2023

Adaptive Critic Network for Person Tracking Using 3D Skeleton Data

Joseph G. Zalameda
Old Dominion University, jzala001@odu.edu

Alex Glandon
Old Dominion University, aglan001@odu.edu

Khan M. Iftekharuddin
Old Dominion University, kiftekha@odu.edu

Mohammad S. Alam (Ed.)

Vijayan K. Asari (Ed.)

Original Publication Citation

Zalameda, J. G., Glandon, A., & Iftekharuddin, K. M. (2023). Adaptive critic network for person tracking using 3D skeleton data. In M. S. Alam & V. K. Asari (Eds.), *Pattern Recognition and Tracking XXXIV, Proceedings of SPIE 12527* (125270J). SPIE. https://doi.org/10.1117/12.2663409

Adaptive critic network for person tracking using 3D skeleton data

Joseph Zalameda, Alex Glandon, Khan Iftekharuddin

Joseph G. Zalameda, Alex Glandon, Khan M. Iftekharuddin, "Adaptive critic network for person tracking using 3D skeleton data," Proc. SPIE 12527, Pattern Recognition and Tracking XXXIV, 125270J (13 June 2023); doi: 10.1117/12.2663409

Event: SPIE Defense + Commercial Sensing, 2023, Orlando, Florida, United States

Adaptive Critic Network for Person Tracking using 3D Skeleton Data

Joseph G. Zalameda, Alex Glandon, Khan M. Iftekharuddin
[a]Vision Lab in Department of Electrical and Computer Engineering, Old Dominion University,
Norfolk, VA 23529

ABSTRACT

Analysis of human gait using 3-dimensional co-occurrence skeleton joints extracted from Lidar sensor data has been shown a viable method for predicting person identity. The co-occurrence based networks rely on the spatial changes between frames of each joint in the skeleton data sequence. Normally, this data is obtained using a Lidar skeleton extraction method to estimate these co-occurrence features from raw Lidar frames, which can be prone to incorrect joint estimations when part of the body is occluded. These datasets can also be time consuming and expensive to collect and typically offer a small number of samples for training and testing network models. The small number of samples and occlusion can cause challenges when training deep neural networks to perform real time tracking of the person in the scene. We propose preliminary results with a deep reinforcement learning actor critic network for person tracking of 3D skeleton data using a small dataset. The proposed approach can achieve an average tracking rate of 68.92±15.90% given limited examples to train the network.

Keywords: Actor Critic Network, Lidar, Skeleton data, Person Tracking, Small Dataset

1. INTRODUCTION

Three-dimensional skeleton data provides a useful modality for human feature representation. Skeleton data may be used with machine learning and deep learning algorithms for applications like person identification[1,2], action recognition[3-5], and gender classification[6]. These use cases demonstrate the applicable advantages of skeleton data in fields such as security and defense, especially when they can be implemented for real time systems. Furthermore, the demand for robust, real-time deployment of these algorithms has given rise to the need to accurately convert image data to skeleton data through the use of joint location estimation techniques.

Accurate and efficient joint location estimation for different image domains is vital for deep learning algorithms to function efficiently. Lidar based silhouette joint estimation for subject skeleton extraction[2] has been shown to achieve promising results when used to extract datasets to train deep Convolutional Neural Network (CNN) based person identification algorithms like Hierarchical Co-occurrence Network for Person Identification[1]. Lidar based skeleton data is more advantageous for on-site deployment as opposed to visual based[7] or motion capture (MoCap) based methods as it can work at long range and in low light environments. However, Lidar based skeleton data exhibits outlier joint location issues that can stem from environmental noise, occluded joints, and null frames. Additionally, acquisition of Lidar data for skeleton extraction can be time consuming and expensive, which leads less available data for training and testing. These issues pose challenges for algorithm development using Lidar based skeleton data.

Utilization of deep neural networks for object tracking have been shown to perform well with image data. Deep CNN based architectures have been used to perform both object classification and tracking in real time for images[8]. This work however focuses on utilization of deep networks for tracking and classification. Deep Reinforcement Learning (RL) based approaches[9] utilize a series of recurrent neural network layers to track objects in image data. Current actor critic based RL approaches to track objects in image data in real time utilizes an actor network with a continuous state space that is trained both online and offline[10] as well as a discrete action space[11]. Both methods performs well when compared with other state of the art approaches. Work on these methods however have been limited to object tracking in the image domain. Inspired by the success of deep networks along with RL and recurrent based tracking approaches, we propose a recurrent adaptive critic-based network to perform noise resilient tracking of skeleton data in 3-dimensional space. Our adaptive critic approach uses the actor critic network structure for training.

