**Summary of Research Papers**

**1. Applying Pose Estimation to Predict Amateur Golf Swing Performance Using Edge Processing**

* Contains information about setting up the camera and the shot. “Down the line” angle view.
* Used TensorFlow’s PoseNet
* No formulae mentioned
* The varying methods of capture resulted in swing pose accuracies from 20% to 81.1%.
* Limitations of a bad dataset causing high inaccuracies (low lighting conditions and lack of data to identify ‘bad swings’)
* Used a publicly available repository called project-posenet that was created for the TPU by Google. Included in the repository was a version of PoseNet that was compiled into a TensorFlow lite model to run on the TPU. ([google-coral/project-posenet: Human Pose Detection on EdgeTPU (github.com)](https://github.com/google-coral/project-posenet))
* Two different models were used based that were included in the project- posenet repository, one that read in images at 641 x 481 and another at 1281 x 781. Running the lower resolution model indicated an approximate 30 frames per second (FPS) limitation on the Coral development board. The higher resolution model saw an expectedly lower performance of around 20 FPS.
* To avoid processing delays on the TPU and to allow us to use existing videos as samples, we chose to post-process the data by pulling frames from captured videos.
* 200 samples from over 20 golfers. The videos analyzed came from YouTube

The general process for collecting the data and analysing the results are:

1. Video data of a swing from a down range camera position is collected

2. Video is converted to individual frames

3. PoseNet machine learning model is run on the frames on the TPU

4. Frame-by-frame pose positions are collected

5. Data is smoothed using a Savitzky-Golay Filter to help minimize sudden changes caused by PoseNet inaccuracies

6. Critical points in the swing and their associated frames are found using the feature table below

**Golf Swing Actions:**

\_ P1- Address

\_ P2- Club shaft parallel with ground on takeaway

\_ P3- Lead arm parallel with ground

\_ P4- Top of backswing

\_ P5- Lead arm parallel with ground on downswing

\_ P6- Club shaft parallel with ground on downswing

\_ P7- Impact

\_ P8- Club shaft parallel with ground on follow-through

\_ P9- Trail arm parallel with ground on follow-through

\_ P10- Finish

**2. Accurate and Efficient 3D Human Pose Estimation Algorithm using Single Depth Images for Pose Analysis in Golf** (Unrelated to our needs)

-Uses Kinect and Vicon motion capture systems, not mobile phone cameras

-7 Swing Actions

-Uses Decision Trees

**3. How Can I Swing Like Pro? Golf Swing Analysis Tool for Self Training (Very short paper)**

-Uses 2 Videos:

* An expert’s swing
* A beginner’s swing

Compares the two to find differences.

Summarized:

* Easier for us as we have an expert’s swing being analyzed by the machine and not us as ‘outsiders to golf’
* The frames are synchronized using TCC (Temporal Cycle-Consistency) as the 2 videos can have different timings for the swings
* Uses the difference between latent vectors to calculate how close the swings are to each other
* Pearson’s correlation value is calculated for the elbow, shoulder, neck, head, wrist, spine, knee, foot, hip, and the whole body.
* HRNet and simple linear network with Procrustes to allow users to visually recognize the difference in motion between them and the expert   
  -**Database** used: GolfDB which contains 1400 vids of experienced swingers.
* The pose of the golf club is also considered for the swings

**4. Golf Swing Correction Based on Deep Learning Body Posture Recognition** (Best suited to our needs)

* Uses OpenPose(more info on paper 4.1) to detect key points and calculates arm angle, wrist distance, hand distance, displacement of the nose, leg angle and spine line features.
* Most of these use simple formulae utilizing Euclidean distance alongside the change in coordinates with an interval of a few frames.
* All the formulae used are included in the report.
* There are a few formulae to fix misdetections too.
* The accuracy is at 98.7% without misdetection fixes, with a sample size of 25,000 and with the misdetection fixes, the accuracy is at 99.9%.

**Golf Swing Actions:** setup frame, back-swing frame, top frame, release frame, hit frame, foreswing frame, and closing frame

**5. Human Pose Estimation for Training Assistance: A Systematic Literature Review**A general context of HPE usage for Exercising.

* Motion blur, ambiguity between the poses and loose fitting clothing can cause misdetections
* Uses Belief Propagation algorithm to estimate pose samples
* Applied trained randomized decision trees to associate each pixel with a particular bone segment for labelling.
* Shape Context is used to extract features and compare query images against the dataset samples.
* Human Mesh Recovery to reconstruct a full 3D mesh of a human body

**6. Golf video tracking based on recognition with HOG and spatial–temporal vector**

Tracking framework based on object recognition that uses ML with histogram of oriented gradients (HOGs) and spatial-temporal vector.

Image quality is limited by these factors:

* Hand and club are very small/ lack details
* Blurry captures for fast swings
* Bad camera quality

**3 steps:**

* Initialization
* Object trajectory prediction
* Object recognition

*Initialization*

Finds initial positions of hand and club. Candidate windows calculated for the hand and the club by using the player’s body position as reference. Player’s body detected automatically using ACF (aggregated channel feature).

*Object trajectory prediction*

Estimates the possible object position in current frame. Predict x and y coordinates using the four previous tracked positions. *( formulae included in report)*

*Object recognition*

Recognizes the object in a searching window. Uses a ‘feature based’ approach. Uses a simpler feature descriptor (HOG) so that the mobile phone can bear it.

For training, OpenCV is used. Trained a boosted classifier.

Sliding window centered on the predicted position with a larger width and height than the patch. If score of patch is not satisfactory, a sliding window is used will be checked until the score is satisfactory.

**Training**

Database contains 13,287 positive samples and 147,671 negative samples from 99 videos. 80:20 training:test split.

**Tracking**

If a patch contains a satisfactory score in the next sliding window, then the program skips to the next frame to avoid misrecognition. Fixes blurry or blocked frames. **The precision and recall rate are both above 97%**.

**Future Improvements**

Limited by small dataset, no videos shot in the dark or other bad situations, did not apply deep learning.

**7. Baseball Swing Pose Estimation Using OpenPose (*Processes Discussed in Detail in this paper*)**

Real time detection. Tradeoffs: accuracy for real time detection

Uses CMU’s OpenPose.

The model has good accuracy and real time performance. Uses the first 10 layers of VGG-19 model ([Understanding the VGG19 Architecture (opengenus.org)](https://iq.opengenus.org/vgg19-architecture/)) to extract feature maps. These feature maps are processed with CNNs to generate 2 branches. Branch 1 is the confidence map of each keypoint that outputs a feature map for each keypoint.

Branch 2 predicts a set of PAFs (part affinity fields). Each limb is represented by a 2D vector.

**Method**

1. **Getting the keypoints:** input baseball swing vids to OpenPose to get keypoint info. The body\_25 model([openpose\_train/README.md at master · CMU-Perceptual-Computing-Lab/openpose\_train (github.com)](https://github.com/CMU-Perceptual-Computing-Lab/openpose_train/blob/master/experimental_models/README.md)) of OpenPose was used to output 25 keypoint coordinates. The resolution of the network is 656 x 368 for optimal speed and accuracy balance.
2. **Detecting only the hitter:** set the threshold higher than the preset value of 0.4. Only clear body parts are rendered at threshold higher than 0.5.
3. **Table

   Description automatically generatedEstimating the Swing Pose:** angle ranges for predicting a ‘good’ swing vs a ‘bad’ one. The report contains a set of ‘custom rules’ shown in the figure below.
4. **Calculating the angles:**  the following formula was used to calculate the angle, here ‘U’ refers to a vector of the limb from one end to the midpoint and ‘V’ refers to the vector of the limb from another end to the midpoint. Such as left shoulder to left elbow(U), right wrist to left elbow(V). ‘θ’ is the angle between the two ends of the limb.   
     
   *cos* θ= (*U*⋅*V*) / |U| |V|
5. **Table

   Description automatically generatedJudgement of the swing:** the report had a simple point system to judge how good the swing was. The table below shows how they calculated the scores.   
     
   A mobile phone was used that captured video at 60 fps at 1080p resolution.

To see if a high score led to success on the pitch, the distance the ball travelled from the home base was calculated. There was a positive correlation between the number of points earned and the distance the ball travelled from the home base. However, this does not consider the position of the bat or the contact with the ball itself, so it might not be an accurate method to test for success.

**8. Synthetic image translation for football players pose estimation**

**-**has solution for low quality video input but not directly related to our needs

**9**.[**Body Pose Estimation Integrated With Notational Analysis: A New Approach to Analyze Penalty Kicks Strategy in Elite Football**](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8964455/)

Used OpenPose.

34 videos

* Mean confidence score 0.80 +- 0.14
* Retest reliability 0.976 +- 0.03
* 25 biometric body part estimations
* Output is a 25 x 3 vector, first 2 columns are x-y coordinates, final is confidence score
* Two frames analyzed, run up and the kick

Target variables:

The penalty taker: nonkick foot orientation, hips, and shoulders

The goalkeeper: anticipation movement (keeper movement pre kick and during kick), right foot, left foot