Posture And Activity Analysis For Patients In Rehabilitation

Dhruv Agarwal, Donna Thomas

Abstract: Recognizing the patient posture and physical actions is the key focus on their rehabilitation to restore or enhance the functional and motor abilities of those with physical disabilities. In this paper; we focus on the gross motor skills in adults who are in rehabilitation from an injury and will present a method for recognizing their posture and activities using a sequence of RGB-D images. The shape features are extracted using the depth information in the frequency domain via spherical harmonics representation.

Index Terms: Dot Map, Heat Map, Human Posture Recognition, OpenPose, Optimum Angle, Posture Estimation, RGB-D Camera, Skeleton Data.

1 INTRODUCTION

Physiotherapy is a medical treatment used to restore movement in patients, who are unable to walk or mobilize properly. It is done by massaging the joints and muscles in the area that has lost mobility by a physiotherapist. In this healthcare practice, the posture of the patient is periodically monitored. The posture of a person is defined as "the position in which we hold our bodies while standing, sitting or lying down". This is determined by our spinal cord and the muscles around it like the hamstrings and most of our back muscles, and is imperative to the normal functioning of the rest of our body. A good posture can reduce tension in the muscles and ligaments, prevent backaches and chronic diseases like arthritis, joint pain and wear and tear of muscles. The posture of a human being can be measured using the angles between various joints located on different parts of the body, mainly the shoulder and the knees. This can be done using OpenCV and libraries like OpenPose. OpenPose is a library for real-time multi-person keypoint detection and multi-threading written in C++ using OpenCV and Caffe. The key points here refer to the joints present on the shoulders, the arms, the knees and the legs. There are a total of 135 key points stored in the library. Using these key points, a complete human skeleton the subject can be created in real-time. This can also be used to classify different activities like kneeling, folding hands or walking.

- *Dhruv Agarwal is currently pursuing bachelor's degree program in computer science engineering in Vellore Institute of Technology, Vellore, PH-8890525200. E-mail: agarwal.dhruv1234@gmail.com*
- *Donna Thomas is currently pursuing bachelor's degree program in computer science engineering in Vellore Institute of Technology, Vellore, PH-9003391053. E-mail: donnathomas312@gmail.com*

2 ARCHITECTURE

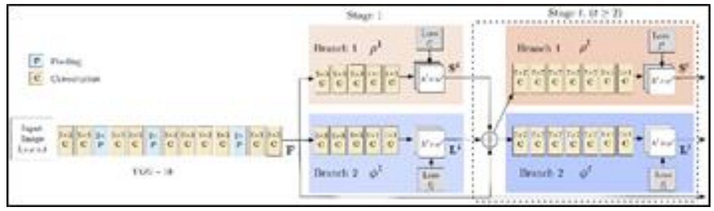


***Fig. 1**. OpenPose Architecture. Image taken from "Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields".*

2.1 Confidence Maps

In Fig. 1, the top branch of the neural network produces a set of detection confidence maps S. This is mathematically expressed as follows.

$S = (S_1, S_2, S_3 \ldots S_j)$

$S_j \in \mathbb{R}^{wxh}$

$j \in \{1 \ldots J\}$ where J is the total number of body parts

2.2 Part Affinity Field (PAF) Maps

In Fig. 1, the bottom branch of the neural network produces a set of part affinity field maps L. This is mathematically expressed as follows.