Pattern Recognition and Tracking XXXIV, edited by Mohammad S. Alam, Vijayan K. Asari,
Proc. of SPIE Vol. 12527, 125270J · © 2023 SPIE · 0277-786X · doi: 10.1117/12.2663409

To the best our knowledge, RL based adaptive critic real time tracking algorithms have not been studied to track person in small Lidar based skeleton data. This work presents preliminary results for adaptation of this network for tracking person utilizing a discrete action space as opposed to the continuous action space used in the current state of the art image-based object tracking actor critic[10]. Our implementation is designed to leverage deep neural networks to approximate a policy function and value function. This implementation will also require the design of a novel skeleton tracking environment.

The contributions of this work are as follows:

1. Design a novel framework that adequately represents skeleton joint locations and human movement given a small and noisy Lidar extracted skeleton dataset.
2. Train an actor critic design network with a discrete policy action space to track skeletons in real time given limited training data.
3. Evaluate actor critic tracking performance on unseen Lidar skeleton data.

2. BACKGROUND

2.1 Actor Critic Design

The actor critic design used in this work consists of an actor network π_{θ_a} with parameters θ_a , critic network J_{θ_c} with parameters θ_c, and an Environment containing an infinite number of states s_t given a finite number of skeleton data samples. following previous feedforward actor critic network implementation examples[12,13]. The actor's job is to approximate a policy function which takes as input, a set of environmental conditions known as the state s and outputs an action distribution that consists of probabilities to take each action. This relationship is given in Eq. 1 where s_t is the environmental state at time t and A_t is the action space output at time t as,

$$A_t = \pi_{\theta_a}(s_t). \tag{1}$$

Here A_t is a 1-dimensional tensor where indexed values represent the probability of the action equaling that index. Because A_t is limited in the number of actions, one actor critic network is trained to track each axis in the scene. Final testing of the network combines the outputs of multiple actor networks to update the position of a bounding box drawn in 2-dimensions in the scene. The critic network approximates the value function given current state s_t. The environment is designed to output the state, which contains information of A_t in addition to the raw joint data. The value function approximates the discounted cost to go function V_t, which estimates total current and future reward given s_t. This function is shown in Eq. 2.

$$V_t = J_{\theta_c}(s_t). \tag{2}$$

There are three main functions of the environment in this design. The first function is to output a set of environmental conditions to the actor and critic networks known as the state s_t. The second function is to interpret and apply the action set A_t with the goal of making a move within the state space. This move results in an updated state at time $t+1$ denoted as s_{t+1}. The third function is to output a reward value r_t to train the actor and critic networks. The environment function is given in Eq. 3.

$$s_{t+1}, r_t = Env(A_t|s_t) \tag{3}$$

As the actor critic is inherently a real time-based approach, network training of both the actor and critic occurs at the simultaneously. The loss function for both the actor and critic networks utilizes the advantage function[13] in Eq. 4, which quantifies current, and future expected total reward given the s_t and s_{t+1}. The critic loss is designed to better approximate the Q value function given observed values of r_t. This advantage function and critic error are adapted from the critic error function used in. The goal of the actor loss is to use the advantage function to weight actions with higher expected reward higher than actions with lower expected reward.

$$Advantage(s_t, s_{t+1}, r_t) = r_t + J_{\theta_c}(s_{t+1}) - J_{\theta_c}(s_t); \tag{4}$$

$$\mathrm{Actor\ Loss} = -\log\left(\pi_{\theta_a}(s_t)\right) * \mathrm{Advantage}(s_t, s_{t+1}, r_t); \tag{5}$$

$$Critic\ Loss = 0.5 * Advantage(s_t, s_{t+1}, r_t)^2 \tag{6}$$

2.2 Skeleton Data

The skeleton dataset used to train this model is extracted from Lidar videos[2]. All videos are acquired at a fixed camera position with the subject walking in a circular pattern in the scene. The dataset consists of 10 subjects divided among 22 videos ranging from roughly 120 to 500 frames each. Skeleton video is denoted by the tensor X_i where i corresponds with the video number. Each video sample X_i contains 13 joints tracked over 3 spatial dimensions and is structured according to Eq. 7.

