Using Pose Estimation to Analyze Rock Climbing Technique

Jerry Qu
Stanford University
Stanford, California
jerryrqu@stanford.edu

Abstract

Rock climbing technique can be incredibly nuanced, where small differences in body position can make a huge difference. Furthermore, climbing technique can sometimes be daunting for beginners to learn. In this project, I created a climbing video analyzer using pose estimation. Using the output of ViTPose, a state-of-the-art model for pose estimation, with YOLOv8 being used for detection, my analyzer outputs climbing technique metrics related to the smoothness of a climber's center of mass trajectory as well as how often the climber's arms are bent. My analyzer is also able to identify and classify moments in a video when certain specific climbing techniques are used. Beginner climbers can use the generated reports to learn more about basic climbing techniques, while more advanced climbers can use these generated reports to compare attempts on a climb to see where there is room for improvement. The technique metrics are actually able to reflect differences in trajectory efficiency and arm flexion efficiency as intended, and the technique classifier achieves F1 Scores of 71-94% (depending on which technique) on the test set.

1. Introduction

Rock climbing is an intricate sport with various technical aspects that involve physics and geometry. One of the biggest challenges for newer climbers is learning techniques that conserve energy and minimize the use of arm muscles (and maximizing the use of legs and core muscles) in order to climb more efficiently. These techniques include but are not limited to skills like flagging and drop knees that keep the climber's center of mass in-line above the driving foot and the climber's hips closer to the wall (which puts more weight on the legs and less on the arms), efficient momentum generation and trajectories for dynamic movement, body positioning to minimize pulling with the arms, etc. Depending on the person, these techniques can take time to learn and get used to, and sometimes may even be unintuitive when first introduced to them.

Even after learning these techniques, climbers are still always in the process of refining them. At higher levels of climbing, even the smallest details in body, hand, or foot positioning can make a massive difference. Many climbers find it useful to take videos of their climbing to see what their body looks like from a third-person view, which can help them understand what's working and what needs to change. Comparing videos of the same climb can show important differences in technique [1] that can be useful to keep in mind; for example, a climber trying a particularly hard move could take videos of their attempts until they successfully get the move, and then see what's different about the successful attempt vs the failed attempts.

While the general strategy of newer climbers learning technique from more experienced climbers and climbers of different skill levels analyzing/comparing technique by watching videos of themselves climbing works quite well, it could be useful to get a computer's perspective. Not only would this automate the process, but a computer could also provide more quantitative feedback about miniscule but important details that are harder to see with the naked eye.

In this work, I create a climbing video analyzer that gives such quantitative feedback. More concretely, given a climbing video that is ideally around 60 seconds or less from a fixed camera angle (preferably head-on to the wall), the analyzer will output graphs of different joint angles vs time, identify any specific climbing techniques utilized at any moment (for now, the left and right versions of drop knees, high feet, forward flags, backflags, and inside flags), display a real-time de-noised center of mass (COM) trajectory, and output technique scores based on the trajectory efficiency as well as elbow flexion. To do this, I use ViTPose, a state-of-the-art model for pose estimation, trained with the COCO dataset, with YOLOv8 being used for detection. The keypoints output by ViTPose were used to perform all the anal-

[1] Throughout this paper, I will be using the word "techniques" in two ways. One way refers to specific climbing techniques such as drop knees, high feet, and flagging. The other way is more general and describes overall technique while climbing (which is an amalgmation of a variety of factors, including the specific techniques mentioned above, keeping hips close to the wall, footwork, body positioning, micro-adjustments in grip, etc.).

ysis above.

For now, this analyzer is expecting videos with resolution 1080px by 1920px and 30 fps (the typical videos taken with my iPhone 11 Pro Max). The accommodated fps can be changed easily, but a bit more work needs to be done to accommodate generating a video report for videos of a different resolution. As I mentioned above, the analyzer is most suited for shorter climbing videos, which would likely be from bouldering (rock climbing on shorter outdoor rock formations or artificial climbing gym walls) or sections of longer climbing routes.