$L = (L_1, L_2, L_3 \ldots L_c)$

$L_c \in \mathbb{R}^{wxhx2}$

$c \in \{1 \ldots C\}$ where C is the total number of limbs

3 OUTPUT FORMAT

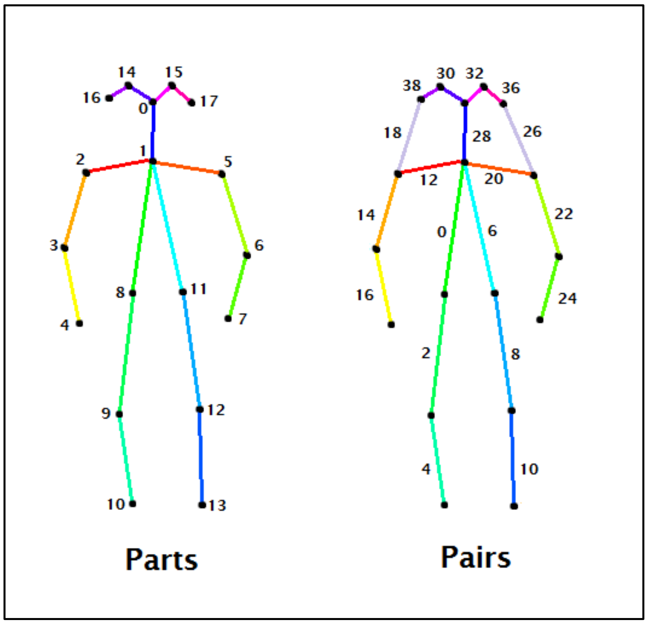


Fig. 2. The COCO output format

Body Part	Number	Body Part	Number
Nose	0	Right Knee	9
Neck	1	Right Ankle	10
Right Shoulder	2	Left Hip	11
Right Elbow	3	Left Knee	12
Right Wrist	4	Left Ankle	13
Left Shoulder	5	Right Eye	14
Left Elbow	6	Left Eye	15
Left Wrist	7	Right Ear	16
Right Hip	8	Left Ear	17

Table 1. Body Parts with dot mapping number

In the context of this paper, three positions have been classified using key points situated on the shoulder, the knees and wrists. These are namely:

A) Hunchback

In this type of position, the upper part of the spine is curved. Here, the neck muscles and the shoulder blades are curved inwards to create a rounded posture. The angle at which the spine hunches can be calculated using:

```
def checkPosition(all_peaks):
    try:
        f = 0
        if (all_peaks[]):
            a = all_peaks[Right Ear]
            f = 1
        else:
            a = all_peaks[Left Ear]
        b = all_peaks[Hip]
        angle = calcAngle(a,b)
        degrees = degrees(angle)
        if (f):
            degrees = 180 - degrees
        if (degrees<70):
            print("Spine: Hunchback")
```

```
Distance Function: disR(right wrist, right palm)-
disL(left wrist, left palm)
```

B) Recline

Here, the (body is lying face down with the upper trunk and head elevated, propped up by the arms, while the lower body is in contact with the supporting surface). This type of position can be expressed mathematically as:

```
def checkPosition(all_peaks):
    try:
        f = 0
        if (all_peaks[]):
            a = all_peaks[Right Ear]
            f = 1
        else:
            a = all_peaks[Left Ear]
        b = all_peaks[Hip]
        angle = calcAngle(a,b)
        degrees = degrees(angle)
        if (f):
            degrees = 180 - degrees
        if(degrees > 110):
            print("Spine: Reclined")
```

C) Straight

This is the recommended type posture, which puts the least amount of stress on the back muscles and spine. Here, the back should be perpendicular to the flat surface and the knees should be perpendicular to the floor. This can be expressed mathematically as:

```
def checkPosition(all_peaks):
    try:
        f = 0
        if (all_peaks[]):
            a = all_peaks[Right Ear]
            f = 1
        else:
            a = all_peaks[Left Ear]
        b = all_peaks[Hip]
        angle = calcAngle(a,b)
        degrees = degrees(angle)
        if (f):
            degrees = 180 - degrees
        if (degrees<70):
            print("Spine: Hunchback")
        elif (degrees > 110):
            print("Spine: Reclined")
        else:
            print("Spine: Straight"
```

In this paper, the authors have set a range of arbitrary values as the optimum angles required for a correct posture. These angles are set between 70 degrees and 110 degrees. A position whose angle is calculated to be less than 70 degrees is classified as hunchback and a position whose angle is calculated to be greater than 110 degrees is determined to be reclined. In addition to classifying the patient's posture, this paper also focuses on determining if the patient is kneeling or not and if their arms are folded or not. These two activities are also important in correcting the posture. As mentioned earlier, OpenPose also marks joints in the knees and wrists as keypoints. These points can then be used to if the knees or arms are folded or not using the mathematical expression given below:

