GYM-BUDDIES: GRADUATE CERTIFICATE INTELLIGENT SENSING SYSTEMS PRACTICE MODULE REPORT

Chun How Oh, Mikhail Kennerley, Vidish Mehta

Institute of Systems Science, National University of Singapore, Singapore 119615

ABSTRACT

Incorrect form is one of the leading factors of injury when performing exercises. This project aims to help users identify mistakes via a real-time or with a pre-recorded video when performing squats or push-ups. This is done via visual feedback made possible by a combination of methods from pose-estimation, 3D-CNNs, statistical approaches and a rules engine with an input of a pair of synchronised monochrome and disparity videos. The visual feedback provides information on the exercise being performed, the number of completed repetitions, a highlight of the users pose where mistakes are made and the number of failed repetitions. The interface is available via both command-line as well as a web-interface.

Index Terms— Pose-estimation, Exercise Form, Exercise Counter, Exercise Rules

1. INTRODUCTION

Correct form is key when performing various exercises. Incorrect form can lead to the exercise not targeting correct muscles, not being as efficient or even leading to both minor and major injuries. In a survey of 30 sports professionals taken during Pan American Games of Guadalajara in 2011, incorrect technique was answered as the top cause of injuries in athletes participating in their sport [1]. Incorrect form can arise from fatigue, overloading, not knowing the correct technique for the exercise or injuries.

This project aims to solve the problem of incorrect form by providing users with feedback on a recorded video of them performing a set of exercises, which are currently limited to squats and push-ups. The feedback includes identifying the exercise being performed, the count of the number of repetitions of the exercise, identify common mistakes of said exercises and the final count which is calculated by the total count less the number of repetitions where a mistake was identified. The mistakes of the user are highlighted in the video for easy reference and the processing of the video was created to be real-time allowing for the potential of live feedback to the user. This was done using a combination of methods, such as pose-estimation using BlazePose, 3D Convolutiounal Neural Networks, statistical approaches and rules.

The inputs to the system is a synchronised pair of monochrome and disparity videos at 1280*800 resolution of a user performing an exercise. Requirements of the video include the user facing the left of the camera either perpendicular or at a 45 degree angle and waiting 2 seconds before starting the exercise for the system to initialise and create a baseline.

The system is able to run via command line or a simple web-interface that was developed for this project.

2. LITERATURE REVIEW

There has been much research on using computer vision for exercise or rehabilitation analysis. Pose Trainer [2] is one such method that uses OpenPose, dynamic time warping and exercise specific rules to correct exercise posture. A drawback of this method is most exercises selected were static, such as dumbbell curls, and would not perform well with dynamic exercises. A method using background subtraction proposed by Faustine et al. [3] is an interesting method that would allow for capturing of key-frames such as the start and end of the exercise but does not solve the problem statement of a pose corrector for a dynamic exercise. Talal A. and Chen Chen [4] paper uses OpenPose and neural networks to identify exercises as well as the ability to count the number of repetitions via angle calculation. An additional note is that most of the methods used only incorporate 2D pose or visual data, the team was unable to find any methods that use 3d data for pose correction.

As the basis of the feature extraction from the image or video, we explored multiple 2D and 3D pose-detection methods to extract the pose of a single-person frame in real-time. Toshev et al. [5] developed DeepPose, a deep neural network to detect joint location in 2014, the first to do so with deep learning. The stacked hourglass method, developed by Newell et al. [6], which consists of multiple bottom-up and top-down inferences utilizing both global and local information to make predictions. UniPose and UniPose-LSTM [7] adopts an additional LSTM layer adding historical features from the previous frames. Zhe Cao et al. [8] developed Openpose which focused on multi-person pose estimation in real-time and is now a common standard which other pose estimation methods are compared against. BlazePose, Bazarevsky et al. [9], was developed for real-time inference and reduces processing time by utilising a detector-tracker method and is also able to provide a rough 3D pose estimate. Unlike earlier methods, BlazePose estimation is not based on heatmaps during inference which allows it to scale to include 3D support, and additional key-points / key-point attributes. This allowed BlazePose to outperform OpenPose on fitness use cases while maintaining a significantly faster compute speed. 3D prediction models [10][11] generally requires a higher-end GPU to enable real-time inference.

For a more standardized and structured exercise regime, incorporation of a rep counter to log user's exercise routine is paramount. Soro et al employed wearable IMU sensors and vibration sensors from smart watch applications to train a deep neural network to achieve a 91±1 percent prediction accuracy of the repetitions [12]. Although training a deep neural network does not require feature engineering, which is also highlighted as an important advantage over classical method in the article, reliance on deep neural network meant a higher computational cost especially for applications that need to include both classification and repetition counting. Several research in the past have focused on employing Human Activity Recognition (HAR) through motion sensor signals (including accelerometers, gyroscope, and linear acceleration sensors) and have achieved rep counting accuracy of over 90±1 percent repetition count.

