Dance With AI – An interactive dance learning platform

Deep Gojariya[1], Vatsal Shah[2], Viraj Vaghasia[3], Neha Kesho Ram[4]

[1,2,3]B.E. Student, Department of Information Technology, D. J. Sanghvi College of Engineering, Mumbai.
[4]Assitant Professor, Faculty of Information Technology Department, D. J. Sanghvi College of Engineering, Mumbai.

---***---

Abstract - *Pose estimation is a technique of identifying human poses in real time which uses computer vision. It has many applications in fields like entertainment, sports, medical,etc. Pose estimation on video applications hasn't been exploredto a greater extent yet and the systems that use pose estimation are generally based on image to image comparison. Our system on the other hand provides feedback/output to the users based on video to video matching of poses. The system is based on the pose estimation of the user and matching the user's pose with thepose of dancer in the source video. Also since it is developedfor children it will have a game kind of an interface where the users are provided with different levels and there also is a leaderboard for fair competition.*

Key Words**: Pose Estimation , Key-points, CNN (Convolutional Neural Network),FPS (Frames per second), Computer Vision, DTW (Dynamic Time Warping), PAF (Part affinity fields).**

1. INTRODUCTION

During the elementary period of learning, children are enthusiastic and spontaneous. Young children need to be active everyday. Promoting correct posture and movement plays a significant role in the development of thechild. Due to the Covid pandemic they cannot move out oftheir houses to learn new activities. Dance is the best way for children to be active and it is an activity that they love to do. We have developed a fun game for children to learn basic dance moves and score them based on their performance.

Our Proposed system allows the user to select a dance step of their choice and learn it from the reference video provided. Then he can choose to upload his/her video or to go live and perform the dance step. The real-time body key-points of the user are then captured through a webcam and stored in the database where they are compared to the reference video's pose key-points and a feedback is generated based on the similarity score.

2. LITERATURE SURVEY

2.1 Pose Estimation

Human pose estimation is a computer vision-based technology that detects and analyzes human posture. Essentially it is a way to capture a set of coordinates for each joint (arm, head, torso, etc,) which is known as a **key point** that can describe a pose of a person.[1].

A short overview of some of the human pose estimation algorithms is given below.

2.1.1 OpenPose

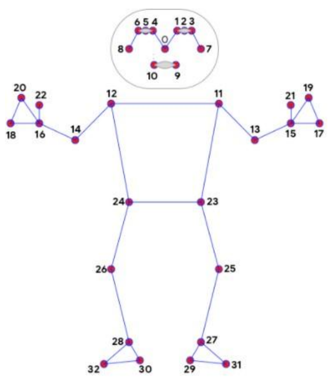
Openpose was developed by researchers at the Carnegie Mellon University and it can be considered as the state-of –the-art for detecting human poses in real time. Openpose follows an architecture in which first an image is passed through a CNN like VGG-16/VGG-19. Although we can achieve a higher average accuracy using OpenPose but on the other hand the fps obtained was very low on average machines due to it's heavy architecture.[4]

2.1.2 PoseNet

Posenet offers single-stage pose detection algorithm which can detect key-points of one human at a time and it can also be termed as a lighter version of Openpose as it uses a lighter MobilenNet CNN instead of a heavier CNN like VGG-16 or VGG-19. MobileNet models are developed for achieving higher fps during live detection and are mainly used in mobile applications. PoseNet can detect 17 key-points in a single human image. It is well-known for its average fps but it gets traded off with moderate to low accuracy.[5]

2.1.3 BlazePose

Google developed a lightweight CNN architectural model for human pose estimation called BlazePose, it can compute coordinates of 33 body key-points. Blazepose contains two machine learning models - a Detector and an Estimator. The pose estimation is done with a two-step detector tracker ML pipeline . Using detector pose region-of-interest (ROI) is first detected . The tracker then predicts 33 pose critical points from the ROI . This model isa perfect balance of both OpenPose and PoseNet as we canachieve good accuracy as well as good fps and also it can run on low-end machines.[6]



0. nose	17. left_pinky
1. left_eye_inner	18. right_pinky
2. left_eye	19. left_index
3. left_eye_outer	20. right_index
4. right_eye_inner	21. left_thumb
5. right_eye	22. right_thumb
6. right_eye_outer	23. left_hip
7. left_ear	24. right_hip
8. right_ear	25. left_knee
9. mouth_left	26. right_knee
10. mouth_right	27. left_ankle
11. left_shoulder	28. right_ankle
12. right_shoulder	29. left_heel
13. left_elbow	30. right_heel
14. right_elbow	31. left_foot_index
15. left_wrist	32. right_foot_index
16. right_wrist	

Fig -1: 33 key-points of the Blazepose algorithm.

Table -1: Comparison of Pose Estimation Techniques[4][5][6].