$$X_i = [dimension, frame, joint] \tag{7}$$

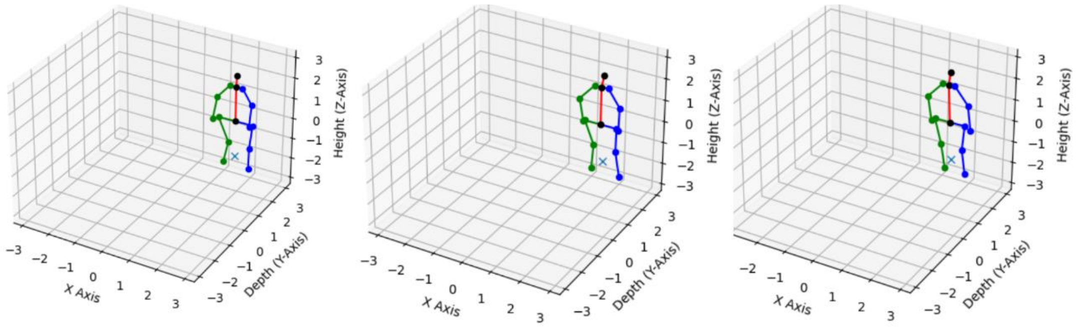

Figure 1. Example Lidar Skeleton Frames Without Occluded Joints

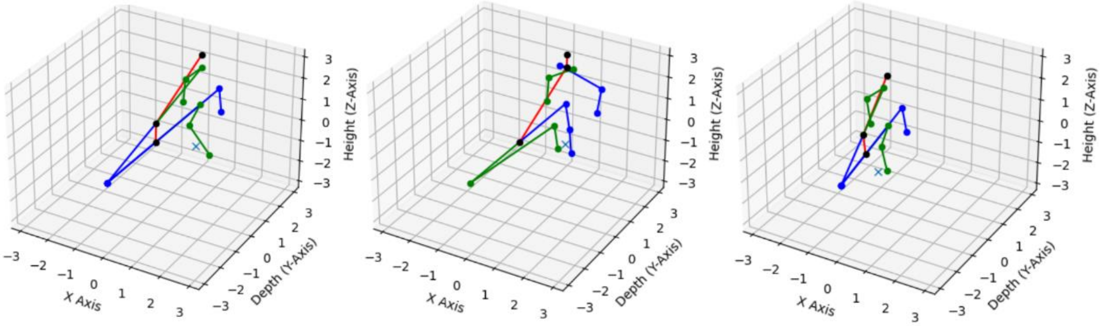

Figure 2. Example Lidar Skeleton Frames with Occluded Joints

Figure 1 gives examples of clean skeleton tracking of the Lidar data. In this case, the subject is oriented in a manner shows all joints to the camera. Therefore, the skeleton extraction algorithm can accurately detect and map the joint locations in the space. Because the skeleton remains on level ground through the entire video duration, tracking will be limited to the X and Y axes.

Figure 2 gives an example of sequential noisy frames where the skeleton is turning. In this case, while the skeleton is turning in the scene, the occluded joint locations are estimated using matrix completion[2]. This results in joint estimations that exhibit high noise. This noise is a huge obstacle to develop classification and tracking algorithms that may be trained and tested on this Lidar based skeleton dataset.

3. METHODOLOGY

3.1 Dataset

Once extracted from the Lidar videos, the skeleton data is preprocessed in two steps. The first step is to normalize the X-axis and Y-Axis from [-3,3]. The second step is to create a set of ground truth locations for all frames of each video. This step is done by utilizing several low pass averaging filter convolutions on the average joint locations projected along the

X and Y Axes separately. This results in a smoother and more accurate location representation along each axis when compared with the average location. This step can only be done with prerecorded videos as it takes into account past and future joint locations, which is not feasible for real time operation. This method however proved to work well for training the actor critic network. To show the actor critic network does not need all joints to track the skeleton, a subset of joints is used as data for the environment which encompasses data from the legs, torso, and upper body.