Exercise classification models have been extensively developed for physiotherapy and posture correction [13] [14] [15]. These research articles have focused on the use of advanced machine learning models such as k-nearest neighbors, Random forest, Support Vector Machine, Convolution Neural Network (CNN) and Convolution-Recurrent Neural Network (C-RNN) for classification. Some of these models primarily obtain data from sensory signals of wearable on users, which is then further processed to obtain information on the movement of the users. Other exercise classification models rely on active posture estimations to obtain key-frame references for further processing. Elforaici et al [16] used silhouette segmentation to develop a 3-D CNN model for exercise classification.The research paper also compared the use of RGB images as well as depth maps in the model classification process and concluded that RGB-D datasets in general provided better classification accuracy. Gupta et al [15] used wearable tri-axial accelerometer sensors and streamlined the time-series senor data to a fully convolutional and Long Short Term Memory (LSTM) model.

It is found that most applications if not all uses rules based approach to evaluate the postures and movement of users to provide guidance and feedback even though the method to obtain the spatial information of the user's body key-points may differ . Generally, they will calculate the distance and angles between key-points of human body parts such as the knee or elbow using the coordinates of those key-points that were extracted from the image or video. Pose Trainer [2] evaluates the user's posture and motion by calculating the angles and distance between body parts using thee coordinates of the corresponding key-points extracted from OpenPose. Similarly, Dorado et al [17] also uses rules based approach in their ArthriKin system that supervises and provides feedback to patients that exercise at-home as part of their rheumatoid arthiritis rehabilitation. Despite the system using Microsoft Kinect to extract the skeleton information instead of the more commonly used, OpenPose, it also evaluates the rules such as the range of motion of exercise based on the angles and trajectory of body parts or limbs calculated from the coordinates of key-points on the body that were obtained from Microsoft Kinect SDK.

3. DATASET

As the self-collected and curated dataset required time to compile, existing datasets were sourced for initial model performance testing and optimizing. The Deep Mind funded Kinetics human action dataset, which replenishes and extends from the kinetics-700 dataset, covered a diverse range of repetitive human actions including an extensive exercise dataset [18]. Another such publicly available dataset is the Countix dataset, which was eventually used in the development of the Temporal Self-smilarity Matrix (TSM) based RepNet repetition counter [19]. Exercise videos were also sourced from the internet to further expand the dataset. The varying frequency based repetitive actions contained in this dataset form excellent testing ground for the rep counter module. Apart from the two official dataset sources, trainers and exercise experts provided a strong collection of data on the correct form and posture for initial model testing.

The self-collected dataset comprised of Mono-D video sets, a collection of matching and synchronised monochrome and disparity videos. The data was collected by utilising an OpenCV OAK-D camera which uses two global shutter monochrome cameras to generate a depth map with its onboard Myraid X VPU. An RGB camera is also present on the device, but it was not used as the device had sync issues with the RGB camera to the depth information that were only solved towards the end of the project. A series of exercises were performed and recorded to create this dataset, which included 10 sets of 5 repetitions of squats and push-ups both perpendicular and 45 degrees from the camera as well as 2 videos of random movement that represented no exercises being performed.

Data augmentation was used to artificially increase our collected data for training of neural networks. This was done by randomly translating the output of pose features in the x,y,z directions by a value of 0.95 - 1.05. Each video went through 4 instances of augmentation which generated 5 feature samples per video.

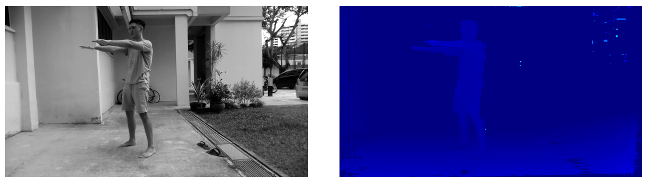

(a) (b)

Fig. 1. (a) Monochrome and (b) disparity images

4. PROPOSED SYSTEM

4.1. Pose Feature Extraction

As we wanted to create a system that was invariant to the direction the user was facing, 3D pose data was essential to generate the pose of the user for analysis. A 1280x800px video-pair consisting of monochrome and disparity information was used as inputs to the system.

As the extraction of the 3D points are heavily dependant on the predicted 2D key-points generated by the pose-estimation model, care had to be taken to identify the most suitable model. The team decided to use MediaPipe's BlazePose for its speed and straight forward API which allows us to directly use the model without training via simple installation. The model was also chosen as it showed the best balance between inference speed and accuracy of the selected key-points compared to models such as OpenPose.

A frame from the monochrome video is taken as the input for Blaze-Pose and an array of 33 key-points which are normalised to the image dimensions are returned from the model. As the system does not require all 33 key-points, a subset of 12 key-points were extracted for further processing.

