FENCINGESTIMATE: A 3D GAME-BASED INTERACTIVE DRIVING SIMULATION TRAINING SYSTEM USING ARTIFICIAL INTELLIGENCE AND COMPUTER VISION

Weixuan Lei[1] and Yu Sun[2]

[1]Woodbridge High School, Irvine, CA 92604
[2]California State Polytechnic University, Pomona, CA, 91768

ABSTRACT

Training for fencing during the pandemic has changed from what it was beforehand. Students have been taking lessons online, and instead of fencing with peers, students now train by watching and analyzing fencing videos. Learning fencing from watching videos of world class fencers is an effective way of learning. However, sabre fencing is so fast that many inexperienced fencers are unable to capture the important information by watching short clips. Therefore, they are unable to learn techniques such as sabre fencing just from watching videos. This paper traces the development of an application that can utilize computer vision and pose estimation to analyze fencing video clips and output accurate scored points as well as the techniques used within the given clips. We applied our application to help less experienced fencers improve their ability to recognize points and conduct qualitative evaluations of different fencing techniques.

KEYWORDS

3D simulation, Computer vision, Artificial intelligence, Fencing lessons.

1. INTRODUCTION

Fencing, like most sports, relies on in-person experience and training. [2, 3, 4] During the pandemic, online learning has been recognized by the vast majority as a more viable option. [1] However, fencing is hard to understand without a solid foundation, and this is one of the reasons why it has had limited access for the vast majority of people. [11] This is especially true for sabre, which is the fastest of the three disciplines of fencing. Even less experienced fencers who fence using sabre themselves struggle to understand clips of world-class fencers, due to its high speed. However, there are many valuable things fencers can learn from such videos, especially during the pandemic, since this type of learning is not as restricted as in-person learning in many places. Learning from videos of world class fencers is useful when fencers have to train at home for whatever reason. This application aims to aid less experienced fencers to learn from fencing videos, and to also make it easier for them to understand the sport through watching fencing tournaments.

Some of the techniques and systems that have been proposed are neural network deep learning. [5, 6] Deep learning is effective when it comes to identifying a point, however, this assumes that the techniques and strategies used by the fencers are not important, which is exactly the purpose

David C. Wyld et al. (Eds): SOFEA, CTCM, BIBC, SIPR, NCWC, CSEN, EDTECH - 2021
pp. 205-212, 2021. CS & IT - CSCP 2021 DOI: 10.5121/csit.2021.111617

of this program. Other techniques, such as using a box to go around the fencers to decide whether the fencers hit each other are also used. This is not ideal because it often results in wrong judgements regarding hitting. This method considers the distance between the fencers as a sign of the end of a point instead of identifying the blade and actual hit. Other proposals include background removal where fencers can be represented by a two-dimensional array. However, a practical problem with this method is that as long as there are any changes in the camera, such as zooming in and out, or changes in the video such as lighting or set up, the background becomes difficult to analyze. [12]

This program uses python as its coding language, and Google colab as IDE in order to use GPUs provided by Google to analyze the video. [7, 8] In order to judge fencing points, this program first processes videos by splitting them using CV2 to get frames from the video feed. These frames are then passed into the pose estimation algorithm in jpeg format to capture the fencers' motions while the machine learning algorithm researches a series of json files containing the positions of the key points of the people detected in the video. Key points include major joints such as shoulders, elbows, and knees, and also important body parts such as the head, hands, and feet. [9, 10] The positions of these key points are checked to confirm if the people detected are the two fencers since many times the algorithm detects people passing in the background or the judge. Frames with people other than the two fencers detected, or with less than two fencers detected are discarded. To analyze the fencers' techniques and points, the positions of the key points are processed with the algorithm we developed, which detects the distance between the fencers, the speed of the fencers, and the slope of the arms of the fencers. With this data, the program returns the technique used, as well as the result of the point, with either the fencer on the left or right earning the point.

In two application scenarios, we demonstrate how the above combination of techniques. First, we show the usefulness of our approach for pose estimation by running all fifteen test cases, which includes fifteen videos with the correct points and tactics being used in the videos. Out of the fifteen videos, all resulted in too much background noise when background removal was applied, and all fifteen worked with pose estimation, with some frames having errors. This can be solved and the solutions are discussed later in this paper. Second, we analyzed the usefulness of the approach of the two main algorithms used in this application: one involving the angle between the lower arm and the straight line from wrist to shoulder, the other utilizing the slope of upper arm and lower arm, when processing the tactics and points in a video. The result revealed a 60% accuracy in judging a point when the method involving the angle between the lower arm and the straight line from wrist to shoulder was used, and an 87% accuracy with the method using the slope of the upper arm and lower arm. Therefore the method utilizing the slopes of the upper and lower arms was deemed the best algorithm when judging points and tactics. This method proved to have an 80% accuracy rate when judging tactics in the given videos.

The rest of this paper is organized as follows: Section 2 provides details on the challenges that we encountered as we designed the program; Section 3 focuses on the details of our solutions corresponding to the challenges mentioned in Section 2; Section 4 provides details about the effectiveness of our solutions in Section 3, followed by related work in Section 5. Finally, Section 6 provides concluding remarks and points out suggestions for future work for this project.

2. CHALLENGES

In order to develop an application that can utilize computer vision and pose estimation to analyze fencing video clips and output accurate scored points as well as the techniques used within the given clips, a few challenges were identified as follows.

2.1. Challenge 1: Gaining inspiration from related works

The first challenge was to find inspiration for ways to approach the development of the app from related works. Most related works that were researched utilize deep learning and train neural networks to recognize a point in fencing. However, all related works that were discovered did not focus on the tactics being used in a point, but rather the result of a point. This was not useful for the development of this app since its primary purpose is to educate beginners or those who are interested in sabre fencing to help them understand and learn from sabre fencing videos. The related works focusing on machine learning to act as a judge could not provide much valuable data or insight for us, so we had to develop our own methods to recognize tactics used by fencers within fencing video clips.

2.2. Challenge 2: Finding the right method

Another challenge was finding the right method. Initially, we focused on the algorithm of background removal, which uses machine learning to differentiate the background from moving objects in a video. The algorithm outputs a two-dimensional array, with the background being 0s and the object being 1s for each frame. However, as we developed the program, it did not work as intended because in many fencing videos, the cameras used aren't stable enough for background removal since they sometimes zoom in and out. Also, the lighting used in the video can affect background removal. All of these factors make the results of background removal noisy, so we had to switch from background removal to pose estimation, which is an algorithm that identifies the key points, mainly joints, of people in a video such as elbows, wrists, hips, knees, as well as the head. Using pose estimation allowed us to capture the movements of the fencers more accurately.

2.3. Challenge 3: Processing data captured by pose estimation

Another challenge came from processing data captured by pose estimation. Before actually processing the data, we needed to remove frames that did not work with the following algorithm that processes the data. An example of this is the pose estimation capturing people other than the fencers, such as the judge or others passing in front of the camera. Having the wrong people detected creates conflict with the following algorithm, which is designed for only two people standing behind the *en garde* lines, which are the starting lines in fencing. The general solution was to limit the number of people that could be detected in the videos, as well as the frames that don't have fencers detected correctly, whether from missing key points or pose estimation error.

3. SOLUTION

The machine learning program takes in user input in video format and processes it by using computer vision to split the video up into frames. With the frames ready, the program passes them to the pose estimation algorithm. This is a machine learning algorithm that can recognize body positions in videos by pinpointing the positions of key points. Key Points include important joints such as elbows, shoulders, wrists, and also the head. The positions of every body part in each frame is stored in json files. All later algorithms are based on these json files. The json files are then passed into later algorithms for further analysis about fencers' movements. Before that, files are processed so that frames that will be used later all correctly recognize the fencers and their body parts. Therefore, frames with people other than the two fencers as well as those with any important body parts left unrecognized are eliminated. Any frames with swapped body parts (left and right) will be swapped back. The left and right of the fencers should always match the first correct frame. Then the json files will be processed with an algorithm developed in order to

judge the point and the technique being used to make the point. The algorithm includes judging the right of way of the fencers, checking which fencer is the first to move, movement speeds of each fencer, as well as the extension of each fencer's arms. With the right of way, the computer checks the movement of the fencers right before and after the touch, which is decided when two fencers are close together. If there is no change in right of way, meaning no valid or successful defense effort, the point goes to the fencer with right of way originally. Otherwise, the point is rewarded to the defender. The frontend of this application was built in the IntelliJ integrated development environment (IDE) using Google's open-source UI software development kit Flutter. Flutter uses the object-oriented programming language Dart. See Figure 1 for a schematic of FencingEstimate's functioning.

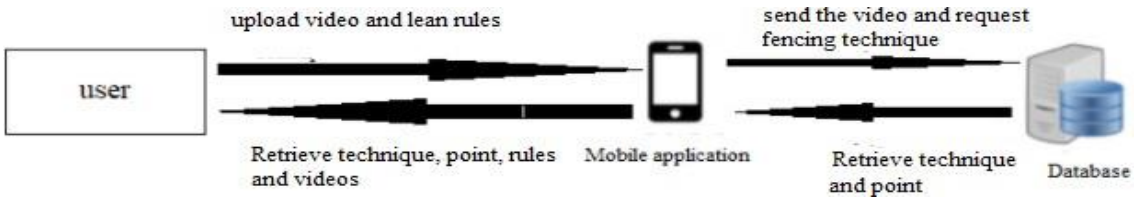

Figure 1. FencingEstimate UI schematic

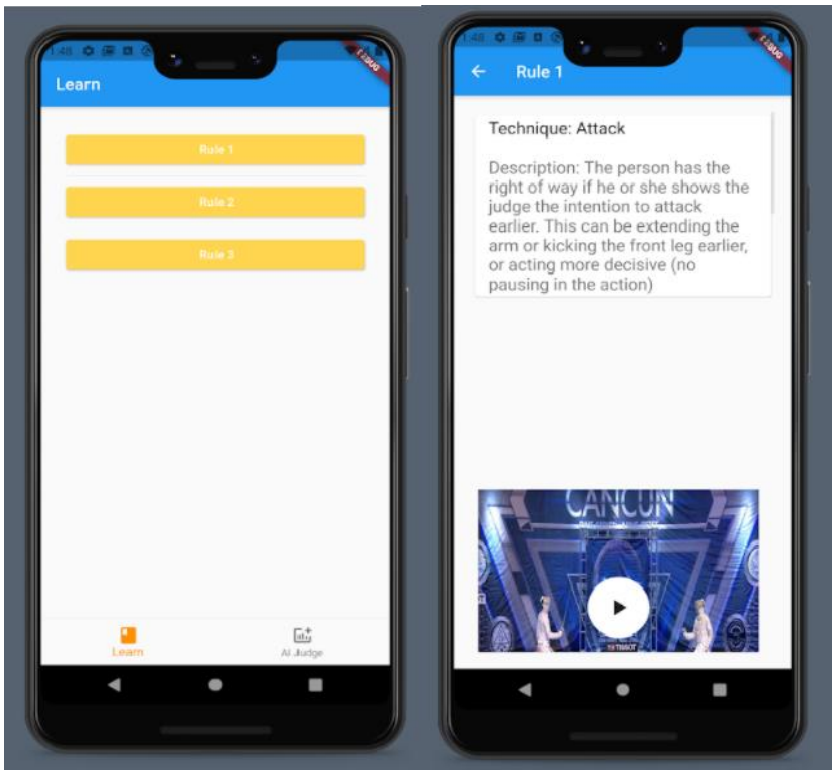

Figure 2. FencingEstimate Learning page and AI judge page (screenshots)

The AI judge page allows users to upload fencing videos, and the application sends them to the server. From there, the server passes them through computer vision and pose estimation to gain movement information about the fencers in json files. The program first checks which fencer gets the right of way by seeing which one moved their leg and arm first. Then, the program continuously checks the distance between the fencers, and only starts to evaluate the score and tactics when the distance between them is within the distance equivalent to the length of a sabre. In order to get the length of a sabre in pixels, and adjust to different resolutions, we take into

consideration the fact that the ratio of the distance between the fencers' front feet is set by the distance protocol, which is 4 meters, and the blade of the sabre, which is 0.88 meters long. Using these perimeters, we are able to calculate the distance between the fencers in pixels, then multiply that number by 0.22, the ratio of the distance between the fencers to the length of the sabre blade. In our application, since we are only given monocular information, it is difficult for the program to identify the position of the blades. Therefore, we set the default to read that as long as the fencers are closer together than the length of their blades, a hit should occur. See screenshots of FencingEstimate's Learning page and AI judge page in Figure 2 and AI Judge screenshots in Figure 3.

Then, the slopes of the fencers' arms are measured. If the slopes of both upper and lower arms are close to 0, it means that the fencer has extended his arm, and hence is currently attacking. If the other fencer is also attacking, then whoever has the right of way, or who attacks more aggressively gets the point. The program checks the decisiveness of the fencers by checking the landing of the front leg, as well as how stretched out their legs are, which is checked by the ratio between a fencers' height and the distance between their front and back legs as they attack; we use this ratio instead of distance because different fencers differ in height. The fencer is considered to be defending if the slopes of the upper and lower arm are closer to 1 and -1 than 0 and the ratio between the fencers' height and the distance between the front and back foot is greater than the other fencer, meaning the fencer is standing more upright and not having their legs more separated than the attacker. In this case, the system checks to see if the parry happens at the same time as the attack, if yes, then the defender gets the point if they extend their arm again for the riposte, otherwise it is considered a failed parry and the attacker still gets the point.

The program will then output the results of the score, and either the fencer on the left or right will get the point, with tactics output given alongside.

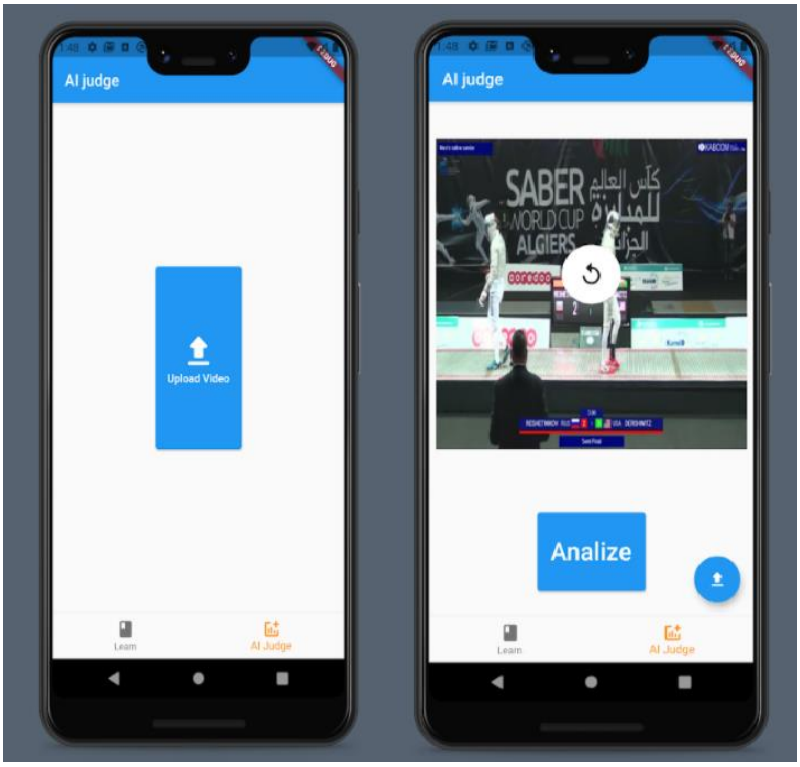

Figure 3. FencingEstimate AI Judge (screenshots)

4. EXPERIMENT

The two approaches compared here are the angle method, which utilizes the angle between the lower arm and the line drawn from wrist to shoulder, and the slope method, which utilizes slopes of the upper and lower arm (see Figure 4).

The results demonstrated a 60% accuracy in evaluating the point using the angle method, which does not include tactics, while the slope method had a rate of 87% accuracy. Therefore, the slope method was considered the best approach for this application. This method demonstrated an 80% rate of accuracy in correctly identifying tactics such as parry riposte, attack in preposition, attack no attack, and simultaneous within twelve video clips.

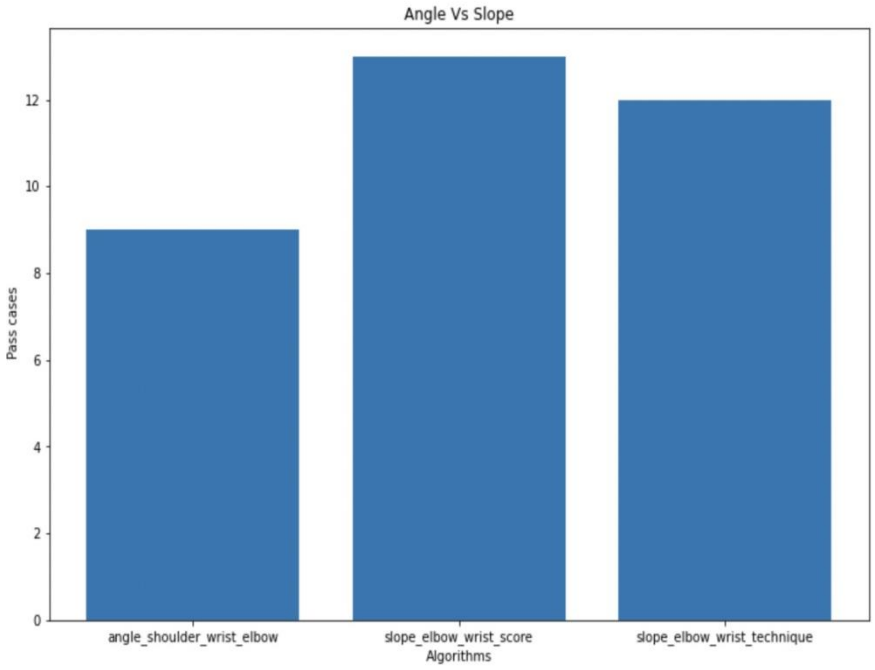

Figure 4. Pass cases: Angle vs. Slope

To evaluate the approach of the app, we use the fifteen test cases that include tactics such as parry riposte, attack in preposition, and attack no attack. All test cases include a short clip that only includes one point in sabre fencing to improve efficiency, as well as the correct score given and the tactics used in the clip. The two approaches of the algorithm were first used to decide the final algorithm of the application, and the one that generates a result with a higher accuracy is the approach for this application.

Out of the fifteen test cases, with the slope method, we sorted the data by technique, separating the test cases into Parry Riposte, Simultaneous, and Attack. As seen in Table 1, out of the seven parry riposte cases, six of them pass. All four of the simultaneous cases passed, and three out of four of the Attack cases passed.

Table 1. Test cases using the slope method

Technique	Parry Riposte	Simultaneous	Attack
Pass	6	4	3
Fail	1	0	1

The result of the experiment proved that our approach as a whole, including computer vision,pose estimation, as well as the later algorithms, works. With an accuracy rate of 87% for judging points, and 80% for judging techniques, the program still has room for improvement,but is a solid solution to the proposed problem.

5. RELATED WORK

Bridgeman, L., et al. presented a machine learning approach to apply multi-person pose estimation to sports. [13] Their methodology involves use of cameras from different angles to enable the detection of 3D skeletons from 2D information. In our application, we only have one camera angle, since most fencing competitions only have one camera in the center for the judge to review. In this case, our application was not intended to make 3D skeletons, since they are not needed for our algorithm to judge fencing points.

