Journal of KIIT. Vol. 20, No. 1, pp. 199-208 Jan. 31, 2022. pISSN 1598-8619, eISSN 2093-7571
http://dx.doi.org/10.14801/jkiit.2022.20.1.199

Real-Time Workout Posture Correction using OpenCV and MediaPipe

Yejin Kwon*, Dongho Kim**

This research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education (No. S-2021-A0496-00167)

Abstract

This paper proposes a program that uses OpenCV and MediaPipe to guide the workout (squat, push-up) posture correction content and the number of workouts by estimating the posture of real-time images. Specifically, the program estimates the landmarks of the user's body from the images acquired using the webcam in real time. Furthermore, it calculates the body angle and numerical values necessary to determine whether the posture is correct, and guides the user with the necessary corrections when the wrong posture is taken. It also increases and guides the number of workouts when the user takes the correct posture. Guidance for posture correction is displayed on the screen, and the number of workouts is guided by screen and voice. This program helps people exercise at home while checking their posture and work out with the proper posture without the help of an exercise trainer.

요 약

본 논문에서는 OpenCV와 MediaPipe를 활용하여 실시간 영상의 운동 자세(스쿼트, 팔굽혀펴기)를 추정해 운동 자세교정 내용과 운동횟수를 안내하는 프로그램을 제안한다. 구체적으로 프로그램은 실시간으로 웹캠을 이용하여 획득한 영상에서 사용자 신체의 landmark를 추정하고, 올바른 자세 여부 판단에 필요한 몸의 각도, 수치를 계산하여 사용자가 잘못된 자세를 취했을 때 교정에 필요한 내용을 안내할 수 있다. 또한, 사용자가 올바른 자세를 취했을 때 운동횟수를 증가시키고 안내한다. 자세교정을 위한 안내 문구가 화면에 표시되고 운동횟수가 화면과 음성으로 안내된다. 이 프로그램은 집에서 운동 트레이너의 도움 없이 자신의 자세를 확인하면서 올바른 자세로 운동할 수 있도록 돕는다.

Keywords

self-training, workout posture correction, openCV, MediaPipe

* Department of Computer Science and Engineering, Dongguk University, Seoul
- ORCID: https://orcid.org/0000-0002-7035-4918
** Dongguk Institute of Convergence Education, Dongguk University, Seoul
- ORCID: https://orcid.org/0000-0003-3349-103X

· Received: Sep. 15, 2021, Revised: Dec. 21, 2021 Accepted: Dec. 24, 2021
· Corresponding Author: Dongho Kim
Dongguk Institute of Convergence Education, Dongguk University
Tel.: +82-2-2290-1405, Email dongho.kim@dgu.edu

Ⅰ. Introduction

Many people affected by confinements related to the COVID-19 are interested in digital home exercise [1]. People familiar with intelligent devices enjoy home training through YouTube home training content [2] and home fitness applications [3]. However, suppose people refer to incorrect information created by non-professionals. In that case, they may exercise with the wrong posture, and there is a risk of injury by exercising without considering their physical strength. While home training has the advantage of convenience, it can be harmful to the body in that it is not possible to check whether one's exercise posture is correct.

Therefore, in this paper, we propose a program that can guide users to exercise with correct posture and the number of workouts while exercising without the involvement of an exercise trainer. Body posture estimation is made through the x, y, z coordinates obtained using the landmark model of MediaPipe Pose, an ML solution. If the user's posture does not meet the correct exercise posture criteria, the corresponding guidelines will be displayed on the screen. If the user's posture is proper, the guideline is not shown, the increased number of workouts is displayed on the screen, and the number of workouts is output by voice.

The composition of this paper is as follows. We examine existing studies related to posture recognition in Section 2. The theoretical background of the study is discussed in Section 3. The posture correction system is suggested in Section 4 along with correct workout posture criteria, body landmarks used for posture estimation, angle calculation, and camera calibration methods. The results of the experiment are in Section 5, and complete the study in Section 6.

Ⅱ. Related works

Studies for posture recognition are attracting attention. Related research includes sensor-based posture recognition, posture estimation through deep learning, and research using wearable devices.