3.2 The Overall Network Framework

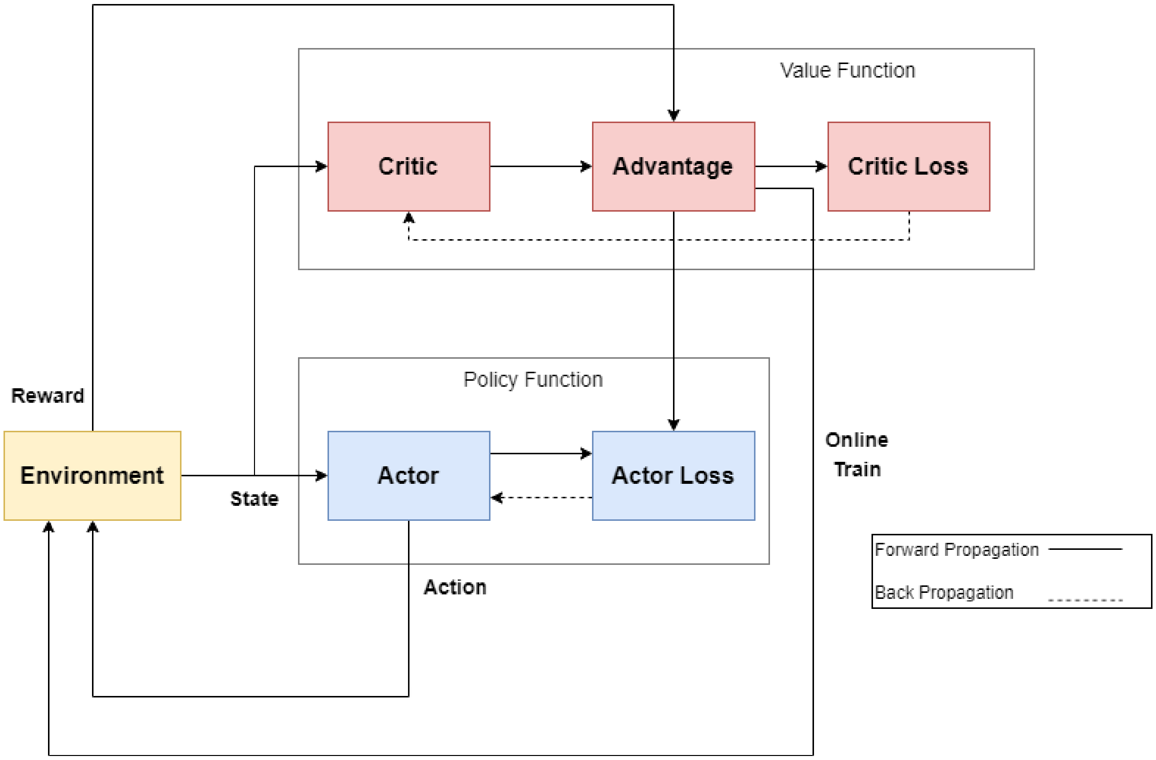

Figure 3. Actor Critic Network

The overall network structure for the online training method is shown in Figure 3. The network performs backpropagation after each time step in the training process. The main difference between the structure for batch training and online training is the link from the advantage function to the environment. In the case of online training, the environment will only move to the next frame in the video sequence when the advantage value meets a predetermined threshold, or a maximum count is reached. This indicates an optimal state trajectory for the network. Meeting this threshold implies the actor network is sufficiently trained on the current frame and the environment should move to the next frame in the skeleton video. This method has the advantage of being able to check for correct convergence at each frame before moving to the next frame, which will ensure the model is training only on highly rewarding state paths within the environment.

3.3 Environment Design

The actor critic network utilized for this tracking problem requires a novel Environment structure that accurately maps actions to location estimations of the skeleton data over time. This environment is designed to provide tracking information to the actor critic along one spatial dimension, indicated by the *dimension* input in Figure 4. This environment can be used for both online and batch training of the actor critic. The reward output from the environment needs to accurately reinforce good tracking actions taken by the actor network. Because the action space A_t is discrete in this design, each action must correlate with a fixed movement for state *t* in the estimated skeleton location $\widehat{Y}_t$ and compared with the actual location Y_t. Both $\widehat{Y}_t$ and Y_t are stored and updated internally within the Environment. Using the stored values, the environment in this network computes four functions that are needed for the recurrent operation of this actor critic design, which are, action interpretation, reward computation, state output, and episode termination.