Fastovets, M., et al. presented another machine learning approach to estimate human posing from 2D, monocular camera views. [14] This work deals with moving camera angles with a trained model, while our work is designed for a stable camera angle. In both works, pose estimation is used to detect key points of bodies, which are the major joints including shoulders, elbows, hips, and important body parts such as the head, hands, or feet. The movement of a human body is captured via the recording of these key points, and most body parts can be represented by drawing lines connecting the key points.

Madelena, L., et al. presented a methodology for background subtraction on tracking moving objects. [15] This work involves the use of computer vision and other machine learning algorithms to achieve the detection of moving objects. This paper mentions some of the weaknesses of background subtraction, including the fact that it can be affected by certain lighting environments or colors. When we first developed our application, we tried background subtraction instead of pose estimation, and encountered the same issues mentioned in this paper.

6. CONCLUSION AND FUTURE WORK

In designing this application, we applied computer vision and pose estimation to judge fencing scores and tactics used by fencers in a given video, and used these algorithms on mobile applications, so that the upload of videos can be done in a convenient way. The mobileapplication features the instruction of basic rules, judging scoring via sample fencing videos, and the uploading of videos for analysis.

A limitation of this application is its limited number of cases. The current application was not designed to handle edge cases, such as when the right of way changes before blade contact, as the attacker purposely moves back to set up a defense position, or when the attacker makes a mistake and the right of way shifts to the defender. The current application also does not apply to situations where the camera is moving, and only applies to cases in which the points end inthe middle.

We plan to continue to add more features to the application to identify edge cases. We also plan to improve the limiting error output by pose estimation when the camera is in motion, add more features, and apply deep learning to detect a point and tactics when the camera is in motion.

REFERENCES

[1] Lightner, Bob. "Media Utilization in Fencing Instruction." *Journal of Physical Education and Recreation* 50.3 (1979): 64-65.

[2] Roi, Giulio S., and Diana Bianchedi. "The science of fencing." *Sports medicine* 38.6(2008): 465-481.

[3] Barth, Berndt, and Emil Beck, eds. *The complete guide to fencing*. Meyer & MeyerVerlag, 2007.

[4] Murgu, Andreia-Ileana, and Ralph Buschbacher. "Fencing." *Physical medicine and rehabilitation clinics of North America*17.3 (2006): 725-36.

[5] Liu, Weibo, et al. "A survey of deep neural network architectures and their applications." *Neurocomputing* 234 (2017): 11-26.

[6] Kriegeskorte, Nikolaus, and Tal Golan. "Neural network models and deep learning." *Current Biology* 29.7 (2019): R231-R236.

[7] Lutz, Mark. *Programming python.* " O'Reilly Media, Inc.", 2001.

[8] Raschka, Sebastian. *Python machine learning*. Packt publishing ltd, 2015.

[9] Haralick, Robert M., et al. "Pose estimation from corresponding point data." *IEEE Transactions on Systems, Man, and Cybernetics* 19.6 (1989): 1426-1446.

[10] Abidi, Mongi A., and T. Chandra. "A new efficient and direct solution for pose estimation using quadrangular targets: Algorithm and evaluation." *IEEE transactions onpattern analysis and machine intelligence* 17.5 (1995): 534-538.

[11] Roi, Giulio S., and Diana Bianchedi. "The science of fencing." *Sports medicine* 38.6 (2008): 465-481.

[12] Chen, Tony Lin-Wei, et al. "Biomechanics of fencing sport: A scoping review." *PloS one* 12.2 (2017): e0171578.

[13] Bridgeman, Lewis, et al. "Multi-person 3d pose estimation and tracking in sports." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops.* 2019.

[14] Fastovets, Mykyta, Jean-Yves Guillemaut, and Adrian Hilton. "Athlete pose estimation from monocular tv sports footage." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 2013.

[15] Maddalena, Lucia, and Alfredo Petrosino. "A self-organizing approach to background subtraction for visual surveillance applications." *IEEE Transactions on Image Processing* 17.7 (2008): 1168-1177.