Specifically, examples of research are as follows. Using an Arduino-based postural pressure measuring device to recognize an incorrect posture through an image, then recommend an exercise for posture correction [4]. Identify a wrong posture using a CNN-based deep learning model and human skeleton model, and provide feedback to correct yoga postures [5]. Provide feedback for correction through vibrations in clothing controlled by an Arduino processor, and suggest a smartphone app that can check alarm signals through Bluetooth [6]. Wirelessly recognize poses by applying a recognition algorithm on human motion data collected by wearable technology [7].

In addition to the healthcare area, such as systems that recognize falls in the elderly and provide monitoring contents [8], many areas utilize posture recognition. In games, people can interact with virtual games by recognizing hand gestures [9], and in the field of security, the government can detect violent behavior in CCTV and prevent crime [10].

However, existing deep learning-based studies take a long time for posture recognition, estimation and feedback because of learning through a large amount of data for posture classification and determination. Also, studies based on wearable devices or sensor-based data collection devices require users to have different equipment.

Therefore, in this study, we suggest a posture correction program that can reduce the time required to analyze posture and not require additional costs, such as equipment for analysis. It provides posture correction guidelines only for the most popular exercises, squat, and push-up through the screen. The number of workouts is provided by voice and text while executing the exercise in real-time to help the user exercise with the correct posture.

III. Background

3.1 Squat

The squat is a representative weight training exercise in which the lower quadriceps, lower quadriceps, femur triceps, femur muscles, and mesenchymal muscles are developed by repeatedly flexing and stretching the knee joint. Squat is effective in developing the lower extremities, but it can cause injury if performed without a proper selection of posture or weight.

The most common squats include wide squats with 120% wider stances, parallel squats with both feet together, and split squats with both feet spread back and forth in a lunge position.

3.2 Push-up

Push-up is one of the representative muscle exercises, which can be done without the use of equipment. While lying face down, the body is lifted by the force of stretching both arms by focusing the entire body's weight on the four spots of the hands and toes. The primary method is to bend the motion and elbow joints so that the body is not attached to the ground, and to repeat.

The most common types of push-ups are classical push-ups, grip-operated Hindu push-ups, grip-operated diamond push-ups, decline push-ups, incline push-ups, and clapping push-ups.

3.3 MediaPipe Pose

MediaPipe Pose is an ML solution for body pose tracking that uses BlazePose research to infer 33 human body key points from RGB video frames. BlazePose is a lightweight convolutional neural network architecture.

Pose tracking uses a two-stage detector-tracker ML pipeline. First, the pose detector finds a pose ROI (region of interest) within the frame. Next, the pose tracker predicts 33 pose key points from ROI [11]. Applications can be built by applying it to various fields that require posture analysis, such as yoga or fitness.

3.4 OpenCV

OpenCV is a programming library aimed at real-time computer vision, focusing on real-time image processing. It supports a variety of programming languages, including C++, Python, and Java, and a variety of OS platforms, including Windows, Linux, OS X, Android, and iOS. It can be used for many algorithms, including machine learning, math operations, video capture, and image processing.

OpenCV is algorithmically designed to focus on computational efficiency and real-time applications, making it easy to create high-quality commercial programs. It also supports multicore processing, which makes it applicable to a variety of situations.

IV. Proposed Approach

4.1 Exercise Posture Correction System

Fig. 1 shows a flow chart of posture analysis process. Exercise posture recognition system converts input images generated using a computer webcam from BGR images to RGB images using OpenCV. Using the Landmark model of MediaPipe Pose detects body landmarks required for postural analysis and saves the position. Calculate the angle of the posture using x, y, z coordinates of the analyzed position and verify that it meets the criteria for the correct exercise posture. The guideline suggests a correction is refreshed and drawn on the screen every second while the user's posture does not meet the criteria.

If the posture is correct, the correction guideline is not printed on the screen. Also, the number of workouts is increased and output by screen and voice.