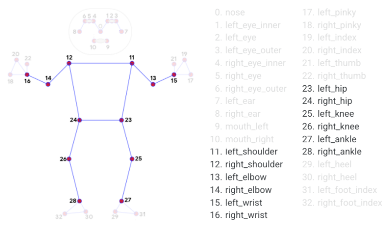


Fig. 2. Selected key-points from BlazePose model

After extracting the subset of key-points, the u and v coordinates of each key-point are used to determine the disparity. The maxima of a 7x7 neighbourhood centered on the u and v coordinates is used to extract the disparity value of each key-point on the disparity map. The reason the maxima is selected is the u and v values may miss the exact location of the interested joint which can produce a low disparity value. The depth can then be calculated by performing the following calculation:

$$z_{u,v} = b * f / d_{u,v} \tag{1}$$

Where z is the depth in mm, b is the baseline in mm, f is the focal length in pixels and d is the disparity. An additional step is taken to transform the z value with reference to the camera origin to the x,y camera reference plane. The distance of the key-point from the centre pixel is calculated and put into the formula below:

$$H = cos(HFOV/2) \tag{2}$$

$$z_{x,y} = (H + ((1 - H)/640) * dist_{x,y}) * z_{u,v} \tag{3}$$

Where HFOV is the horizontal field-of-view of the camera, 73.5 degrees for the OAK-D.

Once we have calculated the z values of each key-point, we would have to calculate the x and y values in a camera frame. This is done by multiplying the inverse of the intrinsic matrix to our matrix of u, v and z values.

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} f_x & 0 & u_0 & 0 \\ 0 & f_y & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}^{-1} * z * \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \tag{4}$$

The intrinsic matrix shown is a simplified version, as there may be non-zero values due to the calibration of the camera. The 3D pose would then be translated so that the origin is the key-point of the left-hip, this is simply done by subtracting the x,y,z values of the left-hip from all the key-points. The values are then normalised by dividing the the key-points by the distance between the left-hip and left-shoulder, the distance is then held in memory to be used in subsequent frames.

As the orientation of the 3D pose would have to be constant, rotations would be necessary. We assume that the only rotation required is in the X-axis, ie. the angle of the user facing the camera. The distance of the hips in the x-direction is used as reference for this rotation and is extracted in the initial frames before the exercise begins. If the threshold of the distance is broken, we apply a fixed rotation to the 3D pose, which we assume to be 45 degrees.

A median filter is then applied to the pose that get the median value over 5 frames. This is to ensure that any noise in the output is suppressed.

As occlusion is a major problem when extracting the depth of the images, mirroring has to be performed to get a clean output. This is possible as the exercises chosen (squats and push-ups) and their common errors tend to be symmetrical. As the requirement for use is for the user to face the right-side of the camera, the left-side of the user would be used as a more accurate non-occluded value. The key-points of the left side are then mirrored about the left-hip + a predetermined amount.

The final 3D output can then be projected to 2-D enabling the estimations of the key-points of the user through 3 views: side view, front view and top view.

The key-points allow for skeletal extraction of the user performing the exercise. This feature extracted is useful for key-frame identification and further processing.

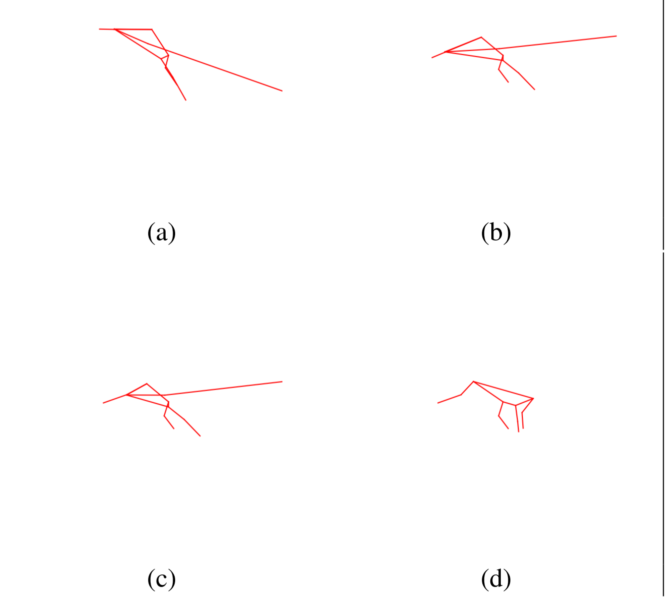


Fig. 3. Transformed Pose: (a) Raw output, (b) rotated, (c) median filtered, (d) mirrored

4.2. Key-Frame extraction and Repetition counter

Several repetition counter methods were experimented with during the system design, however they were observed to be highly sensitive to small variations in the position of key-points often leading to multiple counts per rep and therefore suffering from poor accuracy. Moreover, in dynamic conditions these solutions were observed to be easily skewed by frequency changes. Advanced solutions like Rep Net [19] were too computationally intensive and thus a balance was required in the trade off between accuracy and computational cost. A novel geometric gradient-descent repetition counter based on the euclidean distance traversed during each rep was developed. The rep counter used statistical finite difference theorem and an initialised starting reference pose to compute the real time traversal of the side view projection as the user performed the exercise.

$$D_x = \sum_{i=0}^{n}[x_i - x_{ref}]^2 \quad (5) \qquad D_y = \sum_{i=0}^{n}[y_i - y_{ref}]^2 \quad (6)$$