2. Related Work

There have been a few previous attempts to apply machine learning and computer vision methods or other technology to rock climbing, from academic papers to independent projects. Here, I review some of them.

2.1. Academic Papers

One work also implemented the idea of generating a post-climb report based on a video [7]. They used object detection methods such as YOLO to detect climbing holds, as well as Mediapipe Pose estimation to estimate the pose of the climber. Their report features included the percent completion of a climb, the validity of gym climbs (i.e., making sure the climber is using the same-colored holds for a given gym climb, as is the rule in most climbing gyms), distance the center of the climber moved, number of moves taken, and total time elapsed. Another paper applied pose estimation and ML classifiers to videos of climbing competitions to predict the names of the professional climbers in the videos as well as whether the climber would succeed in completing the climb by analyzing the first 150 frames from the video [11]. While these two works show the power of pose estimation and machine learning to analyze climbing videos, I wanted to see if computer vision could be used to perform analysis more related to climbing technique, aspects of which are harder to ascertain just from a human perspective.

Other works have explored the use of sensors and motion capture to analyze climbing. [11] explores different sensors, motion capture approaches, and motion analysis algorithms that can be applied to climbing. However, I wanted to explore the analysis of climbing from videos without the use of additional sensors. They also explore the use of pose estimation, and they also survey different motion analysis algorithms that have been implemented, including ones that output a metric of agility (a combination of speed and acceleration of the CoM), identify dynamic movements, or provide offline route/beta-planning (beta refers to the sequence of steps and techniques needed to complete a climb). A few years later, those same authors created a system that utilizes the LiDAR of a fourth-generation iPad Pro to convert the climber's 2D skeleton into 3D joints, and then uses this to detect movement errors that are common for novice climbers [5].

Others have introduced the application of analyzing 2D skeletal data on speed climbing [14] [8], a climbing discipline with a single standardized route where the goal is to reach the top as quickly as possible. This is a useful application, since the standardized route allows for easy comparison to show how small changes in trajectory and movement can lead to faster or slower times. However, my system is more applicable to bouldering or sections of longer routes that aren't speed climbing (and thus a wider variety of climbs).

There have also been works using pose estimation to temporally segment a climber's movements into smaller sections [4], or suggest an "interpolated" movement calculated from a beginner climber video and an expert climber video that could as a stepping stone for beginner climbers to learn more advanced technique [12]. Both of these are very unique applications of pose estimation for rock climbing technique analysis.

2.2. Commercial Application

Belay AI, an app that is still in beta [3], is a project that is most closely related to mine. It also aims to generate a climbing report from a climbing video taken from a smartphone/tablet camera. Based on their website and video demos, their app provides a lot of information about trajectory, joint angles and velocities, technique suggestions, etc. I aimed to implement similar features myself (except for the technique suggestions), but also explore real-time technique detection and metrics for climbing efficiency.

2.3. Independent Projects

I've also seen a couple of posts on a discussion forum where people have shared their own projects applying computer vision to climbing videos. One used pose estimation to track elbow flexion over time [2]. One of the criticisms that was brought up in the forum was the lack of quantification of "good" technique; while elbow flexion is an important part of climbing, it alone is not a measure of technique. While the app's actual website indicates the desire to add these features, it doesn't seem to be publicly available yet.

Another app posted on the discussion forum that is publicly available is AscentAI [1], which uses pose estimation to track CoM trajectory as well as other annotations.

3. Methods

3.1. YOLOv8

YOLO, or "You Only Look Once," is a real-time end-to-end object detection model that can accomplish detection with a single pass of the network [13]. It outputs a bounding box, a box confidence score, and a class confidence score. Unlike previous approaches, which either used sliding windows followed by a classifier that had to be utilized for each window or broke the detection into two steps (region proposals and then classifying), YOLO unified the steps by detecting all bounding boxes simultaneously; it does this by partitioning an image into a grid of cells, and then, for each cell, predicting some number of bounding boxes with their center in that cell and their scores as well as the predicted class scores for that particular cell. From both the bounding boxes and confidence scores as well as the class probability map over the grid cells, the model is able to output bounding boxes of objects of an image as well as their identified classes [10]. The model uses 24 convolutional layers (with max pooling) followed by two fully-connected layers (with dropout) [13].