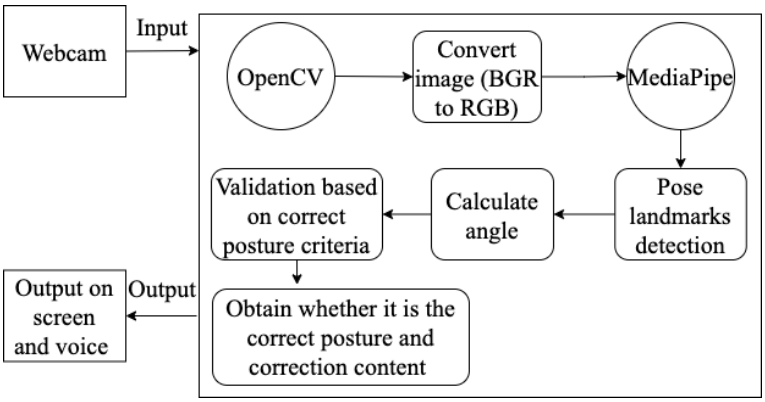

Fig. 1. Posture analysis flowchart

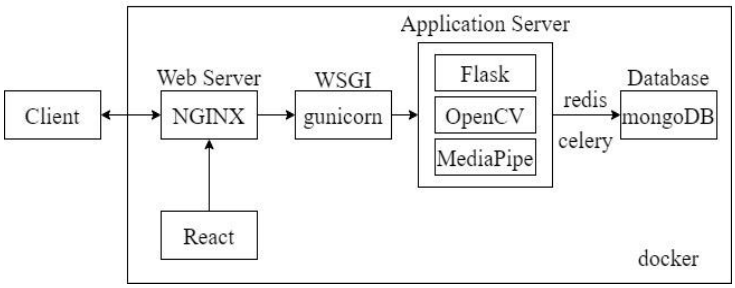

Fig. 2. System architecture

When a user executes a program, the client program sends a real-time exercise video frame from the webcam. The server finds the landmark in the image, verifies that it meets the criteria for correct posture, and, if not, forwards the correction contents back to the client. If the user is in the proper exercise posture, only the number of workouts is forwarded. The user can check the contents on the screen whenever the value is transferred from the server, and when the correct posture is taken, the number of workouts is output by voice. The guide for posture correction provides how to change which body part to be in the right posture.

The development tools used to implement the exercise posture correction program are Docker, React (v.17.0.2 version), flask (v.1.1.1), gunicorn (v.20.4), Nginx, MongoDB, Visual Code, OpenCV (v.4.5.3) and MediaPipe.

React has the performance benefit of rapid rendering using Virtual DOM. Flask is a microframework based light web framework and has the performance benefit of generating fewer generic errors. Nginx is easy to obtain server performance and reliability. OpenCV and MediaPipe are used to estimate posture.

Fig. 2 shows the system architecture of the program. When a HTTP request is sent from the client framework React to the web server nginx, the connection between the nginx and the server application flask is relayed by gunicorn.

In flask, OpenCV and MediaPipe are used to analyze image frames delivered on request from clients and store user records in MongoDB, where DB-related tasks are processed as asynchronous tasks using Redis and Celery so that the processing time can be performed quickly.

For this program, we chose squat and push-up, which are the most popular exercises that we can exercise without tools.

4.2 Definition of Correct Posture

There are three proper posture conditions for squats. The first condition is that the hip angle would be

close to 90 degrees, and the second condition is that the thigh is almost horizontal to the floor. And the third condition is that the knee and toe lines would be close to a straight line. Therefore, the program determines that the squats were performed correctly only when all the following conditions were met.

Condition 1) The hip angle (knee-hip-shoulder) is between 60 and 120 degrees.

Condition 2) Difference of y-values between knees and thighs is within 0.2.

Condition 3) The difference between the x-values of the knee and toe line is not more than 0.1.

Fig. 3 shows the algorithm for generating guidance for squat posture correction. In the case of Condition 1, it is guided to make the height of the hip higher when the angle of the hip is less than 60 degrees, and to make the height lower when it is greater than 120 degrees.

```python
#Guidance function for correcting squat posture
def is_squat(hip_angle,l_knee_hip,r_knee_hip,l_knee_foot,r_knee_foot):
    squat_guide = ""
    action_status = True

    if 60 < hip_angle < 120 and l_knee_hip<=0.2 and r_knee_hip<=0.2 and l_knee_foot<=0.1 and r_
        action_status = False

    else:
        if action_status == False :
            action_status = True
        #Condition 1
        if hip_angle<60 :
            squat_guide +='엉덩이 높이를 올려주세요\n'
        if hip_angle>120:
            squat_guide += '엉덩이 높이를 낮춰주세요\n'
        #Condition 2
        if l_knee_hip>0.2 or r_knee_hip>0.2:
            squat_guide += '허벅지가 바닥과 수평이 되도록 해주세요\n'
        #Condition 3
        if l_knee_foot > 0.1 or r_knee_foot > 0.1:
            squat_guide += '무릎이 발끝선을 넘지 않게 해주세요\n'

    if not action_status:
        text = "성공"
    else:
        text = "실패"
    return action_status, squat_guide, text

#Element calculation function required for the squat posture standard
def do_squat(shoulder_l, shoulder_r, hip_l, hip_r, knee_l, knee_r, foot_l, foot_r):
    #Condition 1
    l_hip_angle = get_angle(knee_l, hip_l, shoulder_l)
    r_hip_angle = get_angle(knee_r, hip_r, shoulder_r)
    hip_angle = (l_hip_angle + r_hip_angle) / 2

    #Condition 2
    def squat_1(p1,p2):
        return round(abs(p1.y-p2.y),3)
    l_knee_hip = squat_1(knee_l,hip_l)
    r_knee_hip = squat_1(knee_r,hip_r)

    #Condition 3
    def squat_2(p1,p2):
        return round(abs(p2.x-p1.x),3)
    l_knee_foot = squat_2(knee_l,foot_l)
    r_knee_foot = squat_2(knee_r,foot_r)

    count, guide, text = is_squat(hip_angle,l_knee_hip,r_knee_hip,l_knee_foot,r_knee_foot)
```

