2021 43rd Annual International Conference of the
IEEE Engineering in Medicine & Biology Society (EMBC)
Oct 31 - Nov 4, 2021. Virtual Conference

Improving Accuracy and Runtime of Skeletal Tracking of Lower Limbs for Athletic Jump Mechanics Assessment*

Aloys Portafaix[1] Thomas Fevens[2]

Abstract— **Previous studies have shown that athletic jump mechanics assessments are valuable tools for identifying indicators of an individual's anterior cruciate ligament injury risk. These assessments, such as the drop jump test, often relied on camera systems or sensors that are not always accessible nor practical for screening individuals in a sports setting. As human pose estimation deep learning models improve, we envision transitioning biometrical assessments to mobile devices. As such, here we have addressed two of the most preclusive hindrances of the current state-of-the-art models: accuracy of the lower limb joint prediction and the slow runtime of in-the-wild inference. We tackle the issue of accuracy by adding a post-processing step that is compatible with all inference methods that outputs 3D key points. Additionally, to overcome the lengthy inference rate, we propose a depth estimation method that runs in real-time and can function with any 2D human pose estimation model that outputs COCO key points. Our solution, paired with a state-of-the-art model for 3D human pose estimation, significantly increased lower-limb positional accuracy. Furthermore, when paired with our real-time joint depth estimation algorithm, it is a plausible solution for developing the first mobile device prototype for athlete jump mechanics assessments.**

I. INTRODUCTION

Hewett *et al.* [1] estimated that in 2001, 646 million USD was spent every year in the United States on surgery and rehabilitation for an estimated 38,000 female athletes with anterior cruciate ligament (ACL) injuries. Furthermore, female athletes are prone to ACL injuries at 4 to 6 times the rate of their male counterparts [2].

With these risk profiles in mind, Hewett *et al.* [1] developed a screening process to identify athletes with a higher risk of ACL injuries. Their findings demonstrated that examining the knee's coronal angles at initial contact and the sagittal and coronal angles of the knee at peak contact during drop jump tests yielded pertinent indicators of ACL injury risk. In a drop jump test, the subject drops down from a moderate height and then immediately jumps in the air.

For their biometrical analysis of the athletes, Hewett *et al.* completed these evaluations by relying on skeletal tracking from a motion capture system and identified contact frames using a force plate. Contact frames are defined as the frames where the individual's feet are touching the ground during the jump assessment. As these biometrical analysis systems are costly, the Microsoft Kinect's skeletal tracking has been validated as an acceptable, cost-effective substitute for lower extremity injury screening [3], [4].

While the Kinect is a cost-effective and portable solution compared to the motion capture system, it requires a powerful computer and the Kinect sensor. With the development of 3D human pose estimation models using only RGB video [5], [6], [7], [8], by avoiding the requirement of a complicated multi-camera setup or depth sensors, it may be possible to transition to mobile devices with video capture capabilities which would make it more accessible to coaching staff and other parties interested in testing for ACL injury risk. One such promising Convolutional Neural Network (CNN) model is Pavllo *et al.*'s VideoPose3D [5]. VideoPose3D uses two stages: the first stage involves detecting 2D keypoints using a 2D detector pre-trained on the COCO 2017 dataset [9] followed by a second stage that using temporal convolutions to determine the 3D joint positions.

Since CNN-based 3D pose estimation is very slow, previous work has also considered real-time approaches to determine accurate 3D positions of joints using simple projection and trigonometric solutions. For example, Chalangari's DeepLEAD [10] uses the 2D joint positions generated by OpenPose [11] which are then refined using trigonometric solutions to determine the lower extremity angles of the knees. This relatively modular solution is flexible in that it may be paired with alternative 2D joint position estimation models that can perform inference in real-time, such as PoseNet [12].

With a particular interest in biometrical analysis to determine ACL injury risk, the present work aims to increase the accuracy of the lower limb predictions of skeletal tracking systems such as VideoPose3D by adding a post-processing step built on robot modeling and inverse kinematics. This technique allows us to constrain the limbs' lengths and the joints' ranges of motion, resulting in a more predictable and natural

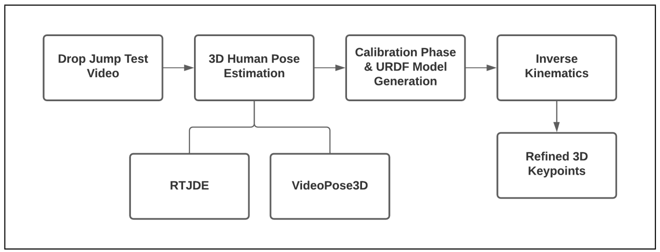

Fig. 1: Proposed pipeline for refining RTJDE or VideoPose3D 3D keypoint predictions

*This work was supported by NSERC Discovery grant RGPIN-2020-06785, NSERC CREATE grant 482735-16, and FRQNT.

[1]Aloys Portafaix is with the Department of Computer Science and Software Engineering, Concordia University, Montréal, QC, Canada a_portaf@encs.concordia.ca

[2]Thomas Fevens is with the Department of Computer Science and Software Engineering, Concordia University, Montréal, QC, Canada thomas.fevens@concordia.ca

movement of the legs. In addition, we investigated techniques of inferring depth using trigonometric solutions similar to DeepLEAD. Our technique differs by estimating the depth of joints in each frame rather than the knees' angles. This approach is advantageous as we can then apply our post-processing steps for better accuracy. Most importantly, this technique can be paired with any 2D human pose estimation model that outputs COCO key points [9] such as Detectron2 [13] and OpenPose [11], or lightweight models such as PoseNet [12]. See Fig. 1 for a diagram of our proposed processing pipeline.

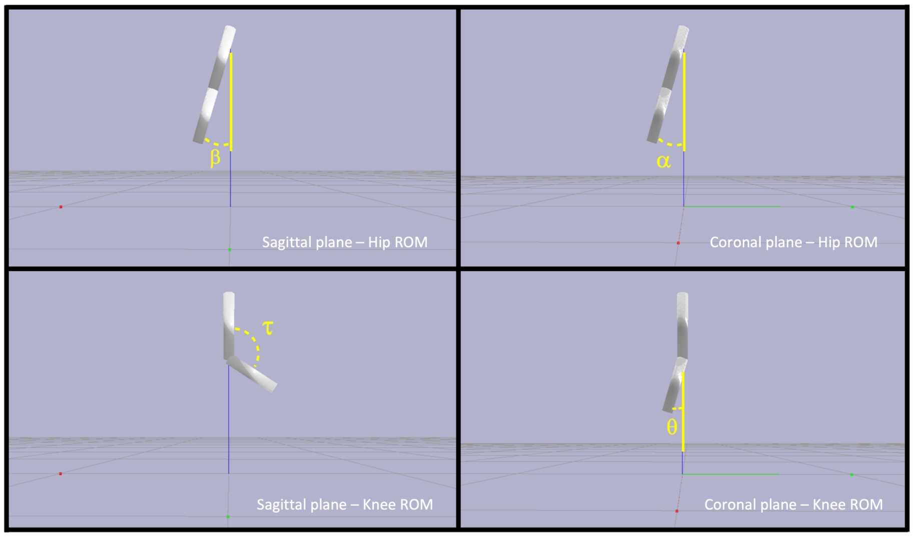

Fig. 2: URDF leg model in PyBullet GUI demonstrating the 4 degrees of freedom.

II. Methodology

A. Calibration Phase and UDRF Model

When running our preliminary trials with VideoPose3D [5], we observed a mean error for a video of the drop jump test of 7-12 degrees for sagittal and 2-5 degrees for coronal angles of the knee when compared to the Microsoft Kinect. Furthermore, when analyzing the subset of frames that include just the contact frames, we see much higher error rates. We observe a mean average error of 25-30 degrees for the sagittal contact frames, with up to 83% of the frames differing by more than 30 degrees. Additionally, we observed a mean average error of up to 15 degrees for the knees' coronal angle. The lack of accuracy, particularly during the contact frames, makes it difficult to argue for using in-the-wild VideoPose3D in our use case. Moreover, the process for in-the-wild VideoPose3D inference on a current-generation machine with no GPU (Intel i5 6-core CPU) takes 45-75 minutes for a 10-15 second 1920x1080 color capture at 30 fps.

Another shortcoming of VideoPose3D is that it does not consider the length of the femur and tibia, nor the range of motion of each joint when performing inference. By constraining the lengths of the bones and the joints' ranges of motion through a dynamically created model of each leg, we wish to restrict the lower limbs' movements to more natural movements.

To achieve this, we add a calibration stage where we set some constraints to the subject appearing in the video capture, similar to that used by DeepLEAD [10]. The video's first frame must have the individual standing with their legs straight in the first frame facing the camera. The 3D inference on this first frame will give us the measurements of each leg's femur and tibia. With these lengths, we define each leg's physical characteristics using a Universal Robot Description Format (URDF) model containing the length of each bone and the constraints on the range of motion of each joint. Additionally, for best results, the subject should always face the camera so that both legs are always visible, consistent with the drop jump test.

The lower limbs of the body contain three articulations: hip, knee, and ankle. We disregarded the ankle for our study as it is not considered an indicator of ACL injury risk. If we reduce the hip and knee joints to their most straightforward representations, we can simulate them with two revolute joints for each articulation—one for the flexion and extension movements and another which imitates abduction. As shown in Fig. 2, the first quadrant demonstrates flexion and extension of the hip joint with constraints -30° $< \beta <$ 100°. The second quadrant is an example of abduction of the hip joint with constraints on the left side: -25° $< \alpha <$ 45° and the right side: -45° $< \alpha <$ 25°. The third quadrant represents the flexion and extension of the knee with constraints -150° $< \tau <$ 0°. The fourth shows the joint simulating abduction of the knee with a constraint on the left side: -20° $< \theta <$ 15° and on the right: -15° $< \theta <$ 20°.

B. Inverse Kinematics (Post-processing)

To simulate the natural movement of each leg, we apply inverse kinematics (IK) for each frame to correct the joints' positions. Inverse kinematics calculates a configuration that is as close as possible to each frame's inferred joint position while retaining the model's constraints. To do this, we use PyBullet, which has an inverse kinematic solver where it is possible to specify multiple end effectors (i.e., knees and feet) and their target positions [14]. We use an incremental solution by prioritizing the movement of the hip to first determine each knee's target position and then solve for the joint of the knee to attain the foot's target position. Using multiple target positions helps to avoid overreaching. We frequently observed this phenomenon with the abduction angle of the knee when solving the IK once for each leg using the foot as the only end effector.

The final step is the orientation and base position of the model. Orientation was set at the beginning of each frame by calculating the orientation vector. This vector is obtained by taking the difference between the hip and knee positions and projecting it onto the floor plane. We then form a quaternion based on the difference between the current orientation vector and the orientation vector from the previous frame. Additionally, we adjust the base position of the model, which is the anchor of the model, by setting it to the hip position.

C. Real Time Joint Depth Estimation (RTJDE)

In a similar fashion to DeepLEAD, we propose a novel lower limb depth estimation algorithm using only 2D human pose estimation and trigonometric identities. The constraints to the subject during video capture are identical to the ones indicated for our post-processing steps. Once again, the length of the femur and tibia are deduced in the first frame and represented by l_{femur} and l_{tibia}, respectively. For each subsequent frame, we determine the perceived femur and tibia length of each leg, m_{femur} and m_{tibia}. By assuming that the hip is the anchor of each leg, we then calculate the depth of the knee z_{knee} using Equation (1) and the depth of the ankle z_{ankle} using Equation (2). See Fig. 3.

$$z_{knee} = l_{femur} * sin(cos^{-1}(\frac{m_{femur}}{l_{femur}})) \tag{1}$$

$$z_{ankle} = l_{tibia} * sin(cos^{-1}(\frac{m_{tibia}}{l_{tibia}})) - z_{knee} \tag{2}$$

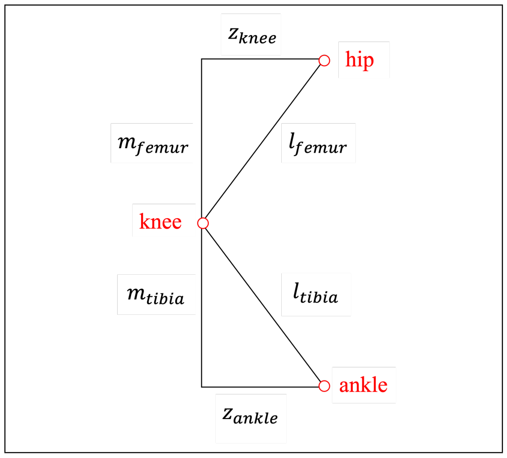

Fig. 3: The knee and its associated variables for RTJDE

III. RESULTS

A. Experimental setup

Our subject was a healthy volunteer (Male, 183cm, 22 years old) with no prior history of ACL injuries. The experimental procedures described in this paper are compliant with NSERC USRA ethics requirements. The RGB video was recorded using the Microsoft Kinect v2 sensor placed at a distance of between 2.5 to 4 meters from the subject. The color camera produces 30 frames per second video at 1920 x 1080 pixels.