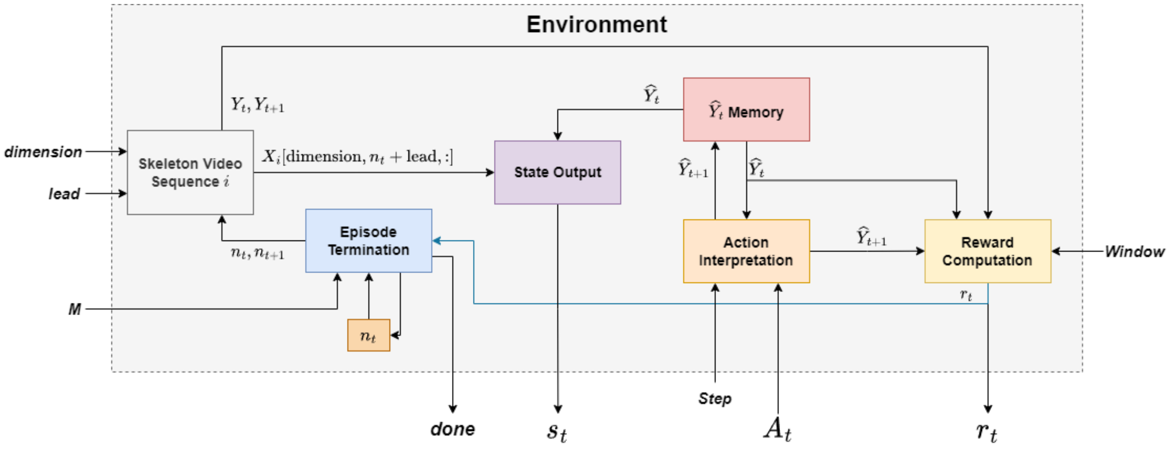

Figure 4. Environment Structure

The goal of action interpretation is to sample A_t using the probability values within the action according to Eq. 8 and update $\widehat{Y}_t$ *to* $\widehat{Y_{t+1}}$ following the logic in Eq. 9. This equation takes as input the current estimated location $\widehat{Y}_t$, the movement size $Step$, and A_t. The *Step* value in Eq. 9 is a predefined constant which influences the maximum movement $\widehat{Y}_t$ can make in one iteration.

$$action = \text{prob\_sample}(sample(-\log(A_t))) \tag{8}$$

$$\widehat{Y_{t+1}} = \begin{cases} \widehat{Y}_t - Step & action = 0 \\ \widehat{Y}_t - \dfrac{Step}{10} & action = 1 \\ \widehat{Y}_t & action = 2 \\ \widehat{Y}_t + \dfrac{Step}{10} & action = 3 \\ \widehat{Y}_t + Step & action = 4 \end{cases} \tag{9}$$

The reward computation function is implemented as a binary reward structure with reward being r_t =1 and punishment being r_t =0 for batch training. The *window* constant seen in Figure 4 specifies the margin of error for the tracking algorithm. Eq. 10 shows the reward conditional output of the reward computation function as follows,

$$r_t = \begin{cases} 1 & \left|\widehat{Y_{t+1}} - Y_{t+1}\right| \leq \dfrac{window}{2} \ or \ \left|Y_t - \widehat{Y}_t\right| \geq \left|Y_{t+1} - \widehat{Y}_t\right| \\ 0 & otherwise \end{cases} \tag{10}$$

This function outputs a reward of 1 if either $\widehat{Y_{t+1}}$ is within the window range of Y_{t+1} with the window being centered at Y_{t+1} or if the $\widehat{Y_{t+1}}$ is outside the window but makes a move towards Y_{t+1}. If either of these conditions are not met, r_t =0, which punishes the network. Online training utilizes the reward structure shown in Eq. 11, which adds extra reinforcement to positive moves within the tracking *window* as well as discounted reinforcement to positive moves outside the *window*.

The episode termination function is required to set two conditions under which the training episode ends. The first condition involves keeping a running total of the number of times $r_t = 0$ during a training episode. Once the total reaches max count M, the episode ends. The second condition is met if the episode has reached the end of the skeleton video. Once the episode is ended, the episode termination function will output a signal of $done = True$, which will trigger the training of the network. In online training this condition is always set to *True* and n_t only resets if the end of the video is reached. The *lead* variable is a constant that offsets the ground truth output of the real skeleton locations by a fixed amount.