Fig. 3. Algorithm to generate squat posture correction contents

In the case of Condition 2, when the difference between the y values of the knee and the thigh is more than 0.2, it is guided to keep the thigh horizontal to the floor. In the case of Condition 3, when the difference between the x values of the knee and the tip of the toe are more than 0.1, the knee is guided not to exceed the tip of the toe.

There are two correct posture conditions for push-ups. The first condition is that the elbow angle will be close to 90 degrees during the push-up, and the second condition is to keep the body close to a straight line during the push-up. Therefore, the program determines that the push-up was performed correctly only when all the following conditions were met.

Condition 1) Elbow angle (wrist-elbow-shoulder) between 70 and 100 degrees.

Condition 2) Body angle (shoulder-hip-ankle) is between 160 and 200 degrees.

```python
#Guidance function for correcting pushup posture
def is_pushup(elbow_angle, body_angle):
    pushup_guide= ""
    action_status = True

    if 70 <= elbow_angle <= 100 and 160<= body_angle <= 200:
        action_status = False

    else:
        if action_status == False:
            action_status = True

        # Condition 1
        if elbow_angle < 70:
            pushup_guide +='너무 내려가셨어요! 팔꿈치 각도를 90도로 맞춰주세요\n'
        if elbow_angle > 100:
            pushup_guide += '조금 더 내려기주세요! 팔꿈치 각도를 90도로 맞춰주세요\n'

        # Condition 2
        if body_angle < 160:
            pushup_guide += '엉덩이를 내려서 몸이 일직선이 되도록 맞춰주세요\n'
        if body_angle > 200:
            pushup_guide += '엉덩이를 올려서 몸이 일직선이 되도록 맞춰주세요\n'

    if not action_status:
        text = "성공"
    else:
        text = "실패"
    return action_status, pushup_guide, text

#Element calculation function required for the pushup posture standard
def do_pushup(l_sh, r_sh, l_elbow, r_elbow, l_wrist, r_wrist, l_hip, r_hip, l_ankle, r_ank
    # Condition 1
    l_elbow_angle = get_angle(l_wrist, l_elbow, l_sh)
    r_elbow_angle = get_angle(r_wrist, r_elbow, r_sh)
    elbow_angle = (l_elbow_angle + r_elbow_angle) / 2

    # Condition 2
    l_body_angle = get_angle(l_sh, l_hip, l_ankle)
    r_body_angle = get_angle(r_sh, r_hip, r_ankle)
    body_angle = (l_body_angle + r_body_angle) / 2

    p_count, p_guide, text = is_pushup(elbow_angle, body_angle)
```

Fig. 4. Algorithm to generate push-up posture correction contents

Fig. 4 shows the algorithm for generating guidance for push-up posture correction. In the case of Condition 1, it is guided that the body has fallen too much when the angle of the elbow is less than 70 degrees, and that the body has to go down a little more when it is greater than 100 degrees. In the case of Condition 2, when the angle of the body is less than 160, it is guided to lower the hip to align the body, and when it is greater than 200 degrees, it is guided to raise the hip to align the body.

The posture correction algorithm analyzes the user's image to estimate the pose landmark, calculates it through a function that calculates the angle and value required for the exercise posture condition, and then generates the posture correction guide content through the correction guide content generation function.

4.3 Pose Landmark

Use the key points of the MediaPipe landmark model to determine whether the squat meets the correct posture conditions. Since the angles of the left and right hips can be different, the average of the two values is used as the hip angle as (1). Fig. 5 shows the key points used in the conditions for validating correct squat posture.

Condition 1) Left hip angle (left knee(25), left hip(23), left shoulder(11)), Right hip angle (right knee(26), right hip(24), right shoulder(12))

$$Hip\ angle = (Left\ hip\ angle + Right\ hip\ angle)/2 \quad (1)$$

Condition 2) Left knee-hip height difference (left knee(25), left hip(23)), Right knee-hip height difference (right knee(26), right hip(24))

Condition 3) Left foot-knee width difference (left knee(25), left foot(31)), Right foot-knee width difference (right knee(26), right foot(32))

Use the key points of the MediaPipe landmark model to determine whether the push-up meets the correct posture conditions. Since the angles of the left and right elbows can be different, the average of the two values is used as the elbow angle as (2). Body angle uses the average value as in (3) for the same reason. Fig. 6 shows the key points used in the conditions for validating correct push-up posture.

Condition 1) Left elbow angle (left wrist(15), left elbow(13), left shoulder(11)), Right elbow angle (right wrist(16), right elbow(14), right shoulder(12))

$$Elbow\ angle = (Left\ elbow\ angle + Right\ elbow\ angle)/2 \quad (2)$$

Condition 2) Left body angle (left ankle(27), left hip(23), left shoulder(11)), Right body angle (right ankle(28), right hip(24), right shoulder(12))

$$Body\ angle = (Left\ body\ angle + Right\ body\ angle)/2 \quad (3)$$

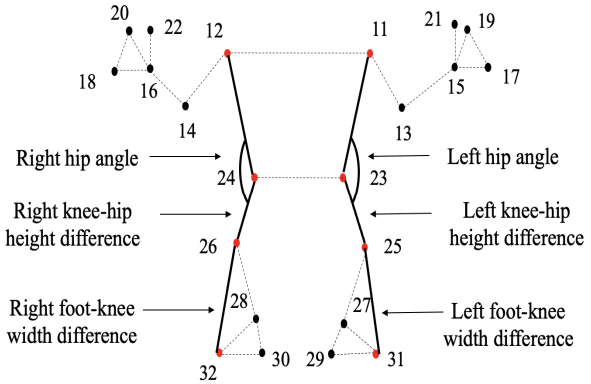

Fig. 5. Landmarks for validating correct squat posture

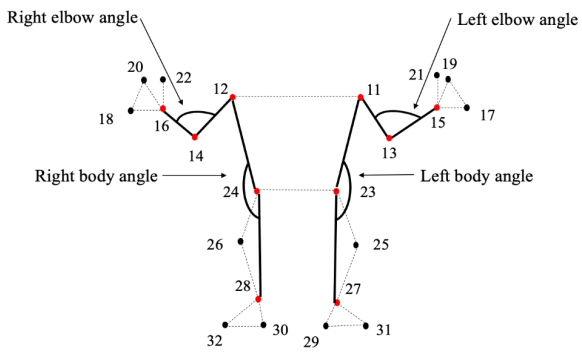

Fig. 6. Landmarks for validating correct push-up posture

4.4 Camera Calibration

When an image is acquired through a webcam, the position, direction, and angle of the camera are different for each user, thus obtaining an image of a modified object that is different from the object of the real world. Therefore, images with calibration for the camera should be used. For correction to the camera, a method is used to obtain a modified rotation angle using a fixed indicator of the exercise posture and then rotate the image to the negative value of the rotation angle.

Since squat is a standing exercise, the angles of both feet should be kept horizontal. Since push up is a lie-down exercise, the angles of both feet should be kept vertical. The angle of the foot uses a straight line between the heel and the tip of the toe. Since the angles of both feet can be different, the rotation angle uses the average value of the rotation angle of each foot as (4).

Fig. 7 shows the key points used to calculate the angle of the foot. Key points used for calculating left foot rotation angle are left heel(29) and left foot(31). Key points used for calculating right foot rotation angle are right heel(30) and right foot(32).

$$Rotation\ angle = \frac{(Left\ rotation\ angle + Right\ rotation\ angle)}{2} \quad (4)$$

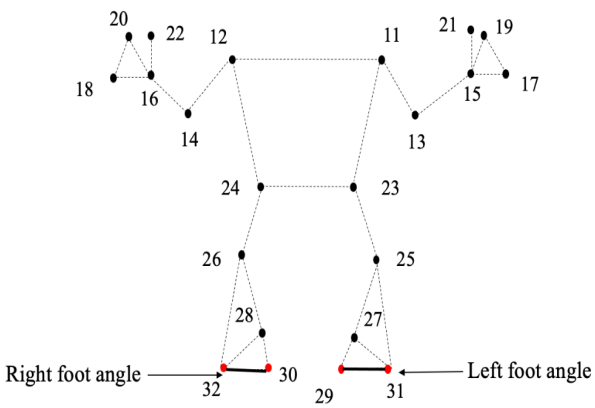

Fig. 7. Landmarks used to find the angle of the foot

After obtaining the rotation angle, calculate the rotation conversion matrix. The rotation transformation matrix is defined as (5). And α and β values can be defined as (6).

$$M = \begin{bmatrix} \alpha & \beta & (1-\alpha) \times Center_x - \beta \times Center_y \\ -\beta & \alpha & \beta \times Center_x - (1-\alpha) \times Center_y \end{bmatrix} \quad (5)$$

$$\alpha = scale \times \cos(\theta) \quad (6)$$
$$\beta = scale \times \sin(\theta)$$

Center is the coordinate of the center point, scale is the ratio, and θ is the rotation angle. The horizontal and vertical lengths of the image can be obtained using OpenCV, and the rotation transformation matrix is calculated with the negative value of the rotation angle obtained before, the rotation center, and the scale as 1. This is then applied to the warpAffine function to obtain the transformed image.

4.5 Calculate Angle

To estimate the pose, x, y, and z coordinates are used to calculate the angle between the three points. Fig. 8 shows the angle between the three points. Let the three points be A, B, and C. And the angle to be obtained is θ.

To obtain θ, we use the dot product of vectors.

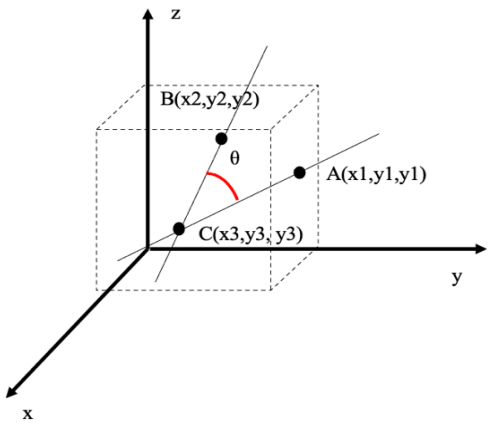

Fig. 8. Angle between three points

Vector with point C as the point of view and A as the endpoint is called $\overrightarrow{CA}$ and vector with point C as the point of view and B as the endpoint is called $\overrightarrow{CB}$. Vector $\overrightarrow{CA}$ and $\overrightarrow{CB}$ are defined as (7).

$$\overrightarrow{CA} = (x1 - x3,\ y1 - y3,\ z1 - z3) \quad (7)$$
$$\overrightarrow{CB} = (x2 - x3,\ y2 - y3,\ z2 - z3)$$

So, θ is obtained by substituting it for the internal formula of the vector. In this way, the angle between the three points can be defined as (8).

$$\overrightarrow{CA} \cdot \overrightarrow{CB} = |\overrightarrow{CA}||\overrightarrow{CB}|\cos(\theta) \quad (8)$$
$$\theta = \cos^{-1}\frac{\overrightarrow{CA} \cdot \overrightarrow{CB}}{|\overrightarrow{CA}||\overrightarrow{CB}|}$$

V. Experimental Results

To verify that the implemented system is executed as designed, we identified the results of taking the correct and incorrect posture, respectively.

5.1 Squat

Fig. 9 shows the output from the program when the correct squat posture is taken. It was found that the hip angle between 60 and 120 degrees, the thighs positioned close to the floor, and the knees not crossing the toe line are required to avoid guidelines. And when the condition was satisfied, the number of workouts was increased on the screen. It was confirmed that the number 8 was output by voice.