The counter was activated based on the statistical finite difference theorem that computed the distance-based gradient across two frames and actively monitored for gradient sign transformations from negative to positive indicating a rep completion. As the exercises chosen had larger motions in the vertical direction, the y^{th} directional distance computation was evaluated using the finite-difference theorem as shown below:

$$F_{yi} = \frac{f(Dy - \delta y) - f(Dy)}{\delta y} \qquad (7)$$

Moreover, counter computed on D_y was also found to have a better accuracy results. Typically, the gradient sign transition (from F_{yi-1} to F_{yi}) represents a maxima or minima and a single rep is effectively calculated at 2 consecutive maximas. To support the rule based systems, the key-frame detection was computed similarly. However, in the key-frame detection, a combination of x^{th} and y^{th} directional distance computation was used to detect points of both maxima and minima. The gradient deviation (from $F_{(x,y)i-1}$ to $F_{(x,y)i}$) using finite-backward difference theorem was used to evaluate both maxima and minima.

$$D_{x,y} = \sum_{i=0}^{n}[(x,y)_i - (x,y)_{ref}]^2 \qquad (8)$$

$$F_{x,y} = \frac{f(Dx,y - \delta x,y) - f(Dx,y)}{\delta x,y} \qquad (9)$$

The statistical based approach yielded positive results as shown on the self-collected dataset below with the maxima and minima correctly identified.

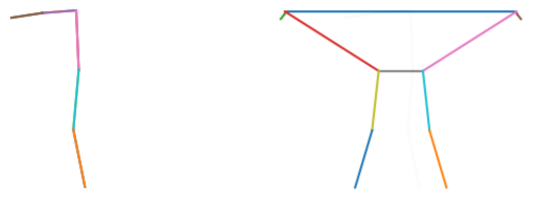


Fig. 4. Side and front view projections from 3-D to 2-D transformation

4.3. Exercise Classifier

The next phase of the proposed system involved exercise classification for the activation of the rule-based engine. Several classification techniques including machine-learning and deep-learning based models were tested on the basis of accuracy and inference speed. In the end, a 3DCNN model was selected for its accuracy as well as inference speed.

The 3DCNN model was trained on the augmented dataset, with the input being a 30 frame window of pose outputs. This resulted in an input size of 30x3x12x1 to the 3DCNN model. As the goal was to be able to identify when the user stops

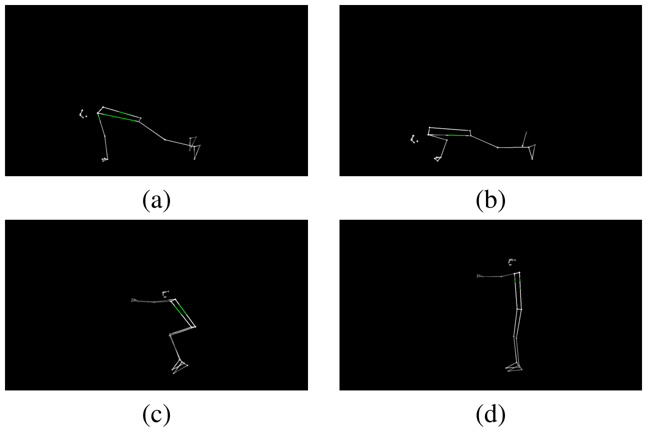
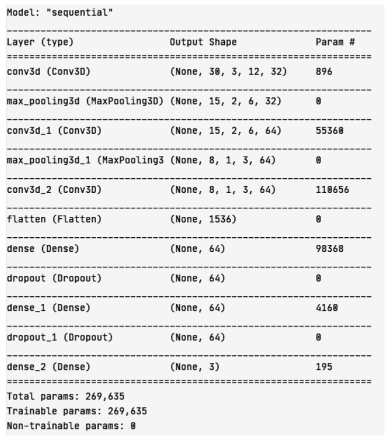


Fig. 5. Key-frame maxima and minima detection for pushup and squats respectively

the exercise, a section of the start and end of each array was labeled as no exercise. The model was then trained over 10 epochs as it was a simple 3DCNN with 11 layers and more epochs would lead to over-fitting of the model.



```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
conv3d (Conv3D)              (None, 30, 3, 12, 32)     896       
_________________________________________________________________
max_pooling3d (MaxPooling3D) (None, 15, 2, 6, 32)      0         
_________________________________________________________________
conv3d_1 (Conv3D)            (None, 15, 2, 6, 64)      55360     
_________________________________________________________________
max_pooling3d_1 (MaxPooling3 (None, 8, 1, 3, 64)       0         
_________________________________________________________________
conv3d_2 (Conv3D)            (None, 8, 1, 3, 64)       110656    
_________________________________________________________________
flatten (Flatten)            (None, 1536)              0         
_________________________________________________________________
dense (Dense)                (None, 64)                98368     
_________________________________________________________________
dropout (Dropout)            (None, 64)                0         
_________________________________________________________________
dense_1 (Dense)              (None, 64)                4160      
_________________________________________________________________
dropout_1 (Dropout)          (None, 64)                0         
_________________________________________________________________
dense_2 (Dense)              (None, 3)                 195       
=================================================================
Total params: 269,635
Trainable params: 269,635
Non-trainable params: 0
```