4 LITERATURE SURVEY

Human posture and action detection and recognition has been extensively studied by various scientists and researchers. Various types of algorithms have been implemented and their results have been compared to determine the most optimal algorithm. Several authors like Adrian Bulat and Georgios Tzimiropoulos (Human pose estimation via Convolutional Part Heatmap Regression, 2016) and Li, Lan et al. (Online Human Action Detection using Joint Classification-Regression Recurrent Neural Networks, 2016) have combined convolutional neural networks with machine learning algorithms like joint classification- regression and Heatmap regression, and have obtained results with improved image quality of the action type, start and end types and pose types. Varol et al. (Long-term Temporal Convolutions for Action Recognition) explore the idea of using convolutional neural networks combined with long term temporal convolutions to improve the quality of video frames that capture minute human actions. Barros, Magg et al. (A Multichannel Convolutional Neural Network for Hand Posture Recognition, 2014) develops a multichannel CNN model to recognize hand gestures with the help of different databases. Algorithms to enhance 2-D and 3-D images have also been studied by Varadarajan et al. (A Greedy Part Assignment Algorithm for Realtime Multi-Person 2D Pose Estimation, 2018), who employs the Greedy Part Assignment algorithm in multi-person images for human pose-estimation to reduce the part-candidates, assigning part-classes to person-clusters and part-candidates within part-classes to person-clusters and selecting the most person clusters, while still preserving the integrity of the multi-person images. Cippitelli et al. (A Human Activity Recognition System Using Skeleton Data from RGBD Sensors, 2016) develop a human activity recognition algorithm to monitor older people in their homes using RGBD sensors, which reads skeleton data, while Gaglio et al. (Human Activity Recognition Process Using 3-D Posture Data, 2015) implement the same system using only RGBD cameras along with Kinect and machine learning algorithms like K-means clustering for posture estimation. Models to recognize the shape and actions of the human skeleton are a wide area of research. While Devanne et al. (3D Human Action Recognition by Shape Analysis of Motion Trajectories on Riemannian Manifold, 2014) implement an action recognition model using Riemannian geometry to create 3-D images of human joints and achieve nearly 90% accuracy in action detection, Althloothi et al. (Human activity recognition using multi-features and multiple kernel learning, 2013) fuses features like 3-D motion of joints and skeletal shape using Multiple Kernel Learning for activity recognition. Depth Motion Maps executed by Chen et al. (Real-time human action recognition based on depth motion maps, 2013) have found to be more efficient than traditional methods in detecting actions in video frames with an accuracy rate of 90.5%. Geometric features, along with machine learning algorithms, were also exploited by Shirbhate, Talele (Human Body Language Understanding for Action Detection using Geometric Features, 2016) to explore human body language to detect and classify different standing and sitting postures as emotions. Convolutional pose machines were developed by Wei et al. (Convolutional Pose Machines, 2016) to predict the actions of in a sequence of images from a specified location in each picture. Many algorithms have been proposed that combine human activity detection and human posture detection such as classifying with rejection by Tang, Sazonov (Highly Accurate Recognition of Human Postures and Activities Through Classification With Rejection, 2014), which has yielded near to perfect results and using Microsoft Kinect along with neural networks and anthropometry by Li et al. (Hybrid approach for human posture recognition using anthropometry and BP neural network based on Kinect V2, 2019). Kinect is also used by Jiang et al. (Informative joints based human action recognition using skeleton contexts, 2015) to produce skeletal representations, which are then used for activity recognition. Qiao et al. (Real-Time Human Gesture Grading Based on OpenPose, 2017) have used functions from the library OpenPose in Python and OpenCV for human action and posture detection to produce a real-time gesture grading system. Cao et al. (Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields, 2017) proposes an approach, which includes greedy parsing algorithms to detect human poses.

5 PROPOSED METHODOLOGY

This paper provides output by depending on the type of inputs the user provides. Input can be provided using pre-recorded images of the user or by recording the user's movements using the webcam. The model can be trained as per the patient's requirements like height and weight. The key points detected by the OpenPose library are color coded to locate the joints relevant to each type of activity. These points are then stored in a list and used later to classify the activities that the patient is doing. Here, these points are used to determine if the position of the spine, of the arms are folded or not and if the patient is kneeling or not.