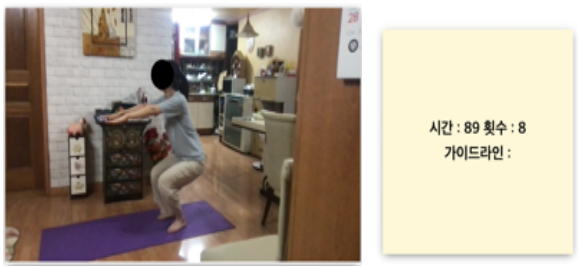

Fig. 9. Correct squat posture

Conversely, Fig. 10 shows the output from the program when the incorrect squat posture is taken. Because the user's posture did not meet the condition to keep the thigh horizontal, a guideline was printed to make sure that the thigh horizontal with the floor. And the number of exercises did not increase.

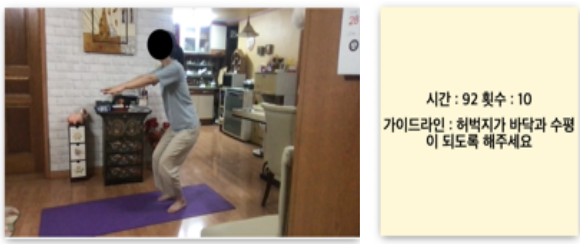

Fig. 10. Incorrect squat posture

5.2 Push-up

Fig. 11 shows the correct user's posture when executing push-ups. When performing push-ups, the elbow angle between 70 and 100 degrees and the body angle close to 180 degrees was required to prevent the guidelines and to increase the number of workouts. And it was confirmed that the number 3 was output by voice. No correction is printed on the screen, and an increased number of workouts is printed. In addition, the increased number of workouts is output by voice.

On the other hand, Fig. 12 shows the output from the program when the incorrect push-up posture is taken. Because the user's posture did not meet the condition that elbow angle was below 100 degrees, a guideline was printed to bend the arm more to approximately 90 degrees. And the number of workouts did not increase.

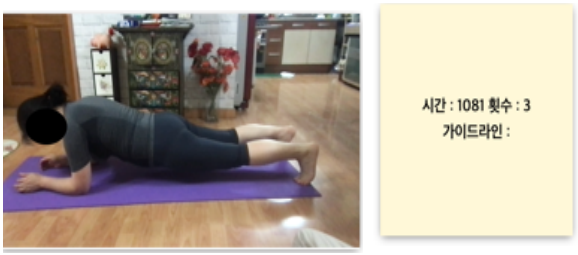

Fig. 11. Correct push-up posture

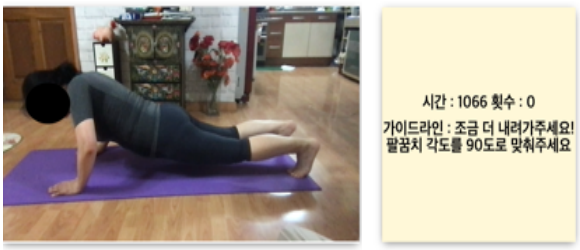

Fig. 12. Incorrect push-up posture

And we conducted an experiment to analyze the performance of the program. The detail of the experiment is to calculate the success rate whether the program accurately calculates the number of workouts when the user performs exercise 1 to 5 times in the correct posture. The formula to calculate the success rate is defined as (9). And Table 1 shows the experimental results.

$$Success\ rate = \frac{Calculated\ number\ of\ workouts}{Actual\ number\ of\ workouts} \times 100 \quad (9)$$

Table 1. Calculation success rate

Number of workouts	Squat success rate	Pushup success rate
1	100%	100%
2	100%	100%
3	100%	67%
4	100%	75%
5	100%	80%

VI. Conclusions

The program proposed in this study converts images of users obtained through webcams by OpenCV and uses the Landmark model of MediaPipe Pose to estimate body landmarks. And analyze posture in real-time based on correct exercise posture criteria. If it is not in the correct posture, the instructions for correction are displayed on the screen. The number of workouts is counted only when the user performs the exercise with the correct posture to encourage the user to exercise with the correct posture. Through this program, people will be able to exercise with correct posture at home without the help of a professional trainer so that people may obtain the cost-effective benefit of exercise. In addition, there is an advantage that the time required for posture analysis is short, so the user can receive immediate feedback.

In this study, camera calibration was only proposed theoretically and was not applied in the actual experimental process. In the next study, more accurate posture correction will be made by adding a camera calibration process to an existing program before estimating body landmarks.

