Realtime Indoor Workout Analysis Using Machine Learning & Computer Vision

[1]Amit Nagarkoti*, Revant Teotia*, Amith K. Mahale and Pankaj K. Das

***Abstract*— These days there are thousands of workout videos available on the internet. *Samsung Health*[2] [1] provides a dedicated section called *programs* containing short workout videos for various exercises. The goal is to assist people perform these workouts independently on their own. A common observation is that even people who visit gym regularly find it difficult to perform all steps (*body pose alignments*) in a workout accurately. Continuously doing an exercise incorrectly may eventually cause severe long term injuries. To help solve this problem and provide assistance in form of a visual feedback while performing a workout, we propose a system to analyze user's body posture during a workout and compare it to a professional's reference workout. We represent human body as a collection of limbs and analyze angle between limb pairs to detect errors and provide corrective action to the user. Our system builds on the latest advancements using deep learning for human body pose estimation. We use techniques for time series data alignment like DTW [2] (Dynamic Time Warping) along with optical flow tracking to synchronize user/reference videos. We are able to detect and locate errors in user's activity (*pose*) very effectively based on some threshold deviation between the limb angles. The system in future can be extended to be used by physicians to monitor patient's recovery following an injury.**

I. INTRODUCTION

With the abundance of workout videos available on the internet and with popular fitness tracking apps like *Samsung Health* providing dedicated workout programs section, more people are following these workouts independently. Due to busy lifestyle and financial issues large number of people have adopted working out indoor independently. Although convenient, if the workout is not performed properly it may lead to severe injuries in long term. There is currently no system to tell a user how accurately was he/she able to follow a particular workout.

We propose a system that can be used with any camera based device capable of streaming the user workout to analyze and detect errors. The system is robust enough to handle out of sync user and reference videos, along with any camera positioning artifacts (tilt, distance from the user, user position in camera frame). Our system uses techniques from Deep Learning and Computer Vision to tackle the problem. We start with a trainer recording his/her workout video which may be a single rep (a repetitive sequence of body alignments involved in an exercise) or a set of reps. We represent i^{th} workout W_i having t frames as

$$W_i = \{sequence\, of\, frames\, f_i\} \tag{1}$$

where f_i is the i^{th} frame of workout W_i. With high frame rate videos not all frames will be equally relevant so a threshold can be defined such that a workout is a sequence of key-frames where

$$keyframes\, W_i = \{frame_i \mid |frame_{i+1} - frame_i| \geq \delta\} \tag{2}$$

Here δ is some threshold distance between two frames belonging to workout W_i such that if distance is less then δ the subsequent frame can be ignored. This can help reduce memory usage for storing trainer (reference) videos. Distance calculation is based on limb pair angle between consecutive frames and will be defined in subsequent sections. Having the set of both trainer and user workout frames, the task is now reduced to comparing these frames such that a user $frame_U$ is compared with the closest matching trainer $frame_T$. These closest matching frame pairs are found using the DTW algorithm which is a widely used algorithm for sequence alignment problems. The paper is organized into three major sections where we first discuss the background techniques and how they apply to the proposed solution, second we discuss the complete integrated pipeline in brief. Finally the results and conclusion follows where we discuss the outcome and limitations of the proposed method.

II. BACKGROUND AND TECHNIQUES

This section discusses the background techniques along with how they apply to the proposed pipeline.

A. Pose Estimation

A critical step of our process is to efficiently and accurately identify the human body parts (pose). We estimate human pose using the model proposed by Cao et al. [3] in *Real time Multi-Person 2D Pose Estimation using Part Affinity Fields*. The network uses first 10 layers of VGG-19 net [4] to obtain a fixed size vector representation for a given image, which is followed by two multi-step branches of CNNs [5] (Convolutional Neural Networks) where the first branch is used for predicting confidence maps for body part (point) locations as represented by dots in Fig. 1. Body parts include neck, elbows, wrists, shoulders, knee, hips, eyes and ears. The second branch predicts Vector Field maps which helps in deducing the association between the body parts

* contributed equally

[1]All authors are affiliated to Digital Health, Samsung Research Institute, Bangalore, 560037, India. Author email ids in original sequence for future reference amit.nagarkoti@outlook.com, r.teotia@samsung.com, amith.m@samsung.com and das.pankaj@samsung.com

[2]Fitness tracking app from Samsung

[3]All colleagues who volunteered for the activity at Samsung Research Institute, Bangalore, 560037, India

978-1-5386-1311-5/19/$31.00 ©2019 IEEE

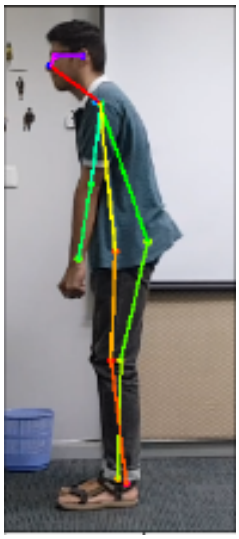

Fig. 1. Key-Points and Limbs on a user image

obtained from first branch using bipartite graph matching. Once we obtain the key body points we represent the human body as a collection of limbs defined by the key-points as shown in Fig. 1. An optimization to consider is that not all limbs will be used when performing a workout, only limbs involved in some motion will be used for the analysis part. Table I presents an elaborate list of the limb pairs. The angle between these limb pairs will be used in further steps.

TABLE I

A LIMB PAIR IS DEFINED BY CONNECTING BODY PART IN COLUMN 1 TO BODY PART IN COLUMN 2 AND BODY PART IN COLUMN 2 TO BODY PART IN COLUMN 3. EXAMPLE NECK RIGHT-SHOULDER RIGHT-ELBOW FORMS A LIMB

Body-Part-1	Body-Part-2	Body-Part-3
Nose	Neck	Right Shoulder
Nose	Neck	Left Shoulder
Neck	Right Shoulder	Right Elbow
Neck	Left Shoulder	Left Elbow
Right Shoulder	Right Elbow	Right Wrist
Left Shoulder	Left Elbow	Left Wrist
Neck	Right Hip	Right Knee
Neck	Left Hip	Left Knee
Right Hip	Right Knee	Right Ankle
Left Hip	Left Knee	Left Ankle

B. Optical Flow Tracking

Since the trainer video is available offline we can for each frame in trainer video run the pose estimation model offline to extract all body parts as a pre-processing step. This meta-data can now be transferred to the local user device (mobile) for performing on-device analysis. For a real-time system we cannot run CNN on each user frame to obtain body parts for the user video. It will slow down the feedback time and also its resource heavy to run forward pass of CNN model on each user frame. To reduce latency for user feedback we will extract the body parts from user video at only say every kth (every $5th$ or $10th$) frame and for the intermediate frames, optical flow tracking is used to obtain the body parts from neighboring frames. We are using Lucas-Kanade algorithm [6] as implemented in OpenCV [7] for optical flow tracking. Lucas-Kanade is based on intensity constancy between two intermediate frames. Optical flow provides the following advantages

- On device estimation of intermediate key-points (body parts).
- Forward and backward tracking to reduce errors in estimation.
- A missing key-point in one frame can be approximated using one in the neighboring frames.

C. DTW (Dynamic Time Warping)

DTW is a commonly used algorithm for comparing (aligning) a pair of time series sequences, applications involve analyzing walking styles of people with different walking speeds or speech detection when speakers have different word rates. Fig. 2 shows an example of DTW to align two wave sequences. DTW requires similarity measure to be defined between any two points/items (frames in our case) in a pair of sequences and returns a mapping of closest matching point/item pairs. In our case the two sequences of points (frames) are obtained from user and reference videos. We define the distance metric between frames assuming each frame has a single person as

$$d_{ij} = \sum_{K} |limbpairangle_U^k - limbpairangle_T^k| \quad (3)$$

where d_{ij} is distance between frame i in user video and frame j in trainer video. K is set of *limb pair angles* (i.e. angle between the pair of limbs) which are used in the current workout W_i. $limbpairangle_U^k$ is the angle between kth user limb pair and $limbpairangle_T^k$ is the angle between kth trainer limb pair. Since not all limbs will be involved in a particular workout so only limbs which have some significant movement are considered. Fig. 3 is a plot of similarity calculated by DTW on a sample trainer/user sequences. The algorithm is able to handle cases where multiple frames from one sequence are mapped to a single frame in another sequence (for example a steady user during beginning of an exercise).

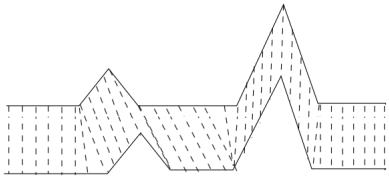

Fig. 2. Alignment of two sample sequences using DTW. Taken from [8]

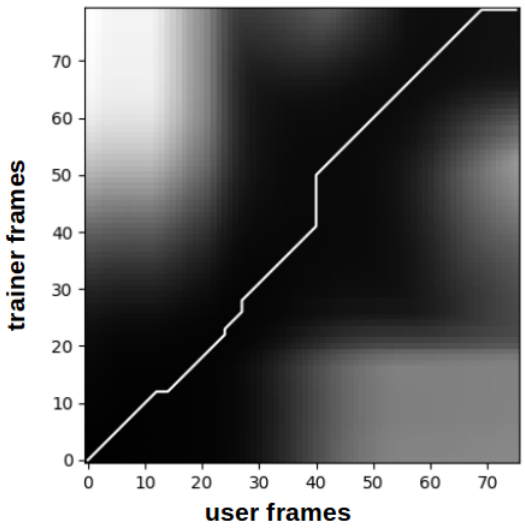

Fig. 3. A DTW frame similarity plot. X-axis shows user frames, Y-axis the trainer's frames. A horizontal line means more than one of user's frame matched trainer's single frame. A vertical line means more than one of trainer's frame matched user's single frame. A line not parallel to either axis denotes a series of matching frames between two sequences.

Fig. 4. User key-points (left) are projected to trainer's frame (right). In left image green lines are used for users limbs. In right image red lines are the trainer's limbs and green lines are the projected user's limb. We can clearly see deviations in hand position where user's hands are slightly farther and knees are bent much more than expected.

D. Affine Transformations

Once we obtain a map of which frame pairs are to be compared after the sequence alignment. Next step is how to analyze and present results to the end user. We need to take care of following issues which may arise

- User and trainer might have different body ratios.
- The user's camera might be tilted.
- Camera to person distance might be different in user and trainer videos.

To handle these problems we can transform the user points to trainer reference frame (or vice verasa) and overlay the skeleton on the trainer's frame. The affine transformation matrix is obtained by solving the least squares problem as suggested in [9]. The objective is to find matrix transformation A such that

$$AX = Y \tag{4}$$

Where X is the user's body part co-ordinates and Y is the trainer's body part co-ordinates. The optimization being

$$\min \|AX - Y\|_2 \tag{5}$$

Fig. 4 presents an example of affine-transform applied to a pair of sample frames.

III. System Overview

A trainer records the workout which is uploaded to a cloud service and body part co-ordinates are extracted using Deep Learning CNN model. Trainer's video with extracted meta-data is sent to the user device. Next a user performs and records a workout using some device (mobile or TV). The body parts are extracted for every kth user frame on the server. For intermediate frames, body-parts are calculated on user device using *optical flow tracking*. Next *DTW* is used to sync user and trainer videos and obtain frame pair mapping to be compared. Now once we have key-frames mapping for comparison we can evaluate errors and at the same time use affine-transformations to project trainer's body parts to user's frame and provide visual feedback. Error is calculated using *limb pair angles* for the matching frames and a visual feedback is shown when the deviation for any involved *limb pair* (say for $limbpair_{ij}$ comprising $limb_i$ and $limb_j$) is greater than some threshold angle δ. Fig. 5 represents the overall process flow.

IV. Results

We used workout videos having a frame rate of $25fps$. User's activity was recorded using a laptop and an external webcam at a resolution of $1280X720$. We were able to process 10 seconds of user video on our laptop with GTX 920M in approximately 30 seconds which can be improved with better hardware. Since we don't have any prior baselines for this work we evaluated effectiveness of the method based on how many observed frames had $deviation > \delta$, here $deviation$ is based on total, average or maximum angle between the limb pairs and $delta$ is some threshold on this angle. We worked with 5 volunteers with varying physique and lifestyle during the development and analysis phase. During experiments the trainer's (reference) video was played and the user was asked to repeat the workout sequence (rep) once he/she was comfortable with the workout. Error ($deviation$) was calculated on three measures, maximum deviation for any involved limb pair, total deviation for all involved limb pairs and average deviation for all involved limb pairs. We considered 3 different values for deviation angles as $25\,^\circ$, $45\,^\circ$ and $60\,^\circ$. The predicted error frames were verified using plots as in Fig. 6. From our analysis we

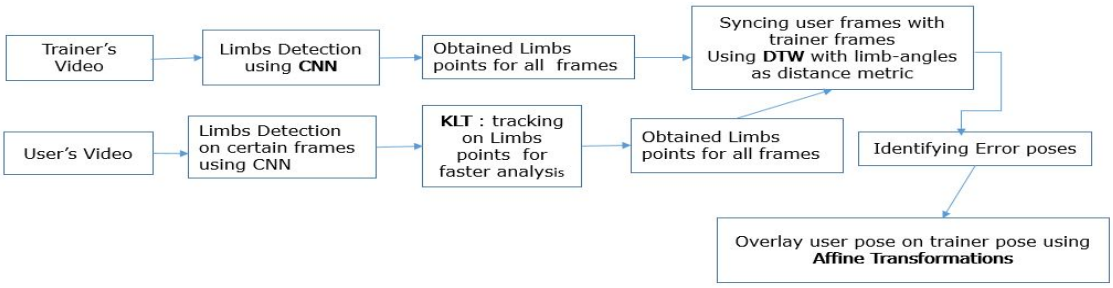

Fig. 5. System Flow Overview

Fig. 6. Analysis from two Exercises: Exercise 1 (Top) and Exercise 2 (Bottom) for a test user. Green lines are user's limb and Red lines are trainer's limb projected on user. Deviations indicate error.

observed that taking max deviation criteria most effectively identified the incorrect limb pair. The effectiveness of the system is highly dependent on the DTW algorithm for frame matching. Qualitatively the volunteers were very satisfied with the analysis presented.

Fig. 6 presents the qualitative results. A higher overlap of user and trainer limbs in *Exercise 1* indicates user was able to perform the workout quite accurately and a higher mismatch in *Exercise 2* indicates poor user performance.

V. Conclusion

In this paper we presented a system for monitoring of user workouts without any involvement of a personal trainer. The system is able to detect minute errors in limb positions which could be critical in many exercises. This presents a use case for patient monitoring by physicians where a physician can get the patient's progress report from the system. Coming to the limitations and improvements, the CNN model used was trained on the COCO [11] dataset for human pose estimation, further fine-tuning using data targeted towards physical workouts might yield much better accuracy for body-parts detection. Also currently the system only works for motion along 2 dimensions but an extension to expand the system to 3 dimensions is very much feasible using extra camera for top view or some depth perception mechanism. Further future improvements can be aimed towards optimizing the CNN model for faster inference.

Acknowledgements

We would like to thank all the colleagues[3] at Samsung who volunteered for the activity.

References

[1] Samsung Electronics America. (2018). "Samsung Health" Internet: https://www.samsung.com/us/support/owners/app/samsung-health, [Dec. 10 2018].
[2] Müller, M., 2007. Dynamic time warping. Information retrieval for music and motion, pp.69-84.
[3] Cao, Z., Simon, T., Wei, S.E. and Sheikh, Y., 2016. Realtime multi-person 2d pose estimation using part affinity fields. arXiv preprint arXiv:1611.08050.
[4] Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." arXiv preprint arXiv:1409.1556 (2014).
[5] LeCun, Y., Bengio, Y. and Hinton, G., 2015. Deep learning. Nature, 521(7553), pp.436-444.
[6] Bouguet, Jean-Yves. "Pyramidal implementation of the affine lucas kanade feature tracker description of the algorithm." Intel Corporation 5.1-10 (2001): 4.
[7] Bradski, Gary, and Adrian Kaehler. "OpenCV." Dr. Dobb's journal of software tools 3 (2000).
[8] Programminglinguist, "File:Dynamic time warping.png". Internet: https://commons.wikimedia.org/w/index.php?curid=41617143, July. 15 2015, [Dec. 10 2018].
[9] Unser, Michael, Matthew A. Neimark, and Chulhee Lee. "Affine transformations of images: a least squares formulation." Image Processing, 1994. Proceedings. ICIP-94., IEEE International Conference. Vol. 3. IEEE, 1994.
[10] Single Pose Comparison a fun application using Human Pose Estimation, Internet: https://becominghuman.ai/single-pose-comparison-a-fun-application-using-human-pose-estimation-part-2-4fd16a8bf0d3.
[11] Lin, Tsung-Yi, et al. "Microsoft coco: Common objects in context." European conference on computer vision. Springer, Cham, 2014.