Between the first version of YOLO and YOLOv8, there have been many improvements and additions, including a more streamlined architecture, batch normalization, accommodation of different input sizes, spatial pyramid pooling, multi-scale prediction, different loss functions, and the combination of high-level features with contextual information [13]. As a result, YOLOv8 is faster and more accurate than its predecessors.

3.2. ViTPose

Once YOLOv8 detects a human in an image, a pose estimation model is needed to extract keypoints. ViTPose is a human pose estimation model that utilizes plain and non-hierarchical vision transformers to extract features for a person instance and a decoder for pose estimation, which outputs key body joints on the person. Because it utilizes transformers, it is highly scalable, flexible, and parallelizable, and performs very well compared to other models on the COCO Keypoint Detection benchmark [15].

3.3. Keypoints, Center of Mass, and Joint Angles

I use ViT-B (the baseline model size) trained on the COCO dataset implemented by [6], utilizing YOLOv8-s (the small model size) for object detection. To extract keypoints, I apply pose estimation to each frame to obtain an array of keypoints over time.

To estimate the center of mass (COM) of the climber, I use the mean of the four back keypoint positions. While this isn't exact for human bodies and the COM position can change depending on the pose, this approximation is good enough and it is very similar to what professional climber Lynn Hill uses to analyze the trajectory of climbers in her climbing course [9].

To find the angles of several joints on the body (right and left elbows, knees, and hips), I take the corresponding three keypoints that form a given angle, find their associated vectors $u = (u_1, u_2)$ and $v = (v_1, v_2)$, and use the following formula to determine the clockwise angle between 0 and 360 degrees:

$$(360 + \operatorname{atan2}(v_2, v_1) - \operatorname{atan2}(u_2, u_1)) \pmod{360}$$

where the $\operatorname{atan2}$ outputs degree values. Then, depending on the joint, I further modified the angle values so that elbow angles would range between 0 and 180 (fully bent and fully extended, respectively), knee angles would range between -180 and 180 (where negative values represent when a knee is pointing inwards while positive values represent when a knee is pointing outwards), and hip angles would range between 0 and 360 (where 90 represents the angle while standing and 180 represents the angle if one was in a full middle splits; hip angles can easily go above 180 when raising one's knee above their waist). Graphs of these angles vs time are displayed on the report, with a moving vertical red line indicating the passage of time through the graphs.

Occasionally, pose estimation will fail for certain frames and keypoint data will be missing. For the missing frames, I linearly interpolate the COM position as well as the joint angles, which is a good enough estimate since these gaps are usually sparse and short.

3.4. Technique Classifier

The techniques that I considered in this work are drop knees (involving internal hip rotation, "dropping" one's knee), high feet (stepping up relatively high with one's foot), forward flags (extending one leg to the side to act as a counterbalance when reaching in the other direction), backflags (another type of flagging used for counterbalance, but using the same-side leg as the direction one is reaching to flag, extending the leg behind the other leg), and inside flags (similar to backflags, except the flagging leg is extended in front of the other leg). I considered the left and right-sided versions of each of these techniques. Examples of each of these techniques are in the Appendix in Figures 4-8. High feet are often performed if there a limited foothold options lower down, while the other techniques mentioned are used to keep the body balanced and hips close to the wall while reaching for the next hold, which saves energy. While there are several other climbing techniques, these cover most of the most basic techniques that newer climbers are introduced to.