References

[1] J. Wilke et al., "Restrictercise! Preferences Regarding Digital Home Training Programs during Confinements Associated with the COVID-19 Pandemic", International Journal of Environmental Research and Public Health, Vol. 17, No. 18, Sep. 2020. https://dx.doi.org/10.3390%2Fijerph17186515.

[2] S. H. Lee and J. H. Kwak, "Analysis of YouTube Channel Related to Home Training through Un Contact Generation", The Korea Journal of Sports Science, Vol. 29, No. 6, pp. 181-191, Dec. 2020. https://dx.doi.org/10.35159/kjss.2020.12.29.6.181.

[3] I. Khaghani-Far, S. Nikitina, M. Báez, E. A. Taran, and F. Casati, "Fitness Applications for Home-Based Training", IEEE Pervasive Computing, Vol. 15, No. 4, pp. 56-65, Oct.-Dec. 2016. https://dx.doi.org/10.1109/MPRV.2016.76.

[4] H. J. Ha and C. D. Lee, "Design of Algorithm for Guidance of Sitting Posture Correction Using Pressure Sensor and Image Processing Interpolation Technique", Journal of KIIT, Vol. 14, No. 1, pp. 37-44, Jan. 2016. https://dx.doi.org/10.14801/jkiit.2016.14.1.37.

[5] A. Chaudhari, O. Dalvi, O. Ramade, and D. Ambawade, "Yog-Guru: Real-Time Yoga Pose Correction System Using Deep Learning Methods", International Conference on Communication information and Computing Technology (ICCICT),

Mumbai, India, pp. 1-6, Jun. 2021. https://dx.doi.org/10.1109/ICCICT50803.2021.9509937.

[6] Q. Wang, W. Chen, A. A. A. Timmermans, C. Karachristos, J. B. Martens, and P. Markopoulos, "Smart Rehabilitation Garment for posture monitoring", 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), Milan, Italy, pp. 5736-5739, Aug. 2015. https://doi.org/10.1109/embc.2015.7319695.

[7] L. Gao, G. Zhang, B. Yu, Z. Qiao, and J. Wang, "Wearable human motion posture capture and medical health monitoring based on wireless sensor networks", Measurement, Vol. 166. pp. 108252, Dec. 2020. https://doi.org/10.1016/j.measurement.2020.108252.

[8] G. Diraco, A. Leone, and P. Siciliano, "An active vision system for fall detection and posture recognition in elderly healthcare", 2010 Design, Automation & Test in Europe Conference & Exhibition (DATE 2010), Dresden, Germany, pp. 1536-1541, Mar. 2010. https://doi.org/10.1109/DATE.2010.5457055.

[9] S. S. Rautaray and A. Agrawal, "Interaction with virtual game through hand gesture recognition", 2011 International Conference on Multimedia, Signal Processing and Communication Technologies, Aligarh, India, pp. 244-247, Dec. 2011. https://dx.doi.org/10.1109/MSPCT.2011.6150485.

[10] G. Morales, I. Salazar-Reque, J. Telles, and D. Díaz, "Detecting Violent Robberies in CCTV Videos Using Deep Learning", Artificial Intelligence Applications and Innovations, Vol. 559, pp 282-291, May 2019. https://doi.org/10.1007/978-3-030-19823-7_23.

[11] V. Bazarevsky, I. Grishchenko, K. Raveendran, T. Zhu, F. Zhang, and M. Grundmann, "BlazePose: On-device Real-time Body Pose tracking", CVPR Workshop on Computer Vision for Augmented and Virtual Reality, Seattle, WA, USA, Jun. 2020.

Authors

Yejin Kwon

2019 ~ present : B.S. degree in Computer Science and Engineering, Dongguk University

Research interests : mobile software, image processing, computer vision, internet of things(IoT)

Dongho Kim

1990 : B.S. degree in Computer Engineering, Seoul National University

1992 : M.S.,

2002 : Ph.D. degrees in Computer Science, University of Southern California

2001 ~ 2006 : Research Scientist, Information Sciences Institute – University of Southern California

2006 ~ 2012 : CEO, Iroonet America

2014 : Visiting Scholar, National University of Singapore

2014 ~ 2015 : CEO, SecuEZ Labs.

2016 ~ present : Professor at Dongguk Institute of Convergence Education, Dongguk University

Research interests : Artificial Intelligence, Networks, Security