Fig. 6. 3DCNN Model Architecture

The model was then tested by the overall accuracy of the video as well as on a frame-by-frame bases. When testing over the whole video, the video is tagged to the class that has the most outputs over the whole length of the video. This resulted in 100% accuracy of the model. On testing on a frame-by-frame basis we have the performance below:

Exercise	Precision	Recall
No Exercise	0.30	0.77
Squats	0.96	0.92
Sit-ups	1	0.85

Table 1. Precision and Recall of 3DCNN Classifier

The model is further improved as a filter is applied over 5 consecutive outputs that selects the class with the maximum count as the final output. This ensures that false classifications that appear over 1 or two frames are removed. The resulting output is a fairly smooth output during exercises but with some false classification of squats during no exercise sequences.

4.4. Exercise Rules Engine

The rules engine consists of set of rules for each type of exercise and a counter-management. After the 3-D CNN has determined the type of exercise, the rules engine will activate the set of rules corresponding to that exercise type and apply the evaluation of the rules on the image. The rules are mainly evaluated by calculating the angles and distance between body parts using the coordinates of key-points pairs. For example, the coordinates of the 3 key-points, hip-knee-elbow are used to calculate the angle at the leg during push-up exercise to ensure the user legs are kept straight throughout the exercise.

The minima and maxima key-frames detection enable the rules-engine to selectively trigger certain rules that required to be evaluated only during the maximum or minimum in user movement in the exercise. This is because rules such as angle of the leg at the knee has to be evaluated at the lowest point of bend for squat exercise and not throughout the exercise to ensure the correct movement. In this example, when the minima key-frame is detected, the angle at the knee will be evaluated to be lower than 90 degree at the lowest point for a valid count.

For squat exercise, the rules implemented are: (i) angles between arm and vertical plane to be around 90 degree; (ii) angle at the knee to be 90 degree or less at the lowest point of squat; (iii) angle at the knee to be 180 degree at the highest point of squat. For pushup exercise, the rules implemented are: (i) angles at the elbow to be around 90 degree or less at the lowest point of squat; (ii) angle at the knee to be around 180 degree; (iii) angles at the elbow to be around 180 degree or less at the highest point of pushup.

Whenever a rule is broken, the system will display a message stating the user's mistake on the image allowing the user to correct the movement immediately. In the consideration of user experience, the message will continue to be displayed on the output image after the rules are broken for a set of frames based on a configurable decay value. The counter-management takes input from the repetition counter computed earlier in the system. Upon increment in the repetition counter input, the counter-management will increment the valid rep counter if no rules were broken in the current single repetition of an exercise. The mistakes feedback and repetitions count are displayed in the output video by writing onto the image frame. Finally, the counter management will output the final count based on the valid rep counter and result of the rules-engine containing the list of broken rules detected to be displayed to the user in the web-interface at the end for user guidance and post review of their exercise.

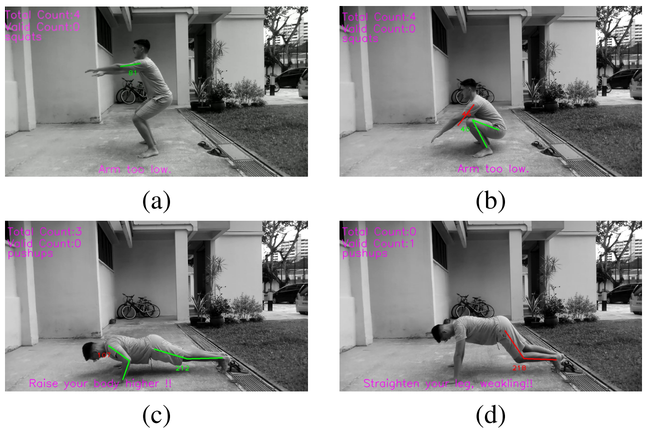


Fig. 7. Implementation of rules: (a) Correct squat, (b) Incorrect Squat, (c) Correct Push-Up (d) Incorrect Push-Up

The team has also developed a fully fledged web-interface containing the full system demonstration (insert youtube link).

5. EXPERIMENTAL RESULTS

5.1. Pose-Estimation

Three options was initially considered for the pose-estimation, OpenPose running on the host computer, an optimized OpenPose via OpenVino running on the OAK-D via its internal VPU and BlazePose running on the host computer. The three options were tested and compared based on frames-per-second and accuracy with accuracy based on visual estimation of the team. Below are the results of the FPS testing:

Model	FPS
OpenPose (Host/GPU)	4.5
OpenPose (OAK-D/VPU)	7.5
BlazePose (Host/CPU)	26

Table 2. FPS Test for Pose-Estimation

We can see that BlazePose vastly outperforms both OpenPose variants in terms of FPS and is able to do so with very similar accuracy results. Thus, we selected to used BlazePose as the pose-estimation method.