Due to a lack of datasets containing climbing images/videos labelled with specific techniques and the time it would take to create such a dataset, I did not use ML to classify techniques. Instead, I opted for a different approach to classify techniques that used relative keypoint locations as well as joint angles. In the vast majority of cases, each of the the techniques mentioned above can be described by a set of conditions on certain joint angles or keypoint relative

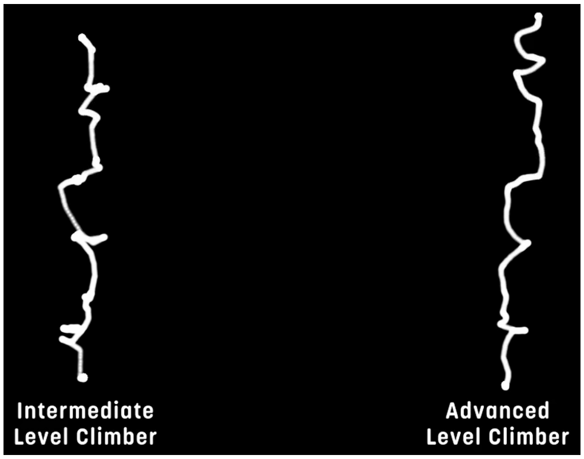


Figure 1: Screenshot from introduction video to Lynn Hill's climbing course, comparing the COM trajectories of climbers of different skill levels on the same climb.

positions, which I determined by analyzing several example clips of each technique. As an example, left drop knees corresponded to when the following conditions were satisified: left knee angle was negative and bent by at least a certain amount (I chose -160 degrees as the threshold) and the left ankle was further to the left than the middle of the two hip keypoints (the "mid-hip"). Another example is that left high feet corresponded to when the vertical distance between the left foot and right foot was at least as large as some threshold (I chose 0.8) multiplied by the vertical distance between the mid hip and right foot. I derived such conditions for each of the 5 techniques, and mirrored them to deal with the left and right cases.

I also only considered frames where the confidence scores of all lower body points output by the pose estimation model were greater than 0.5, since only lower body keypoints are needed to detect the 5 techniques mentioned. I also displayed a "Technique Detection Confidence" score that corresponded to the mean of the confidence scores of the lower body points.

3.5. Trajectory Efficiency

In the introduction to professional climber Lynn Hill's course on climbing, Hill describes how the COM trajectory of an advanced climber is smoother than that of a more novice climber, indicating more efficient movement [9], as can be seen in Figure 1. A smoother trajectory usually indicates fewer wasted movements and thus a higher energy efficiency while climbing. She estimates the location of the COM to be some fixed point on the climber's back at all times. I created a metric that quantified the efficiency of a climber's COM trajectory based on this idea.

I noticed that there was some noise introduced by the keypoint locations from the pose estimation model, artificially adding some "jerkiness" to the COM trajectory. So, before analyzing the trajectory of the COM, I de-noised its trajectory using filtering. By applying a digital filter once forward and once backward to the array of COM points, I effectively used a combined filter with zero phase to denoise the trajectory. I made sure to not use a filter of too high of an order so that the shakiness of the COM trajectory that came from the climbers' movements was still preserved. I then displayed this de-noised trajectory on the output report video.

I then considered a few ways to create a metric for the "jerkiness" of the de-noised trajectory. One of the main ways I tried was calculating the power spectrum from the array of COM points and finding the contribution of the higher frequencies. However, I found that this still penalized sharp changes in direction that were inherent in the climb itself, rather than from the climber's movements. The method I decided on using was applying a higher order filter to the original COM trajectory to produce a smoothed trajectory that was both de-noised and also removed the shakiness from the climbers' movements. Then, the trajectory efficiency metric $e_{trajectory}$ was based on the difference between the de-noised trajectory and the smoothed trajectory, where the smoothed trajectory represented an "ideal" trajectory with fewer wasted movements:

$$\Delta = \sum_{i=1}^{N} \|COM_{denoised,i} - COM_{smoothed,i}\|$$

$$e_{trajectory} = 1 - \left(\frac{\Delta}{L}\right)^{\alpha}$$

where N is the number of frames, Δ is a measure of the difference between the denoised and smoothed trajectory, $COM_{denoised,i}$ is the COM point of the ith frame of the denoised trajectory (analagous definition for $COM_{smoothed,i}$), L is the total length of the smoothed path, and α is a constant that controls the scaling of the scoring system (I ended up choosing $\alpha = 1.15$).