To determine the posture of the patient, the angle between the keypoint is calculated with respect to the X axis in XY plane. As mentioned earlier, if the angle is found to be less than 70 degrees, the posture is classified as “Hunchback”. If the angle is found to be greater than 110 degrees, then the posture is called “Reclined”. Lastly, if the angle results in a value between 70 degrees and 110 degrees then the posture is determined to be “Straight” and this is the recommended posture to be followed.

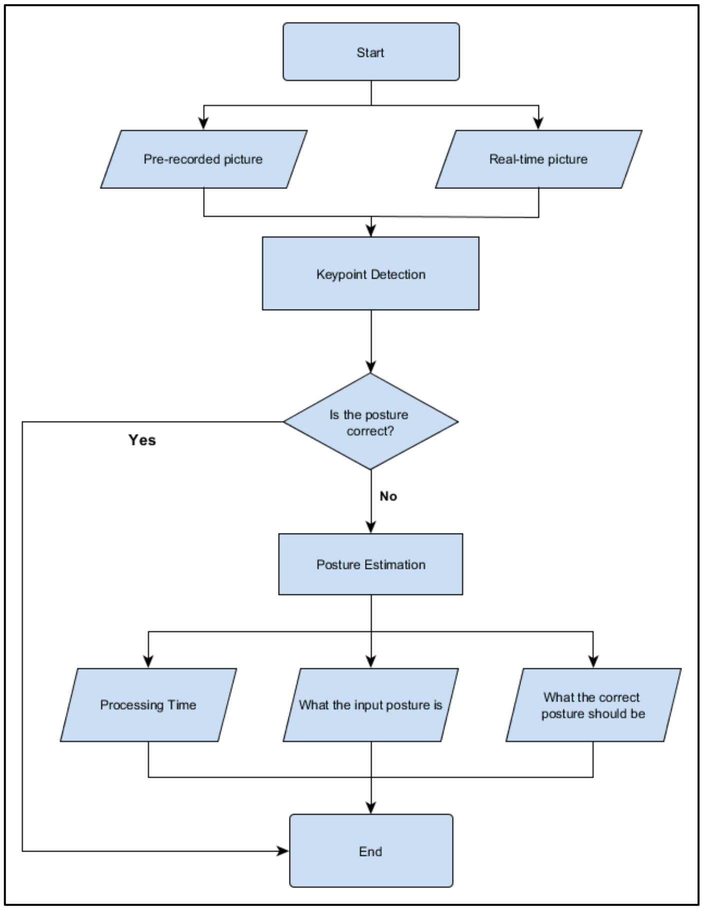


Fig. 3. Algorithmic Flowchart

In order to check the folding of the arms, a function is executed, which finds the distance between the right arm joint, the right palm, the left arm joint and the left palm. This function classifies if hands folded or not using an arbitrary value obtained as a result. The last feature of the paper is to check if the patient is kneeling or not. This is executed by locating the knee and the hip and the angle between them. This function detects one leg at a time and classifies each leg as “Kneeling” or “Not kneeling”. If the result of this calculation is greater than 60 degrees, then it can be determined that both the legs are in the kneeling position. An additional feature of the proposed method is that in case the patient’s posture is determined to be in the reclined or hunchback position, he or she is informed how far to move forward or backward to reach the correct posture. For example, if the patient’s posture is determined to be reclining at an angle of 120 degrees, the patient is told to correct their posture by moving backwards by 10-30 degrees. This can be expressed mathematically as:

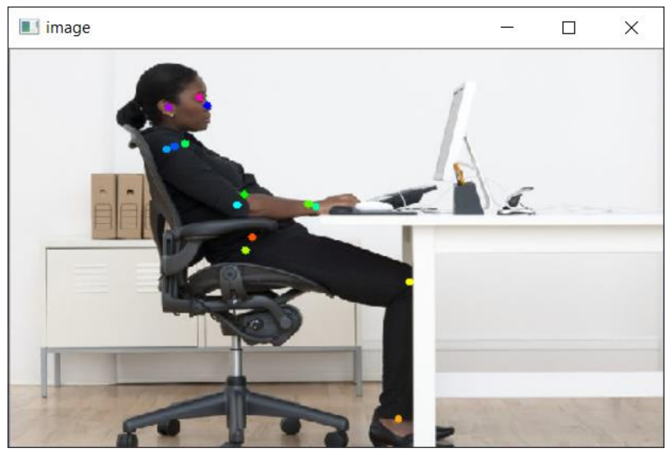
$$\text{degrees} = \text{degrees} - 90°$$

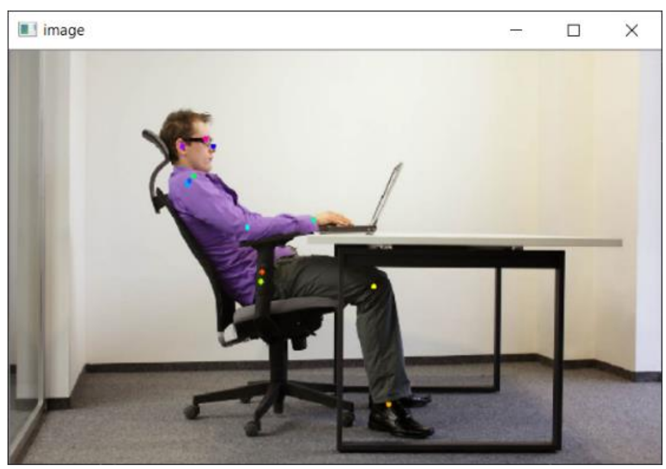
If the posture is determined to be hunchback, then the correct angle of posture can be found using the following mathematical equation:

$$\text{degrees} = 90° - \text{degrees}$$

The Keras model is loaded when the program is executed, which is coded in Python. This model has been trained with images of different positions. The final output displays the image being processed, scaled to the resolution of the screen. It shows the type of posture being detected and if the knees and hands are folded or not. It also gives the angle at which the spine and the legs are folded at, along with the time taken to run the project.