As mentioned previously, the Kinect's skeletal tracking serves as our ground truth. After a thorough comparison of the Kinect's skeletal tracking data and the corresponding video, we witnessed some prediction errors during rapid motion. Particularly, we saw predictions of coronal knee angles of up to 72 degrees which is incoherent with the corresponding video frames (e.g., see Fig. 4). Therefore, we set an upper bound of 20 degrees to the Kinect's coronal prediction of each frame.

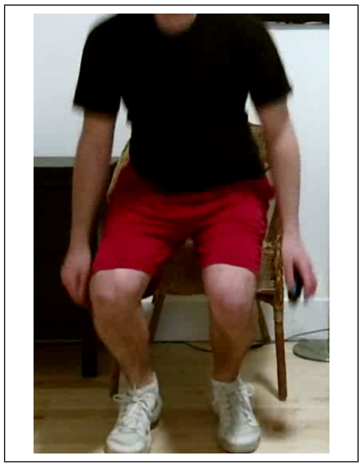

Fig. 4: Example frame of coronal prediction >72 degrees

As can be seen in Table I, a 2D human pose estimator such as TensorFlow Lite's Posenet [12] can deduce 2D human pose in real-time on modern mobile devices. Additionally, we observed run-times on our machine of 0.05-0.08 seconds for 250-300 frame videos when performing depth estimation with our technique. By combining these two results, we can conclude that we have a practical and relatively accurate variant of 2D human pose estimation that can run in quasi-real-time.

TABLE I: Tensorflow Lite PoseNet Runtime [15]

Device	GPU	CPU
Pixel 3 (Android 10)	12ms	31ms
Pixel 4 (Android 10)	12ms	19ms
iPhone XS (iOS 12.4.1)	4.8ms	22ms

B. Accuracy

When evaluating our proposed methods, we focused on the sagittal and coronal view of the knee. As mentioned previously, these are the main indicators of ACL injury risk. When discussing error in our experiments, we refer to the mean difference of degrees of the estimated knee angle from the Kinect's skeletal tracking prediction over the motion sequence.

Table II gives our analysis of the mean error for either all frames (A)) or just the contact frames (B)) of a motion sequence. The contact frames contain the initial contact frame and the peak contact frame, which are vital to ACL injury risk assessment. Here, we present results for VideoPose3D, RTJDE using either Detectron2 (with relatively slow inference time) or PoseNet (with quasi-real-time running time) for the input 2D COCO joint positions, and DeepLEAD. For VideoPose3D and RTJDE we also applied IK for post-processing to refine the 3D joint position. Since DeepLEAD directly estimates the sagittal angles for the knees, post-processing could not be applied.

For VideoPose3D, for all frames (A), post-processing results in less error for the left and right sagittal angle predictions. Indeed, with IK post-processing, for all or contact frames, VideoPose3D resulted in the least mean error for the sagittal angle predictions. For the original VideoPose3D, for all frames, the coronal errors are relatively low at 2-3%. Applying post-processing actually lead to an increase in the error of more than double on each side for the coronal view. This error seems to be caused by some noise in the data propagated into subsequent frames by inverse kinematics as it uses past configurations when searching for an optimal solution for the current frame. For the contact frames (B)), though, the reverse occurs–the original VideoPose3D had larger errors, which were greatly reduced by applying IK post-processing. Indeed, for contact frames, the best overall performance for both sagittal and coronal angle predictions were achieved using VideoPose3D with post-processing, but at the cost of very long inference times.

The performance of RTJDE using Detectron2 roughly mirrors the performance of VideoPose3D (with and without using post-processing) with only marginally higher errors. For RTJDE using PoseNet, for all frames, the performance is comparable to that of RTJDE using Detectron2. For just the contact frames, the performance is still comparable except for larger error for the coronal angle predictions. For RTJDE with PoseNet, the post-processing step didn't improve the average errors for either all frames or just contact frames.

IV. CONCLUSION

Compared to VideoPose3D, RTJDE has three main advantages: 1) it is possible to easily swap in faster human pose estimation models which output COCO key points, 2) video can be down-sampled to speed up inference, and 3) the depth estimation runs in real-time. An advantage over DeepLEAD is that, instead of calculating the sagittal angle of the knee, we infer the joint positions, which can then be input to our post-processing steps for correction.

In this study, developing a methodology to address the two major problems with current 3D human pose estimation models, inference rate and accuracy, was addressed.

To our knowledge, this is the first study to demonstrate that we can begin transitioning from high-end machines and sensors to mobile devices, which are more accessible and easy to use, for athlete jump mechanics assessments.

ACKNOWLEDGMENT

We thank Dr. Paul Martineau (McGill University) for discussions on the clinical aspects of the presented work.

REFERENCES

[1] T. E. Hewett, G. D. Myer, K. R. Ford, *et al.*, "Biomechanical measures of neuromuscular control and valgus loading of the knee predict anterior cruciate ligament injury risk in female athletes: A prospective study," *The American Journal of Sports Medicine*, vol. 33, no. 4, pp. 492–501, 2005.

[2] A. P. Toth and F. A. Cordasco, "Anterior cruciate ligament injuries in the female athlete," *The Journal of Gender-Specific Medicine*, vol. 4, no. 4, pp. 25–34, 2001.

[3] K. Otte, B. Kayser, S. Mansow-Model, J. Verrel, F. Paul, A. U. Brandt, and T. Schmitz-Hübsch, "Accuracy and Reliability of the Kinect Version 2 for Clinical Measurement of Motor Function," *PLoS One*, vol. 11, no. 11, p. e0166532, 2016.

[4] N. Karatzas, J. Corban, S. Bergeron, T. Fevens, L.-N. Veilleux, and P.-A. Martineau, "From the gaming console to the field: Using the microsoft kinect as a portable and accurate tool for assessing of jumping dynamics," *Deutscher Kongress für Orthopädie und Unfallchirurgie*, vol. 158 (S 01), pp. DKOU20–318, Oct. 2020.

[5] D. Pavllo, C. Feichtenhofer, D. Grangier, and M. Auli, "3D human pose estimation in video with temporal convolutions and semi-supervised training," *CoRR*, vol. abs/1811.11742, 2018.

[6] X. Zhou, Q. Huang, X. Sun, X. Xue, and Y. Wei, "Towards 3D human pose estimation in the wild: a weakly-supervised approach," in *IEEE International Conference on Computer Vision*, pp. 398–407, 2017.

[7] J. Martinez, R. Hossain, J. Romero, and J. J. Little, "A simple yet effective baseline for 3D human pose estimation," in *IEEE International Conference on Computer Vision*, pp. 2640–2649, 2017.

[8] P. Chalangari, T. Fevens, and H. Rivaz, "3D human knee flexion angle estimation using deep convolutional neural networks," in *42nd Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 5424–5427, 2020.

[9] T. Lin, M. Maire, S. J. Belongie, *et al.*, "Microsoft COCO: common objects in context," *CoRR*, vol. abs/1405.0312, 2014.

[10] P. Chalangari, "Deep learning and trigonometric adjustment in estimation of lower extremity angles," Master's thesis, Concordia University, Montréal, QC, Canada, 2020.

[11] Z. Cao, G. Hidalgo Martinez, T. Simon, S. Wei, and Y. A. Sheikh, "OpenPose: Realtime multi-person 2D pose estimation using part affinity fields," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

[12] G. Papandreou, T. Zhu, L.-C. Chen, S. Gidaris, J. Tompson, and K. Murphy, "Personlab: Person pose estimation and instance segmentation with a bottom-up, part-based, geometric embedding model," 2018.

[13] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick, "Detectron2," 2019.

[14] E. Coumans and Y. Bai, "PyBullet, a Python module for physics simulation for games, robotics and machine learning." http://pybullet.org, 2016–2019.

[15] Tensorflow, "Pose estimation | TensorFlow Lite." Table:Performance Benchmark.

TABLE II: Mean Difference of Degrees Compared to Kinect Skeletal Tracking for A) All Frames and B) Contact Frames.

	A) All Frames (287 frames)				B) Contact Frames (18 frames)			
Method	Left Sagittal	Right Sagittal	Left Coronal	Right Coronal	Left Sagittal	Right Sagittal	Left Coronal	Right Coronal
VideoPose3D	9.82	7.21	3.12	**2.31**	27.85	27.09	15.49	5.55
VideoPose3D with IK	**7.69**	**6.15**	6.93	4.97	**8.52**	**7.16**	1.6	**1.48**
RTJDE (Detectron2)	16.95	12.29	**2.8**	3.48	11.16	12.66	12.21	15.39
RTJDE (Detectron2) with IK	14.78	10.06	6.84	6.25	10.63	12.81	**1.11**	6.48
RTJDE (PoseNet)	9.74	14.93	4.87	4.88	8.54	14.68	17.09	13.45
RTJDE (PoseNet) with IK	9.4	14.85	4.79	4.87	8.39	14.35	17.19	13.42
DeepLEAD	14.2	9.58	N/A	N/A	21.89	19.76	N/A	N/A