It is important to note that the calculation of this metric is the reason why I require videos from a fixed, non-moving camera angle (the other features of this analyzer don't require this). If the camera was moving, the trajectory generated relative to the confines of the camera frame wouldn't correspond as closely to the actual trajectory of the climber in projected 2D space because there would be relative motion between the reference frames.

3.6. Straight Arms Efficiency

A very common tip newer climbers are often told is "Keep your arms straight!" The idea of having good climbing technique is conserving energy and having one's arms do as little of the work as possible. It is almost always more efficient to have your core, legs, and skeletal structure do as much of the work of climbing as possible: the core and leg muscles tire out much more slowly while climbing, and relying on one's skeletal structure to support one's weight takes very little energy. Having straight arms leverages the skeletal structure: rather than pulling oneself up the wall with arm muscles every move, good body positioning can often allow one to keep their arm mostly straight and simply reach for the next hold. In the latter case, the arm's skeletal structure is still supporting the body, but the arm muscles are working much less hard.

This tip is very useful for easier climbs with good holds as well as overhanging climbs (climbs where the wall is angled backwards, requiring the arms to support more of the weight). However, as one progresses in climbing, they will run into situations where the "straight arms" tip isn't always the most helpful: sometimes it's necessary to have bent arms when dealing with different body positions or worse holds. Still, it is still good advice generally, especially for beginners, and excessively bent arms are generally an indicator of worse technique.

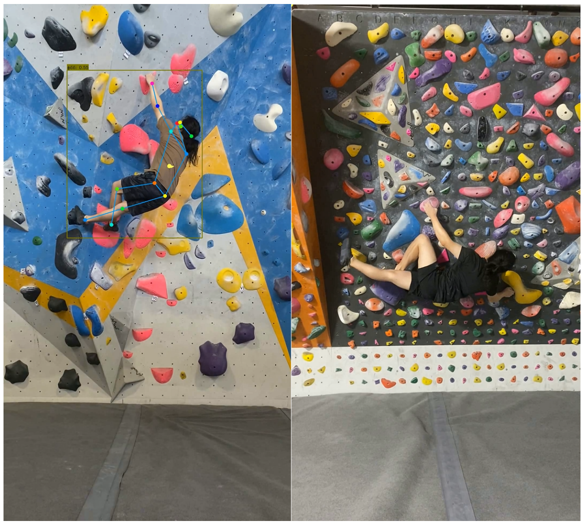


Figure 2: Comparison of two different frames showing variations of a right backflag. Pose estimation succeeded on the left, but failed on the right.

I created a metric based on this tip that quantifies how straight or close-to-straight a climber's arms are during a climb. For a given frame i, if a climber's elbow angles are RE_i and LE_i (right and left elbow angles, respectively), the straight arm score $e_{arm,i}$ for that frame is given by

$$e_{arm,i} = \frac{1}{2}\left(\frac{1}{1+e^{-(\frac{LE_i}{20}-4.5)}} + \frac{1}{1+e^{-(\frac{RE_i}{20}-4.5)}}\right)$$

The final straight arm score e_{arm} is given by the following formula:

$$a = \frac{1}{N}\sum_{i=1}^{N} e_{arm,i}$$

$$e_{arm} = \frac{1}{1+e^{-(7a-2.75)}}$$

These formulas were chosen in a way to try to give a reasonable score between 0 and 1 where neither slightly bent arms nor non-excessive bent arms were penalized too much.

3.7. Gathering Data

To test my analyzer, I took climbing videos of myself and some of my fellow climber friends. I tried to take all videos with an angle head-on to the wall. When testing the trajectory efficiency metric, I used videos with a fixed camera angle as well.

