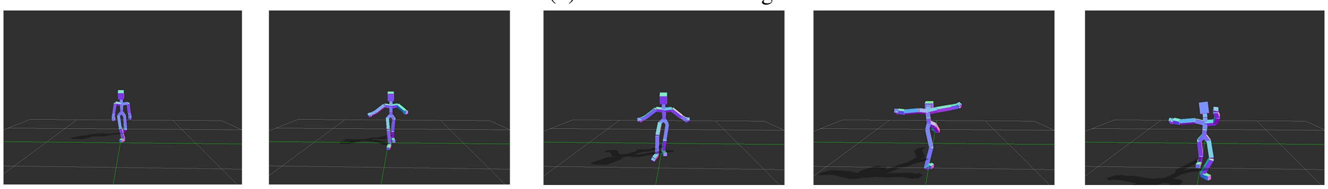
[1] link: https://github.com/uhomelee/DeepInverseKinematicsSolver



(a) basketball dribbling.



(b) basketball shooting.



(c) ballet dancing.

Figure 2: Synthetic motion from training dataset.

ground) from the testing data. Although these examples are not originally included in the training dataset, the ik solver still manages to solve the body configuration and produces natural movement.

4.3. Trajectory Key-framing

One of the direct applications is trajectory key-framing. Users author the trajectory of the end-effectors as a Spline curve, while the IK solver automatically produces the joint angles along the body link for the complete trajectory. We first sample the Spline curve with uniform parametrization, and use a segment of multiple frames as network input.

4.4. Real-time Motion Compression

One of the direct application of our method is motion compression. In the standard type of motion file (such as the BVH format), character motion is stored as the global transformation of the root joint, offset and orientation of the child joint relative to the parent joint. Given the high accuracy of our ik solver, it is sufficient to replace the joint orientation with the end-effector position. This approach could reduce the motion filesize for at least 50%, which is important for data transmission in real-time application.

4.5. Motion Retargetting to Characters with Different Limb Lengths

Since the position of the end-effector is normalized with the limb length, our method could generalize to characters with different limb lengths.

4.6. Motion Synthesis in Contact-rich Environment

Here we demonstrate the capability of our method in synthesizing the natural motion in scenarios involving intensive body-environment contact.

4.7. Comparison with Existing Works

Compare with style-based IK

4.8. User Experiment

We perform user experiment to evaluate our method.

Interaction Animators can use our Maya plugin to complete the keyframe animation. We invite ? animators, ranging from junior (less than 1 year experience) to senir (¿ 10 years experience) to evaluate and rate the interaction friendliness of our plugin.

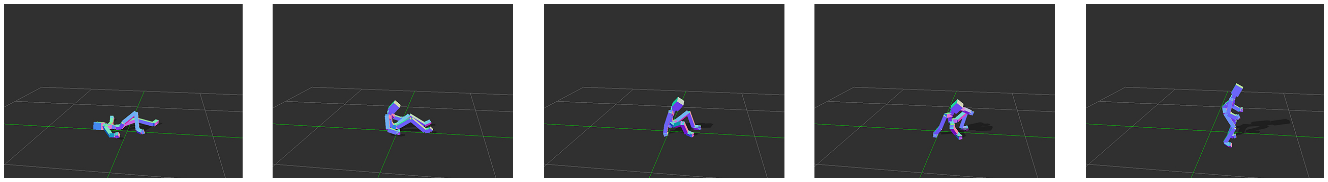
[Screenshot of using Maya plugin]

Naturalness

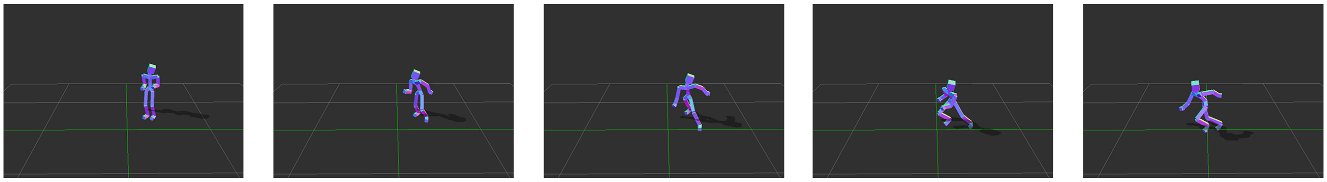
? Participants are invited to evaluate the naturalness of the synthetic motion from our IK solver. ? participants have experience of using 3d character animation (maya), while ? participants have no experience. Participants are provided a side-by-side video comparison of original and synthetic animation, and asked to identify the synthetic one.

4.9. Smoothness

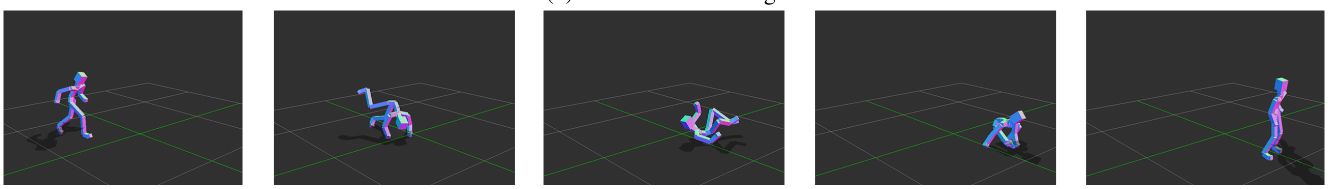
Smoothness of the synthetic motion is a key measure for IK techniques [16], particularly for iterative methods involving the convergence of the algorithm. In general, the



(a) getting up from the ground.

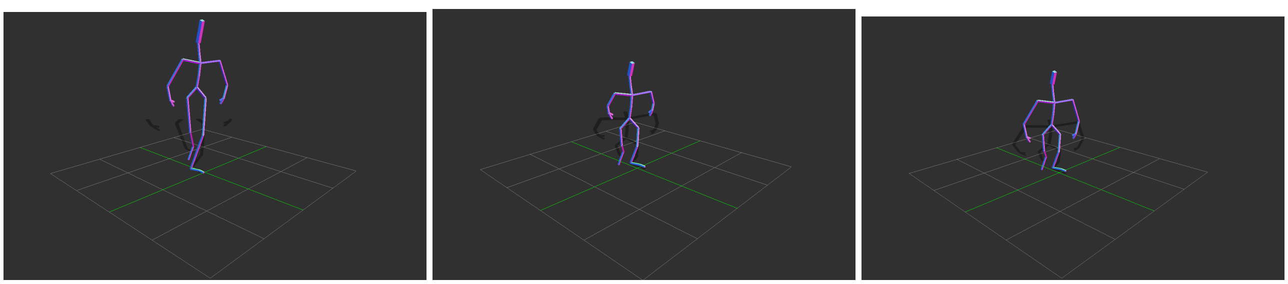


(b) basketball dribbling.



(c) Rolling on the ground.

Figure 3: Synthetic motion from testing dataset.



(a) basketball dribbling. (b) basketball shooting. (c) ballet dancing.

Figure 4: Synthetic motion from training dataset.

iterative loop is advised to terminate after a fixed number of iterations in order to avoid an endless loop. However, this could lead to a sub-optimal result and resulting in fluctuations of the synthetic motion.

4.10. Naturalness

Evaluating whether a motion sequence is natural is difficult and varies for different scenarios and subjects. We here evaluate the naturalness by comparing the difference between the synthetic and original motion data.

4.11. Discussions

4.12. Comparison of Different Network Structures

To effectively validate the choice of our network, we compare the performance against other popular networks, including Convolutional Neural Network, Recurrent Neural Network and Generative Adversarial Network. The implementation of using these networks is included in the public Github repository (please see the link in Section 4.1)

The results show that RNN with considerably smaller size achieves comparable performance with FCN. Meanwhile, FCN is mostly likely to result in the problem of overfitting than RNN.

CNN does not yield satisfactory performance in our case.

4.12.1 Comparison of Different Number of Input Frames

Figure 7 compares the learning performance when different number of frames are used as network input. The plot show that when a single frame is used as network input, the learning performance do not produce satisfactory results. When

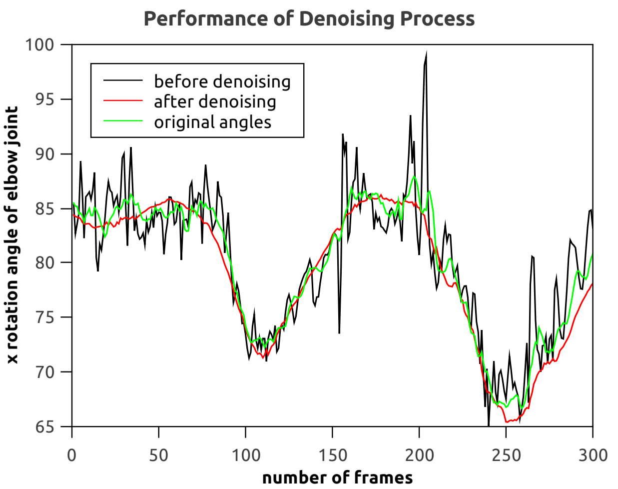


Figure 5: Comparison of the learning performance with different network structures.

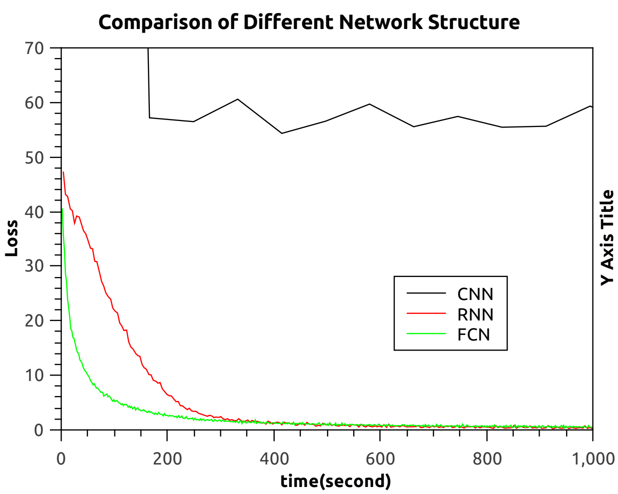


Figure 6: Comparison of the learning performance with different network structures.

increasing the number of frames as input, the learning performance is improved, however with higher variance. As a balance of learning performance and stability, the network input of our implementation is set to 5.

4.12.2 Comparison of Left and Right Arms

Figure 8 compares the learning performance of IK solver for left and right arms. The learning performance is similar in both scenarios, except that the result of left arm is relatively smaller and more stable. Such difference should be caused by the fact that the right arm is responsible for more complex movements, except for the minority group who are left-handed. The higher complexity of hand motion on the

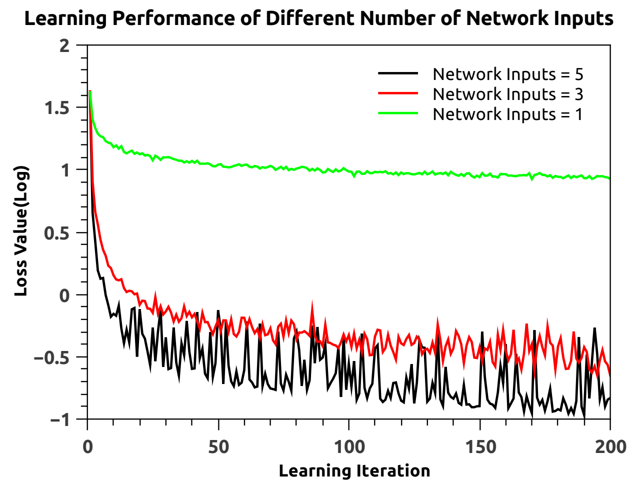


Figure 7: Comparison of the learning performance when different number of frames are used as network input.

right side leads to a higher loss and variance in comparison to the left hand.

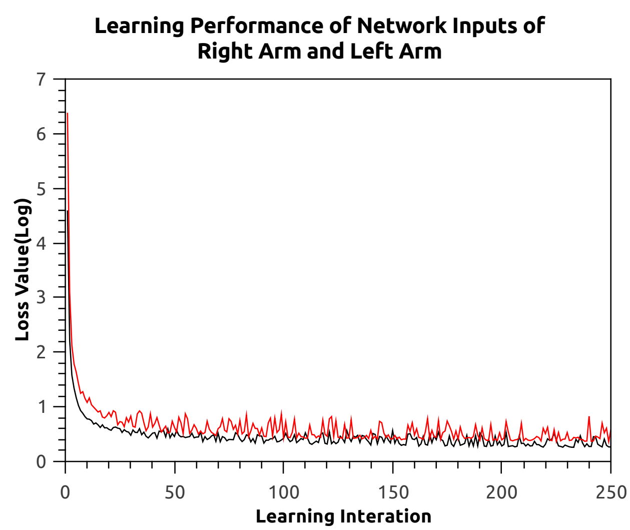


Figure 8: Comparison of the learning performance for left and right arms.

4.12.3 Comparison of Arm and Foot

Figure 9 compares the learning performance of IK solver for foot and arm on the right side. The result shows that the average loss and variance of the foot is much smaller than those of the arm. This reflects the fact that the hand motion is more delicate and complex than the foot motion, which is confirmed in existing research [reference].

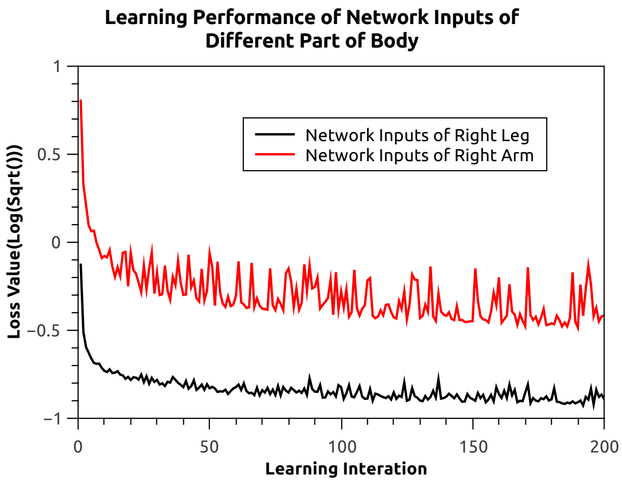


Figure 9: Comparison of the learning performance for foot and arm.

5. Conclusion

This paper proposes a method for solving the IK problem with the deep neural network. This IK solver is capable of handling scenarios involving complex environmental contacts, while still producing postures which are most similar to real humans.

For future works, we plan to explore an automatic solution in designing the network hierarchy. Currently our method constructs the network with human intuition, which is time-consuming. Another research direction is to apply our IK solver to full-body motion capture with inertial sensors attached on the end-effectors. This could improve the naturalness of the synthetic motion.

References

[1] R. F Reinhart and J. J Steil. Neural learning and dynamical selection of redundant solutions for inverse kinematic control. In *Ieee-Ras International Conference on Humanoid Robots*, pages 564–569, 2011. 1

[2] K. Grochow, S. Martin, A. Hertzmann, and Z. Popovic. Style-based inverse kinematics. *Siggraph Acm Siggraph Papers*, 23(3):522–531, 2004. 1, 2

[3] Maxime Tournier, Xiaomao Wu, Nicolas Courty, Elise Arnaud, and Lionel Reveret. Motion compression using principal geodesics analysis. In *Computer Graphics Forum*, volume 28, pages 355–364. Wiley Online Library, 2009. 1, 2

[4] Jing Huang, Qi Wang, Marco Fratarcangeli, Ke Yan, and Catherine Pelachaud. Multi-variate gaussian-based inverse kinematics. In *Computer Graphics Forum*, volume 36, pages 418–428. Wiley Online Library, 2017. 1, 2

[5] Raşit Köker, Cemil Öz, Tarık Çakar, and Hüseyin Ekiz. A study of neural network based inverse kinematics solution for a three-joint robot. *Robotics and Autonomous Systems*, 49(3):227–234, 2004. 1

[6] Bassam Daya, Shadi Khawandi, and Mohamed Akoum. Applying neural network architecture for inverse kinematics problem in robotics. *Journal of Software Engineering and Applications*, 3(03):230, 2010. 1, 2

[7] Yin Feng, Wang Yao-nan, and Yang Yi-min. Inverse kinematics solution for robot manipulator based on neural network under joint subspace. *International Journal of Computers Communications and Control*, 7(3):459–472, 2012. 1, 2

[8] Sergey Levine and Vladlen Koltun. Learning complex neural network policies with trajectory optimization. In *International Conference on Machine Learning*, pages 829–837, 2014. 1

[9] Daniel Holden, Jun Saito, and Taku Komura. A deep learning framework for character motion synthesis and editing. *ACM Transactions on Graphics (TOG)*, 35(4):138, 2016. 1

[10] Daniel Holden, Taku Komura, and Jun Saito. Phase-functioned neural networks for character control. *ACM Transactions on Graphics (TOG)*, 36(4):42, 2017. 1

[11] Alexis Pietak, Siyan Ma, Caroline W Beck, and Mark D Stringer. Fundamental ratios and logarithmic periodicity in human limb bones. *Journal of anatomy*, 222(5):526–537, 2013. 2

[12] H. Choset, K.M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L.E. Kavraki, and S. Thrun. Principles of robot motion: Theory, algorithms, and implementation. 2005. 2

[13] D.C. Brogan, R.A. Metoyer, and J.K. Hodgins. Dynamically simulated characters in virtual environments. 1998. 2

[14] Nicolas Courty and Elise Arnaud. Inverse kinematics using sequential monte carlo methods. In *International Conference on Articulated Motion and Deformable Objects*, pages 1–10. Springer, 2008. 2

[15] Chris Hecker, Bernd Raabe, Ryan W Enslow, John DeWeese, Jordan Maynard, and Kees van Prooijen. Real-time motion retargeting to highly varied user-created morphologies. In *ACM Transactions on Graphics (TOG)*, volume 27, page 27. ACM, 2008. 2

[16] Andreas Aristidou, Joan Lasenby, Yiorgos Chrysanthou, and Ariel Shamir. Inverse kinematics techniques in computer graphics: A survey. In *Computer Graphics Forum*, volume 37, pages 35–58. Wiley Online Library, 2018. 2, 5

[17] C.S.G. LEE. Dynamically simulated characters in virtual environments. 1984. 2

[18] Deepak Tolani, Ambarish Goswami, and Norman I. Badler real-time inverse kinematics techniques for anthropomorphic limbs. 2

[19] Samuel R Buss. Introduction to inverse kinematics with jacobian transpose, pseudoinverse and damped least squares methods. *IEEE Journal of Robotics and Automation*, 17(1-19):16, 2004. 2

[20] Luis Unzueta, Manuel Peinado, Ronan Boulic, and Ángel Suescun. Full-body performance animation with sequential inverse kinematics. *Graphical models*, 70(5):87–104, 2008. 2

[21] Shohin Mukherjee, Deepak Paramkusam, and Santosha K Dwivedy. Inverse kinematics of a nao humanoid robot using kinect to track and imitate human motion. In *Robotics, Automation, Control and Embedded Systems (RACE), 2015 International Conference on*, pages 1–7. IEEE, 2015. 2

[22] Pawan Harish, Mentar Mahmudi, Benoît Le Callennec, and Ronan Boulic. Parallel inverse kinematics for multithreaded architectures. *ACM Transactions on Graphics (TOG)*, 35(2):19, 2016. 2

[23] Adria Colomé and Carme Torras. Redundant inverse kinematics: Experimental comparative review and two enhancements. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 5333–5340. IEEE, 2012. 2

[24] Jianmin Zhao and Norman I Badler. Inverse kinematics positioning using nonlinear programming for highly articulated figures. *ACM Transactions on Graphics (TOG)*, 13(4):313–336, 1994. 2

[25] Charles Rose, Brian Guenter, Bobby Bodenheimer, and Michael F Cohen. Efficient generation of motion transitions using spacetime constraints. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 147–154. ACM, 1996. 2

[26] Ben Kenwright. Inverse kinematics–cyclic coordinate descent (ccd). *Journal of Graphics Tools*, 16(4):177–217, 2012. 2

[27] Andreas Aristidou and Joan Lasenby. Fabrik: A fast, iterative solver for the inverse kinematics problem. *Graphical Models*, 73(5):243–260, 2011. 2

[28] Matthias Rolf, Jochen J Steil, and Michael Gienger. Goal babbling permits direct learning of inverse kinematics. *IEEE Transactions on Autonomous Mental Development*, 2(3):216–229, 2010. 2

[29] Charles F Rose III, Peter-Pike J Sloan, and Michael F Cohen. Artist-directed inverse-kinematics using radial basis function interpolation. In *Computer Graphics Forum*, volume 20, pages 239–250. Wiley Online Library, 2001. 2

[30] Jinxiang Chai and Jessica K Hodgins. Performance animation from low-dimensional control signals. In *ACM Transactions on Graphics (ToG)*, volume 24, pages 686–696. ACM, 2005. 2

[31] Schubert R Carvalho, Ronan Boulic, and Daniel Thalmann. Interactive low-dimensional human motion synthesis by combining motion models and pik. *Computer Animation and Virtual Worlds*, 18(4-5):493–503, 2007. 2

[32] Daniel Raunhardt and Ronan Boulic. Motion constraint. *The Visual Computer*, 25(5-7):509, 2009. 2

[33] Tg.Mureş 540088 Romania. Petru Maior" University of Tg. Mureş, No. 1 N.Iorga St. Neural network based inverse kinematics solution for trajectory tracking of a robotic arm. 2

[34] Rene Felix Reinhart and Jochen Jakob Steil. Neural learning and dynamical selection of redundant solutions for inverse kinematic control. 3

[35] M. Lukosevicius and H. Jaeger. Reservoir computing approaches to recurrent neural network training. *Computer Science Review*, 2009. 3

[36] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew. Extreme learning machine: a new learning scheme of feed forward neural networks. *IEEE Intern. Joint Conf. on Neural Networks*. 3

[37] H SIDENBLADH, M. J. BLACK, and L SIGAL. Implicit probabilistic models of human motion for synthesis and tracking. *Proc. ECCV, LNCS 2353, vol. 1, 784–800*, 2002. 3

[38] Akos Csiszar, Jan Eilers, and Alexander Verl. On solving the inverse kinematics problem using neural networks. In *Mechatronics and Machine Vision in Practice (M2VIP), 2017 24th International Conference on*, pages 1–6. IEEE, 2017. 3