5.2. Repetition Counter

The rep counter uses a statistical approach to check for the gradient transformations in the finite-backward difference computed over time. As the two exercises involve vertical movements instead of horizontal movements, the y^{th} directional movements captured from the side-view projects were used for the computation. Frame sampling is an important hyper-parameter for effective rep counting and the hyper-parameter was selected on evaluating a subset of videos at different sampling rates. The evaluation results are shown in figure 6 with the best accuracy observed at a frame-sampling rate of 32 for push-up and 45 for squats. The system relies on the exercise classification algorithm for selecting the appropriate frame sampling rate.

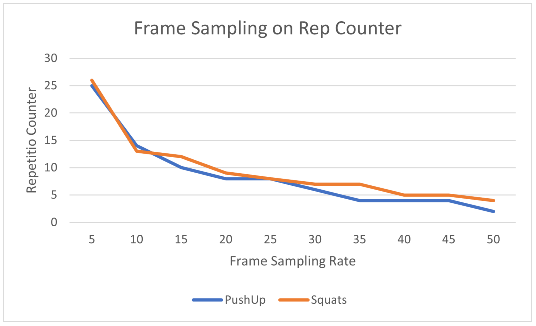


Fig. 8. Effect of frame sampling rate on repetition counter for 5-count push-up and squats video

As with other rep counters, the primary metrics for the evaluation is accuracy. The rep-counting accuracy was determined on the basis of whether the reps evaluated were within ±1, ±2 and ±3 of the actual counts.

Exercise	±1	±2	±3
Pushups	0.90	1.00	1.00
Squats	0.85	0.95	1.00

Table 3. Repetition Counter Accuracy Results

Limitation to the rep counter include abrupt and aggressive frequency changes to the exercise (for e.g. exercise is done excessively fast or excessively slow). In situations such as these, the frame sampling is unable to catch-up with the increased frequency of the exercise and thus is unable to detect the gradient transformations. A possible solution to this could be auto-adjusting the frame sampling rate according to the frequency at which the exercise is performed. Another limitation observed is its sensitivity to small movements during the start of the video which leads to counter counting despite the exercise not being performed. This issue is also addressed in the system specification whereby the requirement is for the user to screen off static movements prior to the start of the exercise.

5.3. Exercise Classifier

As mentioned earlier, multiple methods were proposed for the exercise classification. We compared a Support Vector Machine (SVM) model with inputs computed from Local Binary Pattern Features, a light-weight 2DCNN method that is trained on the distance based vector, a heavy-weight 2DCNN model that was trained on key-frames, a SVM model trained on features extracted via HoG, and a 3DCNN model which was trained on a window of extracted 3D pose-estimations.

Accuracy is based on classifying the overall video. The classified output with the most counts was selected as the representative class of the video and is used for the comparison.

Approach	Acc.	Inference Speed (ms)
SVM-LBP	0.8537	3
SVM-HoG	1	294
LW-2D	0.5366	979
HW-2D	0.948	936
3DCNN	1	38

Table 4. Comparison of Exercise Classifier Approaches

Both SVM-HoG and 3DCNN performs with 100% accuracy when classifying the overall videos, but the inference speed of the 3DCNN is nearly 10x faster. SVM-LBP is the fastest out of all the methods but with the 2nd lowest accuracy. The 3DCNN approach was also the only approach that is able to accurately identify no exercise during the start and end of exercise videos where no exercises were being performed.

As the dataset was not very diverse, it was important to add dropout layers after the dense layers in the model. After experimentation a value of 0.3 was selected for the final dropout layers. The model was also trained on 10 epochs after testing various lengths, with the accuracy of a hold-out set dropping from 0.94 to 0.89 when increasing the epoch to 15.

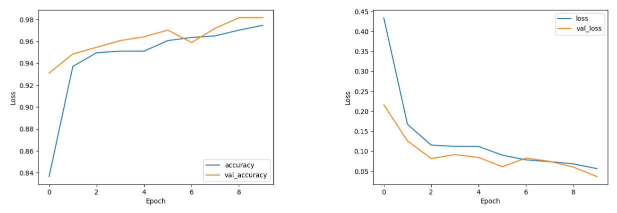


Fig. 9. Accuracy and Loss of 3DCNN Training

5.4. Exercise Rules

We attempted to implement distance based rules such as distance between the feet or hand by using the depth, z value provided by the extraction step. However, the computed distance values are highly unstable and fluctuated greatly of up to 300 percent jump between the frames. Hence, it is deemed not usable. X and Y coordinates cannot be used to compute this distance because it would not be able to handle the varying angle of the user. A multi-angle cameras setup would resolve this limitation and enable more comprehensive rules to be implemented for better guidance to the user.

Certain rules such as to check if user has lowered the body sufficiently can only be triggered at a specific point of the rep. Initially, these rules are triggered based the movement of elbow or knee along Y-axis. The user body is assumed to be at the lowest or highest point of the exercise when there is a change in the direction of those key-points' movement along the Y-axis. The Y-coordinates of the elbow or knee's key-points were used to compute direction of the movement. However, Y-coordinates of the key-points were found to fluctuates too frequently and caused false trigger of these rules. The alternate method to trigger the rules upon detection of maxima and minima key-frame was chosen as it is more precise. However, this method has a limitation because key-frame are extracted with sampling of frames. Consequently, the rules may not be able to trigger on the exact frame where the user is at the lowest or highest point of the exercise. Subsequently, when the rules are triggered a few frames later, the angle at the elbow or knee may have exceeded the values, thus caused the rule to be incorrectly evaluated as broken.

Due to the same limitation of using maxima / minima key-frame as trigger in the rules, deque and output averaging method was not used to stabilize the evaluation result of the rules that may fluctuates caused by the fluctuation of the coordinates. Instead the angles and distances values to be checked in the rules engine are given 10 percent allowance to mitigate this limitation.

5.5. Performance of Overall System

The performance of the overall system was tested based on the accuracy of total number of valid counts per video.

Exercise	± 0	± 1	± 2	± 3	± 4	± 5
Squats	0.4	0.6	0.7	0.7	0.7	1
Squats-45	0.2	0.2	0.4	0.5	0.7	1
Push-ups	0.1	0.3	0.7	0.8	0.9	1
Push-ups-45	0	0.2	0.4	0.4	0.7	1

Table 5. Valid Counts Accuracy Results

As seen in the table, the performance based on valid counts is low with some videos not detecting any valid counts when there should be 5. The performance in videos where the person is at an angle consistently performs worse as well, likely due to the rotation not being accurate as a predetermined angle was used. There was an attempt to use the angle of the hips or knees of the detected pose to calculate the required rotation, but the output of this method was not consistent as occlusion is common and there are many instances where the key-points miss the joints. These instances where the key-points miss the joints also presented another problem where the depth values extracted were wrong, causing the calculated x and y values to have an error as the calculation of these values are based on the z value.

A problem which existed in a few videos is that a pose may be detected in an area where no person exists, this causes the system to initialise with the wrong normalisation and rotation parameters which resulted in the subsequent frames to be processed with the wrong data leading to false mistakes or measurements.

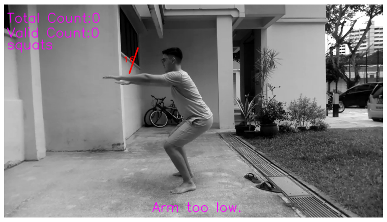


Fig. 10. Example of wrong pose detection

6. CONCLUSIONS AND FUTURE WORK

Gym-buddies addresses the importance of correct form and posture in regular workouts. The proposed system allows users to submit their exercise videos in Mono-D format and transforms the 3D-pose information derived from BlazePose and the disparity map into projections of the front, side and top view of the user performing the exercise. It employs a statistical spatial-frequency based approach of repetition counting as well key-frame detection and is connected to a comprehensive rule based engine for real time posture correction. Exercise classification is performed using the 3DCNN model that provides competent accuracy results with fast inference speeds.

The road-map to system finalisation involved experimenting with many different methods to achieve the system goals. As seen in the experimental results, there are many areas for improvement and future work.

The pose-estimation could be improved by finding a model that would more accurately track the key-points of the joints, which would reduce the errors when extracting the depth from the disparity map. Another way this could be improved is to find or create a pose-estimation model that uses depth information along with RGB to directly estimate the joint in 3D, this may be a more robust method and reduce the amount of errors when deriving the 3D pose of the user.

The repetition counter detection accuracy can be further improved by performing a 4-point gradient transformation check instead of the current 2-point check. Moreover, the performance can be improved by using temporal frequency to fine-tune the critical frame sampling hyper-parameter on a real-time basis.

The exercise classifier could be improved by increasing the amount of data available for it to be trained on. The annotation of data is also an important part of the system. Currently a rough estimate is used to determine the transition between the frames of a specific exercise and no exercise. If more time was available, the exact frames could be determined for this transition that is unique to each video for more accurate training of the model. It would also be good to have more people perform these exercises so that the data is not biased, though the normalisation earlier in the pipeline would alleviate this slightly.

The rules engine can be further improved with multi-view camera setup to extract a more precise 3-D spatial information. This would allows more rules to be implemented to provide a more comprehensive feedback and guidance to users. It also enables various types of exercises to be added into the system. The detection of maximum and minimum point of the movement can be improved to give a more accurate trigger point and evaluation of rules that require to be evaluated at those points.

7. CONTRIBUTIONS

Chun How Oh created the rules engine of the project which allows identification of incorrect poses and output the results with valid repetitions count and list of broken rules during an exercise. He experimented on exercise classification model by developing the HoG-SVM model for feature extraction and classification of the exercise. He collected exercise videos from the internet to be used as the initial dataset in the beginning of the project.