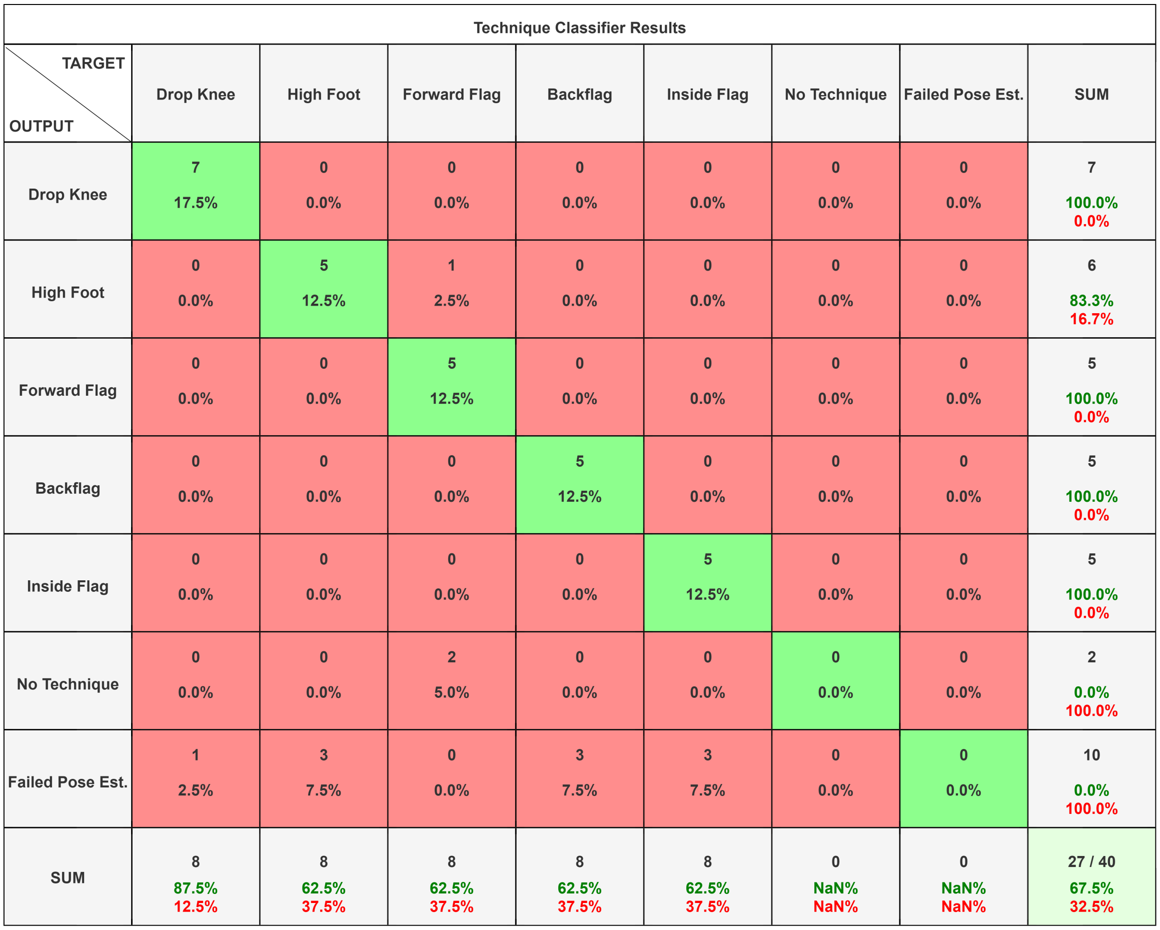
To specifically test the technique classifier, I took 40 short clips to form a test set, with 8 videos for each of the 5 techniques mentioned above. For each technique, the 8 videos were split so that 4 of them were clips of the "left" version of the technique, while the other 4 were of the "right" version. I manually labelled each of the clips (e.g., "right drop knee, left high foot, right backflag, etc."). It would've been nice to gather a larger test set, but I was limited by time.

To specifically test the technique scores, I found 2 bouldering problems (when bouldering, we refer to climbs as "problems") and climbed them each 3 times, taking videos of each. One time, I climbed intentionally with very bent arms the entire time, which made my arms noticeably more tired. The second time, I climbed intentionally with more jerky and wasted movements, which made me somewhat more tired overall. The third time, I tried to climb with the best overall technique I could, like I would normally try to climb a problem (this was always less tiring than the first two ways). I then compared the efficiency scores output for each attempt to see if the scores I calculated were actually a reflection of how I climbed. It's obviously a bit harder to test this as objectively as the technique classifier, since it's harder to exactly quantify the "effort" exerted, the "jerkiness" of a trajectory, and overall how bent my arms are throughout a video, but this comparison still gave a general indication of the usefulness of these scores. Once again, it would've been nice to test this on more climbs, but I was limited by time.

4. Results and Discussion

4.1. Sample Climbing Report

Here is a link to a video showing a sample climbing report generated by my project.



Technique Classifier Results

TARGET OUTPUT	Drop Knee	High Foot	Forward Flag	Backflag	Inside Flag	No Technique	Failed Pose Est.	SUM
Drop Knee	7 17.5%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	7 100.0% 0.0%
High Foot	0 0.0%	5 12.5%	1 2.5%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	6 83.3% 16.7%
Forward Flag	0 0.0%	0 0.0%	5 12.5%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	5 100.0% 0.0%
Backflag	0 0.0%	0 0.0%	0 0.0%	5 12.5%	0 0.0%	0 0.0%	0 0.0%	5 100.0% 0.0%
Inside Flag	0 0.0%	0 0.0%	0 0.0%	0 0.0%	5 12.5%	0 0.0%	0 0.0%	5 100.0% 0.0%
No Technique	0 0.0%	0 0.0%	2 5.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	2 0.0% 100.0%
Failed Pose Est.	1 2.5%	3 7.5%	0 0.0%	3 7.5%	3 7.5%	0 0.0%	0 0.0%	10 0.0% 100.0%
SUM	8 87.5% 12.5%	8 62.5% 37.5%	8 62.5% 37.5%	8 62.5% 37.5%	8 62.5% 37.5%	0 NaN% NaN%	0 NaN% NaN%	27 / 40 67.5% 32.5%

Figure 3: Confusion matrix for technique classifier.

https://youtu.be/KN7bWMGh4iU

The input video was a video of me climbing a boulder problem on the Kilter Board. For the graphs of joint angles, LE, RE, LK, RK, LH, and RH represent left elbow, right elbow, left knee, right knee, left hip, and right hip, respectively.

4.2. Pose Estimation

While pose estimation worked decently well for most climbing videos, there were also frames where it failed. For some frames, certain keypoints would be missing, and sometimes a human wouldn't even be detected. Figure 2 shows a comparison of a frame where pose estimation succeeded and pose estimation failed; the frames show different variations of the right backflag, and my analyzer was only able to successfully detect a human for the left frame and correctly identify the technique.

After testing my analyzer on many different videos, I found a few factors that contributed to the pose estimation failing in some way. One factor was fully/partially obstructed limbs: certain body positions and techniques (like the high foot) combined with a given camera angle can make a limb hidden from view. Another factor was more "extreme" body positions: very "deep" backflags (such as the one shown on the right in Figure 2 and other positions were very hard for pose estimation to capture. One more factor was the collection of colors in the video: it seemed like certain combinations of colors of the wall, climbing holds on the wall, and climber's clothes made it more likely for pose estimation to fail.

4.3. Technique Classifier

I ran all 40 clips in the test set through my analyzer and looked at what the climbing report generated. In addition to the 5 technique classes, I also considered 2 additional classes representing some sort of failure: a "no technique detected" class where pose estimation succeeded but none of the 5 techniques was detected, and a "failed pose estimation" class where pose estimation failed (this includes cases where a human was detected but there were lower body keypoints that were missing).

It's important to note that there should actually be 10

	Precision	Recall	F1 Score
Drop Knee	1.000	0.875	0.933
High Foot	0.833	0.625	0.714
Forward Flag	1.000	0.625	0.769
Backflag	1.000	0.625	0.769
Inside Flag	1.000	0.625	0.769

Table 1: Performance metrics of technique classifier on each technique.

technique classes, since each of the 5 techniques has a left and right version. However, I found that the classifier never made a mistake between left and right; even if the classifier misclassified one technique as another, it identified whether it was "left" or "right" correctly. So, to save space, I combined the "left" and "right" classes together for each technique.