6 DATASET





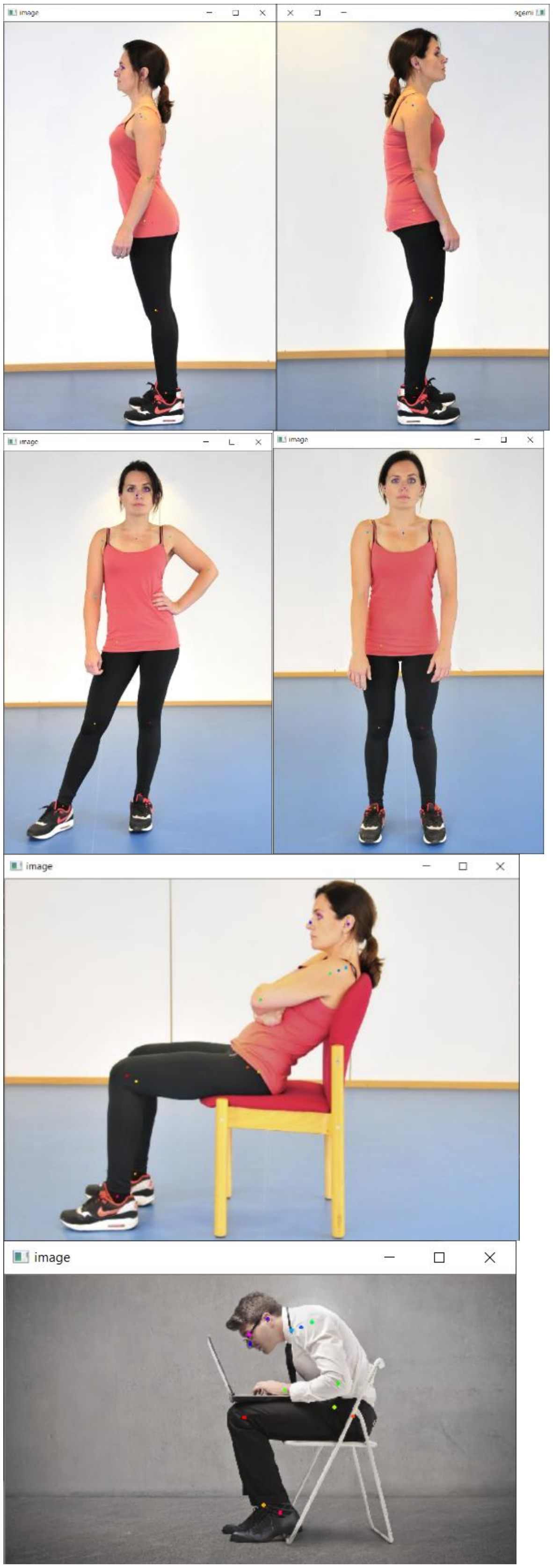


Fig. 4-11. The color coded dot mapping after running the sample images through our model. The legs are color coded to red and yellow the nose is color coded to blue. The shoulder, elbow and wrist are green and eyes are pink.

The above pictures were used as a sample set that the authors ran through the proposed model and the output were color coded dot mapped skeletal points. Then a dataset of 10 people of varying sizes was taken into account to do an α value study to recognize the best possible value for the arbitrary "α" used in the algorithm to tailor the model to the body shape and type of the person in question. We manually figured out the α value for these 10 subjects and then mapped those values to their BMIs to get a relationship between those two values. The range of height in the following dataset is 0.255m (1.803m - 1.548m), the range of weight in the following dataset is 48kg (96kg – 48kg) and the range of BMI is 9.6kg/m^2 (29.5kg/m^2 – 19.9kg/m^2).

Subject	Gender	Height (m)	Weight (kg)	BMI (kg/m^2)	α value
1	M	1.778	63	19.9	100
2	M	1.676	59	21	90
3	M	1.752	65	21.2	95
4	M	1.752	79	25.7	95
5	M	1.803	96	29.5	110
6	M	1.752	70	22.8	95
7	F	1.574	53	21.4	80
8	F	1.600	52	20.3	84
9	F	1.625	59	22.3	87
10	F	1.548	48	20	75

Table 2. Anthropometric variations to study the α value suitable for the patient

7 RESULTS

The following is the figure obtained by mapping the BMI of the 10 subjects to get a relation between BMI and α value.

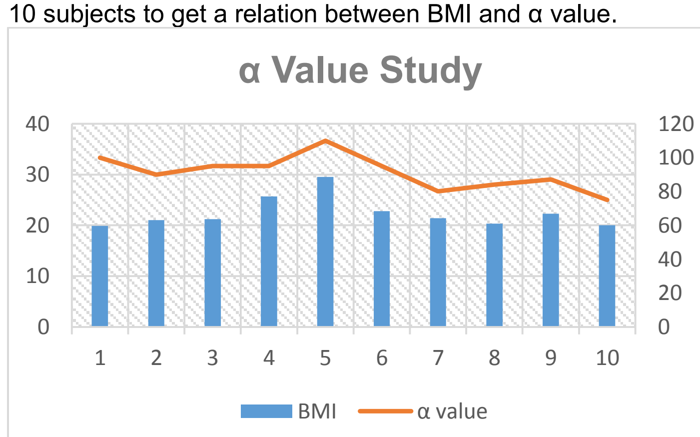


Fig. 12. α value mapped against the BMI

Spine	Angle	Correction
Hunchback	degrees<70°	90°-degrees
Reclined	degrees>110°	degrees-90°
Straight	70°<degrees<110°	No Correction
Exception	-	Unable to detect

Table 3 *Spine position according to the angle and angle of correction*

Legs	Angle	Classification
Left Leg	leftdegrees>60°	Left Leg Kneeling
Right Leg	rightdegrees>60°	Right Leg Kneeling
Both Legs	leftdegrees>60° and rightdegrees>60°	Both Legs Kneeling
Both Legs	leftdegrees<60° or rightdegrees<60°	Not Kneeling
Exception	-	Unable to detect

Table 4 Classification of legs as “Kneeling” or “Not Kneeling” according to angle.