Mikhail Kennerley performed the feature extraction of the pose estimation, which includes transforming, filtering and cleaning the extracted 3-dimensional pose data. He explored and experimented various pose estimation models to use in the project. He collected the RGB-D dataset by performing and recording the required exercises. He created the augmentation pipeline to generate additional pose-data from existing videos. He contributed in exercise classification by creating the 3D-CNN model which used pose coordinates as features.

Vidish Mehta explored various deep learning, statistical and spatial-frequency based approaches for repetition counting and developed the frame-sampling enabled finite-backward difference based spatial rep counter. He experimented on exercise classification models and was responsible for the feature extraction, mapping and development of LBP-SVM and 2D-CNN exercise classifiers. He also created the web-based user interface for use.

All team members contributed to the creation of the project report and the mid-project presentation.

Code is available at:
https://github.com/mecarill/ISS-GymBuddies

8. ACKNOWLEDGEMENT

The team would like to thank Dr. Tian Jing and Dr. Jen Hong for teaching us important concepts on computer vision and video analytics which were critical to the success of this project as well as their continuous guidance on the development of the project.

9. REFERENCES

[1] Bruno T. Saragiotto, Carla Di Pierro, and Alexandre D. Lopes, "Risk factors and injury prevention in elite athletes: a descriptive study of the opinions of physical therapists, doctors and trainers," *Brazilian journal of physical therapy*, vol. 18, no. 2, 2014.

[2] Steven Chen and Richard R. Yang, "Pose trainer: Correcting exercise posture using pose estimation," *CoRR*, vol. abs/2006.11718, 2020.

[3] Faustine John, Irwandi Hipiny, Hamimah Ujir, and Mohd Shahrizal Sunar, "Assessing performance of aerobic routines using background subtraction and intersected image region," *CoRR*, vol. abs/1810.01564, 2018.

[4] Talal Alatiah and Chen Chen, "Recognizing exercises and counting repetitions in real time," *CoRR*, vol. abs/2005.03194, 2020.

[5] Alexander Toshev and Christian Szegedy, "Deeppose: Human pose estimation via deep neural networks," *CoRR*, vol. abs/1312.4659, 2013.

[6] Alejandro Newell, Kaiyu Yang, and Jia Deng, "Stacked hourglass networks for human pose estimation," *CoRR*, vol. abs/1603.06937, 2016.

[7] Bruno Artacho and Andreas Savakis, "Unipose: Unified human pose estimation in single images and videos," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[8] Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei, and Yaser Sheikh, "Openpose: Realtime multi-person 2d pose estimation using part affinity fields," 2019.

[9] Valentin Bazarevsky, Ivan Grishchenko, Karthik Raveendran, Tyler Zhu, Fan Zhang, and Matthias Grundmann, "Blazepose: On-device real-time body pose tracking," 2020.

[10] Gyeongsik Moon, Ju Yong Chang, and Kyoung Mu Lee, "Camera distance-aware top-down approach for 3d multi-person pose estimation from a single rgb image," 2019.

[11] Julieta Martinez, Rayat Hossain, Javier Romero, and James J. Little, "A simple yet effective baseline for 3d human pose estimation," in *ICCV*, 2017.

[12] Simon Tanner Andrea Soro, Gino Brunner and Roger Wattenhofer, "Recognition and repetition counting for complex physical exercises with deep learning," in *NCBI*, 2019.

[13] Steven Chen and Richard R. Yang, "Pose trainer: Correcting exercise posture using pose estimation," 2020.

[14] Jaehyun Lee, Hyosung Joo, Junglyeon Lee, and Youngjoon Chee, "Automatic classification of squat posture using inertial sensors: Deep learning approach," *Sensors*, vol. 20, no. 2, 2020.

[15] Rinki Gupta, Devesh Saini, and Shubham Mishra, "Posture detection using deep learning for time series data," in *2020 Third International Conference on Smart Systems and Inventive Technology (ICSSIT)*, 2020, pp. 740–744.

[16] Mohamed El Amine Elforaici, Ismail Chaaraoui, Wassim Bouachir, Youssef Ouakrim, and Neila Mezghani, "Posture recognition using an rgb-d camera : exploring 3d body modeling and deep learning approaches," 2018.

[17] Javier Dorado, Xavier del Toro, Maria J Santofimia, Alfonso Parreño, Ruben Cantarero, Ana Rubio, and Juan C Lopez, "A computer-vision-based system for at-home rheumatoid arthritis rehabilitation," *International Journal of Distributed Sensor Networks*, vol. 15, no. 9, pp. 1550147719875649, 2019.

[18] Lucas Smaira, João Carreira, Eric Noland, Ellen Clancy, Amy Wu, and Andrew Zisserman, "A short note on the kinetics-700-2020 human action dataset," 2020.

[19] Debidatta Dwibedi, Yusuf Aytar, Jonathan Tompson, Pierre Sermanet, and Andrew Zisserman, "Counting out time: Class agnostic video repetition counting in the wild," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.