Parameter	**OpenPose**	**PoseNet**	**BlazePose**
Multi-stage	Yes	No	Yes
Base CNN Model	VGG16 Or VGG19	MobileNet	Single Shot Detector
Avg. Accuracy	0.79	0.75	0.67
FPS	Bad	Good	Excellent
Number of Key-points	135	17	33

Table 1 shows comparison between different pose estimation techniques based on certain parameters. We can see that the best accuracy obtained is for the OpenPose algorithm but due to its heavy CNN model the fps obtainedis quite low whereas in the case of Posenet algorithm we obtain good fps since it does not have a multi-stage architecture and also uses a lightweight CNN model. BlazePose is another great pose estimation algorithm as it has a good balance of accuracy and fps obtained during poseestimation. We decided to choose Blazepose out of the three because Openpose was not giving us a good fps whereas Posenet was considering only 17 body key-points for detection compared to 33 of Blazepose and also in our application we needed an algorithm which can give good results for single person pose estimation and Blazepose mainly focuses on that, also it gives normalized coordinates as the output whereas in other algorithms we explicitly needto normalize them for further processing.

2.2 Pose Comparison Techniques

Pose estimation algorithms when combined with pose comparison techniques can give rise to many real-world applications. The technique which compares two different poses is called as pose comparison. It compares two different poses on the basis of how similar they are or how dissimilar they are. Pose comparison requires body coordinates or key-points of two different users so that it can match them. There are several techniques to perform comparison of poses like superimposing poses , cosine similarity and dynamic time warping.

2.2.1 Cosine Similarity

Cosine Similarity is a technique for comparing poses. This method compares angles between body joints. Since these angles xare xindependent xof xphysical xlength, xit would give xgood xresults. For xcalculating xjoint angles the cosinex triangle xrule is used. xBut this methodx has a drawbackx which is that the two different poses can also have the same joint angle.eg: The elbow joint angle for hands facing inwards and outwards is the same whereas the pose is different. In order to get over this drawback, along with thex joint anglex we also checkx the coordinate position of the limbs which are connected to that joint. If for bothx the posesx we getx the same xcoordinates then we can sayx that both poses arex matching [3].

2.2.2 Dynamic Time Warping(DTW)

DTW is a fast and an efficient algorithm for measuring similarity between two sequences of videos which are of different timeframes . Similarity can be calculated by coinciding two sequences and measuring eucledian distance between them at each time interval or phase. It can handle sequences with different scale and translation.It has less effect of noise and therefore it enhances the functionality of the applications that use it [2].

For calculating the DTW score we consider two sequences X,Y of lengths n and m respectively.

Input: $X = \{x_1,x_2,x_3,....,x_n\}$, $Y=\{y_1,y_2,y_3, \quad ,y_m\}$

Now we create a cost matrix called D with dimensions (N+1),(M+1) whose first row and column get initialized by infinity.

```
Initialization:
For i=1 to N : Di,0 = ∞,
For j=1 to M : D0,j = ∞,
D0,0 = 0
```

After initialization we then calculate the cost matrix values,

```
For i=1 to N
    For j=1 to M
        Di,j = d(xi,yj) + minimum(Di-1 , j-1 , Di-1 , j , Di , j-1)
        d(xi,yj) = |xi-yj| is the Euclidean distance.
```

In this way , we can find out the similarity between two posesequences.

We have chosen DTW over cosine similarity for pose comparison due the the few advantages that the DTW algorithm provides which are i] It is independent of the sequence size i.e. both sequences need not be of samelength. ii] It is more accurate than cosine similarity.

3. PROPOSED SYSTEM

Our system is a web application which just requires a browser to run it. The system uses the BlazePose algorithm to detect human poses and DTW for pose comparison. The user first needs to select a dance step from the list of different available dance steps. When the user selects a desired dance step that he wants to imitate, he then needs toperform that dance step and our system will then compare the user's pose with reference video's pose and give him a feedback based on the similarity score obtained. To understand more about our system let's have a look into oursystem architecture.

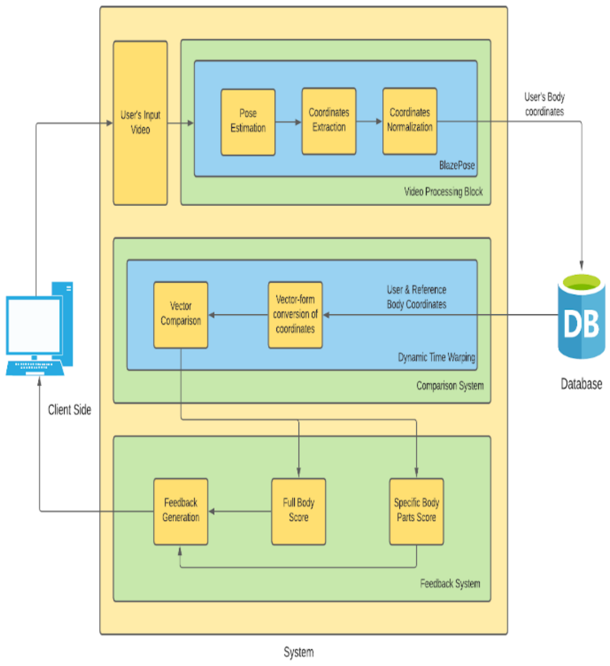


Fig-2: System Architecture

So from the client/user's side a video is fed to our system which is then passed to our video processing block which makes use of the BlazePose algorithm to estimate human pose. It also extracts and normalizes the coordinates of the human pose skeleton which are then stored in our database. We have used MongoDB as our database which stores information in the form of documents. After storing the user's body coordinates into the database we move to the video comparison block where the user's video is compared to the reference video and this is done by comparing the vectors of different body-key points of the user and reference dancer. The reference dancer's body-key points are already stored in the database. While comparing the twovectors we make use of the DTW algorithm. Later the output from the comparison system is given to our feedback system so that it can generate feedback based on the similarity scoreobtained, and finally this feedback is given back to the userso that he could improve on his dance step or try a new dance step.

4. RESULTS

We have taken here different test cases which are in the form of dance videos of three different levels and the similarity score obtained by comparing them to our reference dance videos has been listed in the table below. Slow test case is one in which the user performs a little slower than he shouldperform whereas in a fast test case he performs a little fasterthan expected . In a proper test case the user tries to replicatethe dance form correctly whereas in an improper test case he does a totally different step. Looking into the results, wecan observe that due to the DTW algorithm there is not much difference obtained across the first three test cases and the score obtained in improper test cases for three levels is very bad as the user didn't perform the same dance step as the reference video. Through these test cases we can justify that the DTW algorithm is independent of different timeframes.

Table -2: Similarity Percentage obtained based on different test cases.

Different testcases	Similarity Percentage(%)		
	Level 1	Level 2	Level 3
Slow	70	71	64
Fast	65	69	61
Proper	79	80	75
Improper	20	18	22

Table -3: Level feedback (case-passed)

Level	Body Parts	Key-points	milarit yScore	Similarity Percentage	Overall Similarity Score	Level Status
evel1	Leftleg	Left knee	10	86%	466	Passed
		Left hip	10			
	ight leg	Right knee	11			
		Right hip	11			
	Left arm	Left shoulder	11			
		Left elbow	15			
	ight arm	Right Shoulder	11			
		Right elbow	9			

Table -4: Level feedback (case-failed)

Level	Body Parts	Key-points	SimilarityScore	Similarity Percentage	Overall SimilarityScore	Level Status
Level 2	Left leg	Left knee	49	19%	3036	Failed
		Left hip	42			
	Right leg	Right knee	55			
		Right hip	57			
	Left arm	Left shoulder	26			
		Left elbow	24			
	Right arm	Right Shoulder	30			
		Right elbow	27			

Table -3,4 shows the feedback for Level 1 and 2 respectively and contains a detailed feedback which is based on the fourmain body parts that come useful while dancing i.e. the leftarm, left leg, right arm and right leg . The score for these body parts is obtained by comparing the user's body parts with the reference video body parts and based on the DTW pose comparison algorithm a similarity score is generated which in our case is of two types the overall body similarityscore and an individual body parts similarity score. The higher the similarity score the lower is the similarity percentage. Once we obtain a similarity score we can convert it into similarity percentage by using the formula :

Similarity percentage = 100 - ((similarity_score - MIN) * 100) / (MAX - MIN).

here, MAX and MIN are constants.

5. CONCLUSION AND FUTURE SCOPE

The user after providing his/her video to the system can expect a feedback on the basis of the similarity score obtained by comparing the user's dance with the reference's dance .In case of failure the system should give a feedback that he has failed the level and also will tell the user specific body parts he needs to work on to improve his score. Apart from this the system can also be improved by providing the user with a dashboard containing his/her past performances along with the score.

REFERENCES

[1] Derrick Mwiti, "A 2019 Guide to Human Pose Estimation," August 5,2019. https://heartbeat.comet.ml/a-2019-guide-to-humanpose-estimation-c10b79b64b73

[2] Salvador, Stan and Philip Ka-Fai Chan. "FastDTW: Toward Accurate Dynamic Time Warping in Linear Time and Space." (2004).

[3] Borkar, Pradnya Krishnanath et al. "Match Pose - A System for Comparing Poses." International journal of engineering research and technology 8 (2019): n. pag.

[4] Cao, Zhe et al. "OpenPose: Realtime Multi-Person 2D Pose Estimation Using Part Affinity Fields." IEEE Transactions on Pattern Analysis and Machine Intelligence 43 (2021): 172-186.

[5] Papandreou, George et al. "PersonLab: Person Pose Estimation and Instance Segmentation with a Bottom-Up, Part-Based, Geometric Embedding Model." ECCV (2018).

[6] Bazarevsky, Valentin et al. "BlazePose: On-device Real-time Body Pose tracking." ArXiv abs/2006.10204 (2020): n. pag.