$$r_t = \begin{cases} 1 & \left|\widehat{Y_{t+1}} - Y_{t+1}\right| \le \dfrac{window}{2} \ and \ \left|Y_t - \widehat{Y}_t\right| \neq \left|Y_{t+1} - \widehat{Y}_t\right| \\ 2 & \left|\widehat{Y_{t+1}} - Y_{t+1}\right| \le \dfrac{window}{2} \ and \ \left|Y_t - \widehat{Y}_t\right| = \left|Y_{t+1} - \widehat{Y}_t\right| \\ 0.5 & \left|Y_t - \widehat{Y}_t\right| > \left|Y_{t+1} - \widehat{Y}_t\right| \\ 0 & otherwise \end{cases} \tag{11}$$

3.4 Network Training

Both batch and online methods are used to train the actor critic network on this environment. Both batch and online training methods leverage the same loss functions in Eq. 5 and Eq. 6, with the main differences being gradient computation at time of backpropagation and design of the Episode Termination function. Figure 3 shows the actor critic network structure with backpropagation.

The batch train method is performed by utilizing the actor network to navigate the environment using an episodic approach. Each episode allows the network to navigate the environment until the *done=True* condition is satisfied by the Episode Termination function. At each time step, the reward, actor output, and critic output are saved to a memory buffer. Once the episode terminates, the returns and Advantage at each time step are compiled into a set of actor and critic losses. This compilation of losses is then used as a batch loss and the mean derivative is used for backpropagation through the network. This method has the advantage of observing more of the environment per training step.

The network structure for the online training method performs Backpropagation after each time step in the training process. The main difference between the structure for batch training and online training is the link from the advantage function to the environment. In this case, the environment will only move to the next frame in the video sequence when the advantage value meets a predetermined threshold or a maximum count is reach, indication a sub optimal state for the network. Meeting this threshold implies the actor network is sufficiently trained on the current frame and the environment should move to the next frame in the skeleton video. This method has the advantage of being able to converge correctly at each state, which will ensure the model is training only on highly rewarding state paths within the environment.

3.5 Experiment

The metric used to quantify performance of the network is tracking success rate, which computes the percentage of frames within a video the network is able to estimate the correct skeleton location within the window size it is trained on. To assess the ability of the actor critic to learn to track skeletons through the proposed environment, we organize our train/test structure into folds. Each fold will only train on one continuous video in our 22-video dataset and test on all 21 remaining videos in the dataset. We perform this train/test cycle once per each video. The testing results for each video are then averaged across all folds for both training methods.

Because training the actor critic method using the batch method is inherently unstable, an early stopping feature is implemented, which stops training the network if the maximum frame number in the video is reached in 5 consecutive episodes. This helps to prevent the network from overfitting on a suboptimal state trajectory, or hopping between state trajectories, which can occur in later stages of training. The hyperparameters for batch training are determined empirically, with an actor learning rate of 0.001, a critic learning rate of 0.01, a step value of 1, window size of 2, max count of 5, and a lead value of 0, respectively. The Adam optimization algorithm[14] is used with $\beta_1 = 0.9$ and $\beta_2 = 0.999$ to update the network weights.

To assess network performance with online training, the networks are trained for 30 cycles over the skeleton video following the inherent online training capability of the actor-critic network to increment the frames in the video. Early stopping is not needed in this case because the online training method increases the chance that the network will train on more optimal state trajectories. The hyperparameters for online training are determined empirically, with an actor learning rate of 0.001, a critic learning rate of 0.001, a step value of 1, and a lead value of 0, respectively. An internal training iteration limit for each state of 30 is used to prevent overfitting. The Stochastic Gradient Descent optimization algorithm is used to update the network weights.

4. RESULTS

4.1 Training Convergence

The proposed network shows a strong tendency to converge toward a given state trajectory during training. An example of this convergence can be seen in Figure 5 (A) and (B), which shows the mean loss for each episode using batch training. Here the network converges to multiple state trajectories which are indicated by the sudden spikes around episode number 100 and 300. This is an example of the network moving between multiple states that are deemed to yield high accumulative reward values and therefore produce a good tracking output. Figure 5 (C) shows the iteration total per episode. This measures then number of frames the network can track before the environment ends the episode. This is a good indication of how well the network is tracking the skeleton because the closer the estimated location is to the actual location, the higher the probability the environment will continue the track. The goal of tracking the whole video without an episode termination is reached around episode 100 which is again reflected in the convergence in both the actor and critic losses. This pattern is reflected in the online training functions as well, though the state transition logic requires convergence at each frame, which significantly increases the noise in the overall training plots in this case.

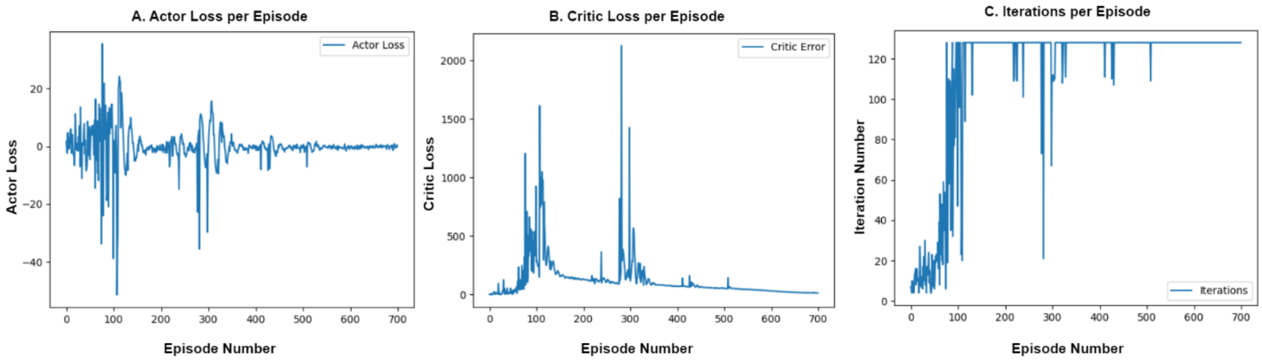

Figure 5. Batch Train Metric Plots Actor Loss (A.), Critic Loss (B.), and Iterations per Episode (C.)

4.2 Quantitative Results

Because the convergence of the actor critic network can be unstable, tracking success rate over all folds exhibits high variance for both batch and online training. Table 1 shows the test time tracking success rate average across all folds for both training methods. Table 1 breaks down the tracking success rate for only X-axis, only Y-axis, and a bounding box approach which combines the X and Y axes. The batch training method achieves the highest bounding box tracking success rate at 68.92±15.90%. The highest single axis accuracy is achieved by the X-axis through batch training at 86.53±7.99%. While both networks learn to track the skeleton in the space, the online training method achieves a much lower bounding box tracking success rate at 39.79±14.80%. The per axis results also achieves lower tracking success rates when compared with batch training.

Table 1. Average Test Time Tracking Success Rate.

Train Method	X-Axis	Y-Axis	Bounding Box (X and Y Axis)
Batch Train	86.53±7.99%	79.26±16.87%	68.92±15.90%
Online Train	69.20±6.89%	58.78±20.56%	39.79±14.80%

4.3 Qualitative Results

Figure 6 shows a 3-frame qualitative example of the bounding box tracking using the actor critic model on a skeleton with additive zero mean gaussian noise with a magnitude of 0.4 to emphasize variability in tracking performance between the actor critic network and simply taking the mean of the joints used. The blue box in Figure 6 represents the bounding box drawn by taking the mean of the joints locations and the black box represents the bounding box estimated by the trained actor critic model. The actual skeleton location between all 3 frames in Figure 6 should remain close together. However, the added gaussian noise along with the noise caused by the occluded joints along with the gaussian in frame 2 shown the actor critic can outperform tracking by simply taking the average location.

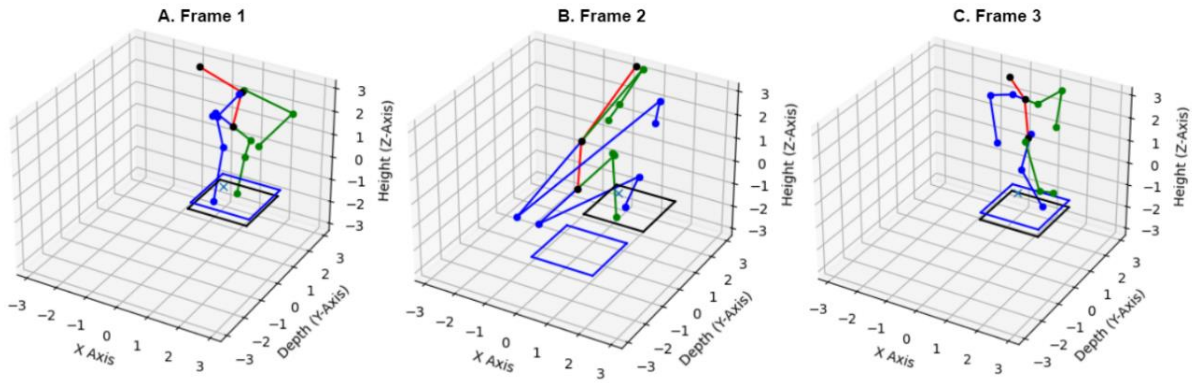

Figure 6. Bounding Box Tracking of Turning Skeleton Sequence - Mean Location Tracking (Blue Box), Actor Critic Tracking (Black Box)