Hands	Distance	Classification
Both Hands	disR<(disL+α) and disR>(disL-α)	Not Folding
Both Hands	Otherwise	Folding
Exception	-	Unable to detect

Table 5 Classification of hands as “Folding” or “Not Folding” according to the difference of distance between the right wrist and right palm and left wrist and left palm.

8 CONCLUSION

This topic is highly beneficial for the betterment of the people in rehab and there is immense scope for improvement in the techniques and the algorithms used as we discovered after the literature survey. Henceforth we will be working on to detect and analyse the human posture through various models. The algorithm was trained and tested using datasets with images of subjects exhibiting different types of postures. Using the images of the subject, the three different types of postures were found. A posture is concluded to be hunchback if the angle found between the key points is less than 70 degrees. To attain a straight posture, the alpha value to be corrected was calculated to be between 5 and 20 degrees. For postures that were found to be reclines, where the angle between key points was greater than 110 degrees, α was found to be the same.

ACKNOWLEDGEMENT

The authors would like to thank Dr. Sureshkumar N. (Associate Professor, Vellore Institute of Technology, Vellore) for his guidance and insight during the course of this research.

REFERENCES

[1] Althloothi, S., Mahoor, M. H., Zhang, X., & Voyles, R. M. (2014). Human activity recognition using multi-features and multiple kernel learning. Pattern Recognition, 47(5), 1800–1812.

[2] Barros, P., Magg, S., Weber, C., & Wermter, S. (2014). A Multichannel Convolutional Neural Network for Hand Posture Recognition. Artificial Neural Networks and Machine Learning – ICANN 2014 Lecture Notes in Computer Science, 403–410.

[3] Cao, Z., Simon, T., Wei, S.-E., & Sheikh, Y. (2017). Realtime Multi-person 2D Pose Estimation Using Part Affinity Fields. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

[4] Chen, C., Liu, K., & Kehtarnavaz, N. (2013). Real-time human action recognition based on depth motion maps. Journal of Real-Time Image Processing, 12(1), 155–163.

[5] Cippitelli, E., Gasparrini, S., Gambi, E., & Spinsante, S. (2016). A Human Activity Recognition System Using Skeleton Data from RGBD Sensors. Computational Intelligence and Neuroscience, 2016, 1–14.

[6] Devanne, M., Wannous, H., Berretti, S., Pala, P., Daoudi, M., & Bimbo, A. D. (2015). 3-D Human Action Recognition by Shape Analysis of Motion Trajectories on Riemannian Manifold. IEEE Transactions on Cybernetics, 45(7), 1340–1352.

[7] Gaglio, S., Re, G. L., & Morana, M. (2015). Human Activity Recognition Process Using 3-D Posture Data. IEEE Transactions on Human-Machine Systems, 45(5), 586–597.

[8] Jiang, M., Kong, J., Bebis, G., & Huo, H. (2015). Informative joints based human action recognition using skeleton contexts. Signal Processing: Image Communication, 33, 29–40.

[9] Li, B., Han, C., & Bai, B. (2019). Hybrid approach for human posture recognition using anthropometry and BP neural network based on Kinect V2. EURASIP Journal on Image and Video Processing, 2019(1).

[10] Qiao, S., Wang, Y., & Li, J. (2017). Real-time human gesture grading based on OpenPose. 2017 10th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI).

[11] Shirbhate, N., & Talele, K. (2016). Human body language understanding for action detection using geometric features. 2016 2nd International Conference on Contemporary Computing and Informatics (IC3I).

[12] Tang, W., & Sazonov, E. S. (2012). Highly accurate classification of postures and activities by a shoe-based monitor through classification with rejection. 2012 Annual International Conference of the IEEE Engineering in Medicine and Biology Society.

[13] Varol, G., Laptev, I., & Schmid, C. (2018). Long-Term Temporal Convolutions for Action Recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence, 40(6), 1510–1517.

[14] Wei, S.-E., Ramakrishna, V., Kanade, T., & Sheikh, Y. (2016). Convolutional Pose Machines. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR).