Virtual Training Coach

Giulia Bertazzini
giulia.bertazzini@stud.unifi.it

Niccolò Guiducci
niccolo.guiducci@stud.unifi.it

Abstract

In this paper, we used MeTRAbs tool to extract 3D skeletons from some videos representing fitness exercises and we compared the skeleton joints, used as landmark to represent the pose of a person in the shape space, to establish which parts of the exercise are not correctly executed. We implemented three metrics based on different approaches to detect errors: in particular, we find the most wrong joints in each error frame, the most wrong joint in an exercise repetition and finally the most wrong joint in an entire exercise.

1 Introduction

Human pose estimation from camera input is a longstanding problem with a wide range of applications. In the last years it became interesting to apply it to fitness, in order to check if an exercise is correctly executed or not.
In this project, we used 3D skeletons extraction techniques to evaluate the correctness of fitness exercises from home-made videos, which have been previously analysed by other students [2] using 2D skeletons. To that end, we compared the estimated poses of a trainer with the one of a user for the same exercise to establish which parts of the exercise are not correctly executed; in particular, to evaluate the user performance, after identifying the wrong poses, we search for the most wrong joints in each error frame (the ones that exceed a certain threshold), and, thanks to this, the most wrong joint in an exercise repetition and the most wrong joint in an entire exercise. To make comparisons in order to detect these errors, we defined some different metrics, as detailed in the section below 4.

2 Skeleton Body Representation

To extract 3D skeleton, we firstly used Metro-Pose3D tool, but since that it didn't work well enough with some type of exercise and some viewpoint of the camera, we used an extended journal version of it, **MeTRAbs** [4]. MeTRAbs' models predict the 24 joints of the SMPL body model [3], which is a realistic 3D model of the human body based on skinning and blend shapes. All the skeleton joints, represented by three coordinates $\mathbf{J} = (x, y, z)$, are enumerated in the figure below.

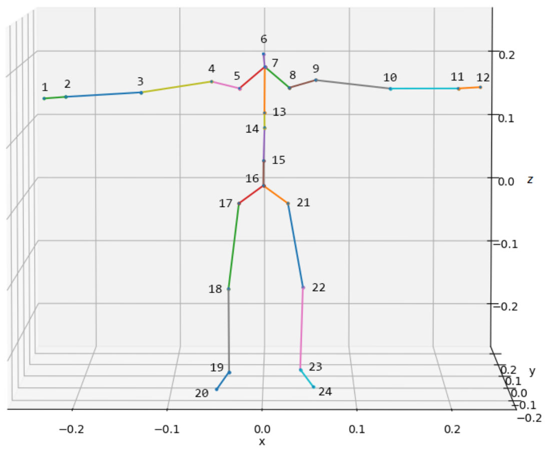

Figure 1: Skeleton body representation extracted with MeTRAbs - SMPL body model (24 joints)

To better analyze exercises of different types, the joints of the body have been divided into two categories: joints from 1 to 12 belong to upper body part, joints from 13 to 24 belong to lower body part.

3 Dataset

In this section we described the dataset provided to us and some necessary operations to prepare it for MeTRAbs and to make comparisons between skeletons.

3.1 Dataset description

To reach the goal of this project, we used a dataset built by other students (see [2]). It is composed by 6 types of exercises: **arm-clap, dumbbell-curl, single-lunges,**

double-lunges, squat and **push-ups**. The samples are obtained from the videos of 16 people plus one trainer (used as term of comparison) and they have been processed frame by frame.
In particular, we considered three different categories for the exercises, as suggested in the paper:

- **upper body exercises**: in this category only the upper body part is involved and, consequently, only the joints from 1 to 12. Arm-clap and dumbbell-curl belong to this type of exercise.
- **lower body exercises**: on the contrary, in this exercises only the lower body part is involved and so only the joints from 13 to 24. Single-lunges, double-lunges and squat are in this category.
- **full body**: the execution of this exercises, as a push-up, involves the movement of all body and so requires all 24 joints from the skeleton.

3.2 Image preprocessing

To speed up skeleton generation, it was necessary to use the **predict_multi_image** method of MeTRAbs, which requires RGB images of the same dimensions; for this reason we resized each frame, adding a padding to make the image square without changing the aspect ratio.
Once resized all frames, we generated the 3D skeleton giving in input to **predict_multi_image** not all frames, but one out of 5, to save space and time. This was possible since that the movements in each video are slow enough, and so discarding 4 consecutive frames there is not an effective loss of information.

3.3 Normalization

Since that the 3D skeletons extracted with MeTRAbs depend on the camera position, we addressed coordinate normalization as described in [1]. According to this paper, variation in pose estimate due to scale, location and rotation are modelled as linear transformations, considering the vectorized form of a pose estimate.
For this reason, given a pose estimate with n joints, $\widetilde{x} = (J_1, ..., J_n)$, we compute a reference point p_c as the mean vector of the two hip joints and the pelv joint; moreover, scale is normalized by standardizing $\widetilde{x}$ to unit norm. Concerning to rotational variation, it is approximated by estimating the camera pose with respect to a fixed world coordinate system, as suggested in the paper. To that end, we considered J_{cl} the left clavicle position of a centered pose estimate (joint 8 in Figure 1); an orthogonal vector to J_{cl} in the direction of the right clavicle J_{cr} (joint 5 in Figure 1) is then estimated as

$$J_{cl}^{\perp} = J_{cr} - (\frac{(J_{cl})^T}{||J_{cl}||_2} J_{cr}) J_{cl}$$

where $J_{cl}^{\perp}$ is an orthogonal vector to J_{cl}. The last orthogonal vector is estimated through the cross product between J_{cl} and $J_{cl}^{\perp}$:

$$(J_{cl}, J_{cl}^{\perp})^{\perp} = J_{cl} \otimes J_{cl}^{\perp}$$

The orthonormal version of the above three vectors, $M = (J_{cl}, J_{cl}^{\perp}, (J_{cl}, J_{cl}^{\perp})^{\perp})$, constitute the camera position estimate with respect to a fixed world coordinate frame.
Finally, a given pose estimate is standardized to a fixed coordinate orientation as follows $\widetilde{x} = M^T \times \widetilde{x}$ where $(\cdot)^T$ denotes matrix transpose.

4 Metrics

To make comparisons between 3D skeletons, we implemented four metrics based on different approaches.

4.1 Euclidean distance

The simplest and most intuitive approach to see how far two skeletons are is to calculate the euclidean distance between all 24 matching joints, each one described by three coordinates. In this way, if two skeletons are similar, the euclidean distance between them will be very small (approximately around zero), and the bigger it is the greater the difference.
Considering that we want to compare 3D skeletons of different people (a trainer and a user), two skeletons can never be exactly the same and due to that we introduced a threshold, below which the skeletons are considered equals (with a zero distance between them).

4.2 Angles distance

An other method to figure if two skeleton are in a similar position, is to consider the angles, which are computed between joints, as suggested in [2]. Unlike the paper, since that the skeleton are in 3D and not in 2D, we computed more angles between the joints (see Figure 1 for joints number); more specifically, we considered 21 angles in total, divided in three categories based on the exercise category:

- **upper body angles** (11 angles): (7,14,16), (13,7,9), (8,9,10), (9,10,11), (13,7,4), (3,4,5), (2,3,4), (4,5,7), (6,7,8), (5,7,6), (7,8,9);
- **lower body angles** (12 angles): (13,14,15), (14,15,16), (15,16,17), (16,17,18), (17,18,19), (18,19,20), (15,16,21), (16,21,22), (21,22,23), (22,23,24), (7,16,17), (7,16,21)
- **full body angle**s: all 21 angles

To find parts of an exercise not correctly executed, we computed a distance which calculate the difference, in absolute value, between matching angles of a trainer skeleton and a user skeleton. Same as euclidean distance, we introduced a threshold below which we can consider two skeletons equals.

4.3 Combined distance

Due to the fact that angles in 3D are rotation invariant, a better approach might be to introduce a combined version of the two methods previously presented, considering two angles and an euclidean distance to estimate how different two skeletons are. To that end, we transformed the Cartesian coordinates into spherical coordinates, where a point is represented by a triple (r, θ, ϕ) which represents the point's distance from the origin of the system, the inclination angle and the azimuth.
To evaluate skeleton distances, we introduce a distance which calculate the euclidean distance between the matching radius of the skeletons (always between a trainer and a user skeleton) and an angle distance between corresponding angles of the skeletons. Even in this case, we consider the same threshold as before.

4.4 GRAM Matrices distance for skeletons)

An alternative method is to represent a body pose through the Gram matrix of joint coordinates:

$$G = \Gamma\Gamma^T \in \mathbb{R}^{n\times n}$$

where n is the number of joints (24) and Γ is a matrix 24×3 that represent one pose. Unlike the other metrics, this one can be only use to calculate the distance among two skeletons and not also for distance between matching joints.

5 Sequences alignment

Each exercise has a different number of repetitions; due to that we had to identify these repetitions in order to then apply the Fast Dynamic Time Wrapping (FastDTW) to each sequence, corresponding to a single repetition.
To that end, for each exercise we considered a reference skeleton (the one corresponding to the first frame) and, calculating the distance between two skeletons, we looked for the more similar skeletons to the reference one (corresponding to the beginning of a new repetition).
To evaluate the distance between two skeletons we can use one of the metrics defined in 4.

6 Error Detection

First of all, we looked for frames corresponding to some errors. To that end, after aligning the skeleton sequences of the trainer and of a user, we calculated the distances between corresponding skeletons with one of the metrics defined. Even in this case, we introduced a threshold below which an user skeleton is not considered wrong.
Once the error frames have been found, we focused on detecting which parts of the exercise are not correctly executed. In particular we looked for three different type of errors:

- the most wrong joints in each error frame;
- the most wrong joint in an exercise repetition;
- the most wrong joint in an entire exercise.

Regarding the first one, we calculated distances between corresponding joints (using one of the metrics previously presented, except for the GRAM Matrices distance) in each error frame and we sorted them in descending order.
Then, we showed on the user skeleton the joints which exceed a threshold (calculated as the mean distances among matching joints and multiplied for a variable number, which can be decided at runtime). As we can see in the figures in section 7, these wrong joints are marked with a bolded "X" on the skeleton: in particular a red "X" shows the most wrong joints and the orange and yellow the less ones (an orange joint is more wrong than a yellow one).
Concerning the second and the third type of error searched (cumulative errors), we associated a counter to each joint, which stores how many times that joint is considered wrong. Thanks to it, we can show the most wrong joint in an exercise repetition and in an entire exercise. Furthermore, we show the accuracy with an exercise has been executed, calculated as:

$$repetition\_accuracy = \frac{w}{N_r}$$

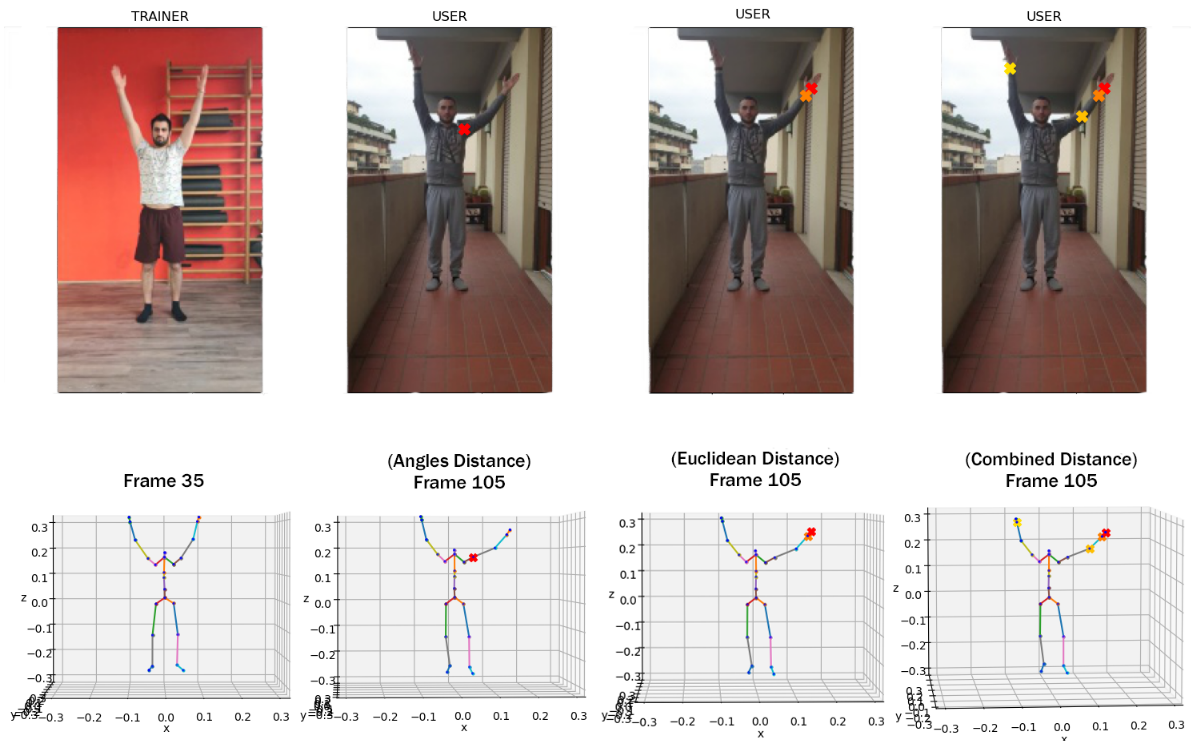

Figure 2: This figure shows one error frame for two arm claps compared with the three defined metrics.

Distance	Errors	1Rep. SR	2Rep. SR	3Rep. SR	4Rep. SR	5Rep. SR	Ex SR
Angles	16	100%	100%	97.38%	92.8%	90.84%	93.68%
Euclidean	16	89.97%	95.64%	100%	94.77%	89.53%	92.48%
Combined	16	91.71%	92.58%	100%	94.77%	86.04%	91.28%

Table 1: This table summarized the results achieved on a wrong arm-clap exercise with all metrics at the same thresholds (**Pose_thr**=1.2 and **Joint_thr**=1.4); in this case the results are quite homogeneous, probably due to the strong geometry of the exercise.

$$exercise\_accuracy = \frac{w}{N}$$

where w is the number of wrong joints, N_r is the total number of joints in the repetition, which is simply 24 multiplied for the number of frames belonging with that repetition, and equally N is the total number of joints in the exercise, which is 24 multiplied for the total number of frames of the exercise.

7 Test and Result

In this section we present the test carried out and the results obtained for each type of exercise. In particular, we report three different tests, based on the metric used. For each type of exercise we identified repetitions using the GRAM Matrices distance (see 4.4), even if also the other distances worked pretty well.

7.1 Upper body exercise

As an example of upper body exercise, we present an **arm clap**. Figure 2 shows the results achieved (for only an error frame) using all the three implemented distances to detect errors. We can notice that with the same threshold the three metrics catch different errors: indeed the angle distance marks as the most wrong joint the left shoulder, while both the euclidean distance and the combined distance mark the left wrist. These last two distances also catch a greater number of errors (2 with euclidean and 4 with combined). In table 1 are shown the other two kinds of errors and the corresponding accuracy (as described in 6) for this specific exercise.

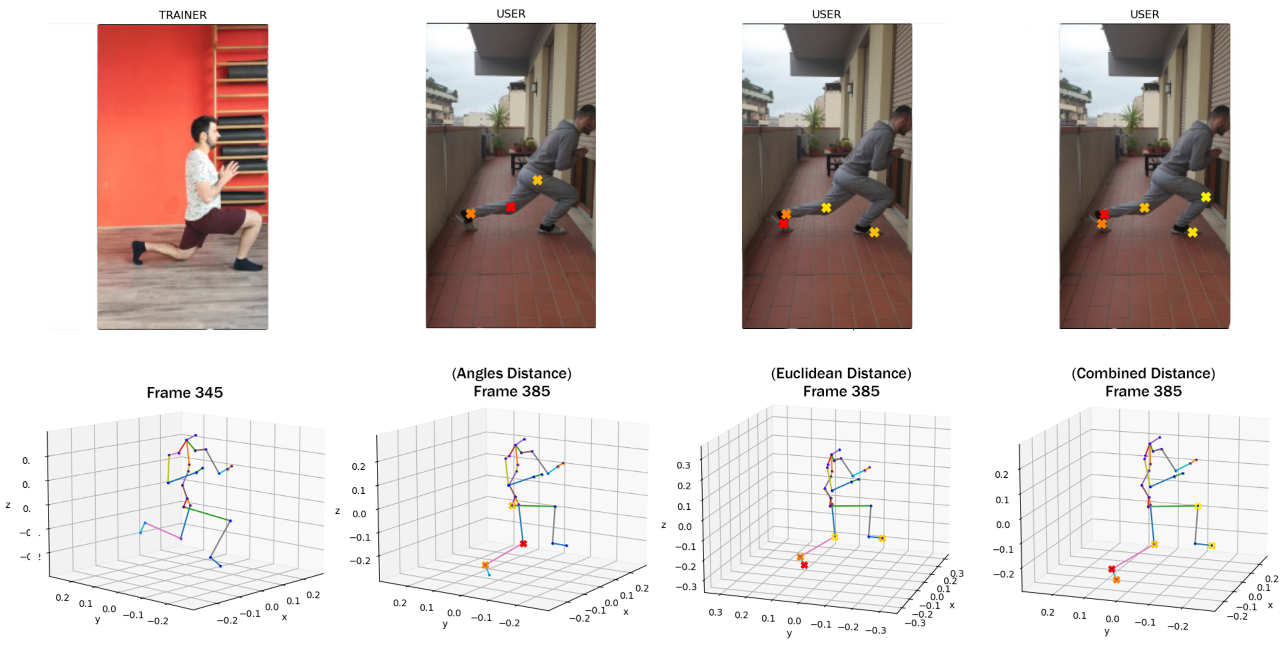

Figure 3: This figure shows one error frame for two single lunges compared with the three defined metrics.

Distance	Err.	1Rep.SR	2Rep.SR	3Rep.SR	4Rep.SR	5Rep.SR	6Rep.SR	7Rep.SR	Ex SR
Angles	24	95.33%	91.33%	92.0%	92.0%	88.67%	94.67%	98.0%	93.14%
Euclidean	37	95.33%	80.0%	91.33%	77.33%	79.33%	87.33%	88.0%	85.52%
Combined	39	92.0%	85.83%	84.0%	79.33%	79.33%	82.0%	92.0%	84.86%

Table 2: This table summarized the results achieved on a wrong single-lunges exercise with all metrics at the same thresholds (**Pose_thr**=1.3 and **Joint_thr**=1.5); since the metrics evaluate different spatial aspects they can lead to experience more or less errors depending on the exercise we are considering.

7.2 Lower body exercise

As an example of lower body exercise, we report a **single lunges**. Same as before, figure 3 shows the results obtained in one error frame using all the three distances. Even in this case, we can notice that with the same threshold the three metrics catch different errors: in particular, the angle distance marks as the most wrong joint the left knee plus other two minor errors, while the euclidean distance marks the left toe plus other three errors and the combined distance marks the left ankle plus other fours errors. In this case the distance which catch the greatest number of error is the combined one. In table 2 are shown the other two kinds of errors and the corresponding accuracy for this specific exercise.

7.3 Full body exercise

Finally, we present a **push ups** as an example of full body exercise. The results obtained for one error frame using the three metrics are shown in figure 4.
In this case the three metrics achieve very different results: with angles distance the most wrong joint is represented by neck and the other errors are mostly relative to shoulders and clavicles; on the contrary, with euclidean distance and combined distance the most wrong joint is represented by left toe. However, while combined distance is able to catch errors located on back (that seems to be the most obvious), this not happen in the case of euclidean distance, where not even a joint belonging to back is considered a mistake; with angle distance only one joint on the back is marked.
Table 3 present the other two kinds of errors and the corresponding accuracy for this specific exercise.

8 Conclusion

In conclusion, we presented three different approaches thanks to which we can establish which parts of a fitness

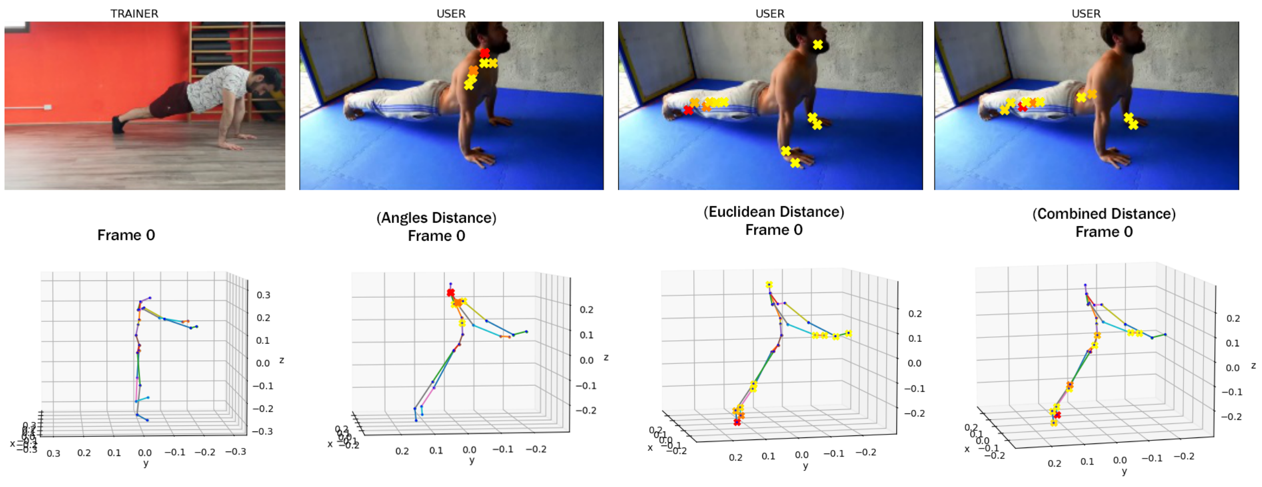

Figure 4: This figure shows one error frame for two push ups compared with the three defined metrics.

Distance	Errors	1Rep. SR	2Rep. SR	3Rep. SR	4Rep. SR	Ex Succ. Rate
Angles	10	95.01%	90.03%	97.78%	92.08%	93.91%
Euclidean	15	81.26%	76.18%	73.96%	74.52%	76.45%
Combined	20	73.96%	60.11%	58.45%	67.87%	65.1%

Table 3: This table summarized the results achieved on a wrong push-ups exercise with all metrics at the same thresholds (**Pose_thr**=1.0 and **Joint_thr**=1.1); since the metrics evaluate different spatial aspects they can lead to experience more or less errors depending on the exercise we are considering.

exercise are not correctly executed. For each exercise category, the combined distance (that consider both an euclidean distance and an angles distance) seems to be the best one. Indeed, in all the exercises the combined distance is the one that catch the greatest number of errors and the most significant ones (like the joints on the back in Figure 4).
Clearly there are some limitations: it's important that the initial pose of a user complies with the initial pose of the trainer, because otherwise, considering as example a push up, if at the beginning of the repetition the trainer has straight arms and instead the initial pose of the user to compare is lying, all the exercise will be considered wrong. This is due to the fact that when we search the exercise repetitions, we simply look for the similar skeleton to the reference one (which is the skeleton corresponding to the first frame and so depending on the beginning of the video).
Nevertheless the introduced methods work pretty well with each exercise category, remaining however preferable to use the combined distance instead of the other two. Concerning the identification of repetitions, is slightly better to use the GRAM Matrices distance, even if also the other distances correctly works.

References

[1] Girum G. Demisse, Konstantinos Papadopoulos, Djamila Aouada, and Björn Ottersten. Pose encoding for robust skeleton-based action recognition. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 301–3016, 2018.

[2] Luca Ciabini Ettore Maria Celozzi. Body skeleton action recognition.

[3] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. SMPL: A skinned multi-person linear model. *ACM Trans. Graphics (Proc. SIGGRAPH Asia)*, 34(6):248:1–248:16, October 2015.

[4] István Sárándi, Timm Linder, Kai O. Arras, and Bastian Leibe. MeTRAbs: metric-scale truncation-robust heatmaps for absolute 3D human pose estimation. *IEEE Transactions on Biometrics, Behavior, and Identity Science*, 2020. in press.