5. CONCLUSION

This work utilizes an actor critic network to show feasibility of a novel tracking environment for a small Lidar extracted skeleton dataset. The actor critic network is trained using both batch and online training methods. The batch training method exhibits better convergence of the network resulting to better person tracking when compared that with the online training method. This convergence however is extremely sensitive to underlying conditions within the networks and environment which yield high variability in the results. Future work should address this high degree of variability when training the actor critic model, which should improve both mean tracking test success rate and variance of this success rate. Future work should also assess tracking performance of the actor critic network with extended videos and multiple objects within the scene. The recurrent structure of this actor critic design can, in the future, be leveraged to predict future skeleton locations using a similar environment to the one utilized in this paper.

6. ACKNOWLEDGEMENTS

The authors would like to acknowledge partial support of this work by US Army NVESD, CERDEC through Grant No. 100659, DoD Center of Excellence in AI and Machine Learning (CoE-AIML) under Contract Number W911NF-20- 2-0277 with the U.S. Army Research Laboratory, and by the National Science Foundation under Grant No. 1828593, Grant No. 1950704, and Grant No. 1753793.

REFERENCES

[1] J. G. Zalameda et al., "Generalized Adversarial and Hierarchical Co-occurrence Network based Synthetic Skeleton Generation and Human Identity Recognition," in *2022 International Joint Conference on Neural Networks (IJCNN)*, 2022.

[2] A. Glandon et al., "3D Skeleton Estimation and Human Identity Recognition Using Lidar Full Motion Video," in *2019 International Joint Conference on Neural Networks (IJCNN)*, 2019.

[3] Y. Du, Y. Fu and L. Wang, "Skeleton based action recognition with convolutional neural network," in *2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)*, 2015.

[4] A. Kerboua and M. Batouche, "3D Skeleton Action Recognition for Security Improvement," *International Journal of Intelligent Systems and Applications,* vol. 11, pp. 42-52, 8 March 2019.

[5] C. Li, Q. Zhong, D. Xie and S. Pu, "Co-occurrence Feature Learning from Skeleton Data for Action Recognition and Detection with Hierarchical Aggregation," *CoRR*, vol. abs/1804.06055, 2018.

[6] M. Navarro et al., "Gender classification of full-body biological motion of aperiodic actions using machine learning," in *Optics and Photonics for Information Processing XIV*, 2020.

[7] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei and Y. Sheikh, "OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields," *CoRR*, vol. abs/1812.08008, 2018.

[8] G. Chandan, A. Jain, H. Jain and Mohana, "Real Time Object Detection and Tracking Using Deep Learning and OpenCV," in *2018 International Conference on Inventive Research in Computing Applications (ICIRCA)*, 2018.

[9] D. Zhang, H. Maei, X. Wang and Y.-F. Wang, "Deep Reinforcement Learning for Visual Object Tracking in Videos," *CoRR,* vol. abs/1701.08936, 2017.

[10] B. Chen, D. Wang, P. Li, S. Wang and H. Lu, "Real-Time `Actor-Critic' Tracking," in *Computer Vision – ECCV 2018*, Cham, 2018.

[11] O. Dogru, K. Velswamy and B. Huang, "Actor–Critic Reinforcement Learning and Application in Developing Computer-Vision-Based Interface Tracking," Engineering, vol. 7, pp. 1248-1261, 2021.

[12] J. Si and Y.-T. Wang, "Online learning control by association and reinforcement," *IEEE Transactions on Neural Networks,* vol. 12, pp. 264-276, 2001.

[13] S. Bhatnagar, R. S. Sutton, M. Ghavamzadeh and M. Lee, "Natural actor–critic algorithms," *Automatica,* vol. 45, pp. 2471-2482, 2009.

[14] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," arXiv preprint arXiv:1412.6980, 2014.