Figures 4-8 in the appendix show examples of each of the 5 techniques being successfully classified. The confusion matrix for the technique classifier on the test set is shown in Figure 3. The performance metrics are shown in table 1. For all 5 techniques, precision was significantly higher than recall. In fact, all techniques were classified with a precision of 1 except for high feet, which had a precision of 0.833. Meanwhile, all techniques were classified with a recall of 0.625 except for drop knees, which had a recall of 0.875. The lower recall scores were almost always due to failed pose estimation. The exception to this is forward flags, where the lower recall was due to one misclassification and two missing classifications. However, the very high precisions show that when pose estimation succeeds, my classifier is very good at classifying techniques.

Overall, the drop knee was the easiest technique to classify, with a precision of 1, recall of 0.875, and the highest F1 score of 0.933. This is likely due to the fact that the conditions for a drop knee are relatively simple, as well as the fact that drop knees rarely result in obstructed limbs. The high foot was the hardest technique to classify, with a precision of 0.833, recall of 0.625, and the lowest F1 score of 0.714. This is likely due to the fact that high feet often result in at least partially obstructed limbs. Despite the three types of flagging having relatively complicated conditions, they showed a high precision. Their low recall can partially be attributed to difficult-to-detect body positions (e.g. the deep backflag).

4.4. Efficiency Scores

As mentioned above, I tested the scores by climbing 2 boulder problems in 3 different ways each: one with bent arms, one with jerky and wasted movements, and one that was an improvement on the previous two. Figures 9-11 in the appendix show snapshots of the generated video reports of the climb 1 attempts (bent, jerky, and improved, respectively). Figures 12-14 in the appendix show snapshots of the generated video reports of the climb 2 attempts (bent, jerky, and improved, respectively). Table 2 shows the scores my analyzer output for each of these attempts.

Climb 1	$e_{trajectory}$	e_{arm}
Bent	0.69	0.47
Jerky	0.68	0.78
Improved	0.74	0.77
Climb 2	$e_{trajectory}$	e_{arm}
Bent	0.73	0.52
Jerky	0.6	0.78
Improved	0.71	0.86

Table 2: Comparison of technique scores for 2 different climbs, each climbed in 3 different ways.

As the table shows, the scores work as expected, at least for these two climbs. The "bent" arm attempts for both climbs resulted in lower scores for e_{arm} than the other two attempts, while the "jerky" attempts for both climbs resulted in lower scores for $e_{trajectory}$ than the other two attempts. The "improved" attempts showed higher score for both e_{arm} and $e_{trajectory}$ than the other two attempts, with the exception of $e_{trajectory}$ being very slightly higher in the "bent" arm version of climb 2 than in the "improved" version. This isn't too much of an issue; it could very well be that when my arms were bent, I also happened to move very slightly more stably. Importantly, there was a large difference in e_{arm} scores for those two attempts.

An important thing to note about these scores is the difficulty of comparing them across different climbs. Even though the scores are normalized by the length of the climbing videos, different climbs can require very different body positions and lead to very different trajectories. For some boulder problems, it may be very hard to keep one's arms straight often, and for other boulder problems, it may be very hard to prevent movements that may be seen as "jerky". So, I would advise against comparing these scores between different climbs (a score of 0.5 may be "good" for one climb and "bad" for another). However, these scores are useful for comparing different attempts of the same climb (even attempts done by different climbers). The absolute values of the scores may not say much, but comparing the relative values of the scores between attempts can give climbers insight into how technique is utilized between attempts.

4.5. Time To Analyze Videos

Using YOLOv8-s for object detection and ViT-B for pose estimation, keypoint extraction for a climbing video takes approximately 0.14 seconds per frame with Google Colab's T4 GPU. The generation of the new video with the technique report takes approximately 0.13 seconds per frame using Google Colab's CPU. So, the keypoint extraction for a 30 second bouldering video at 1080x1920 resolution and 30 fps would take around 126 seconds, while the generation of the new video would take around 117 seconds, which means that the total analysis time would be around 243 seconds. While this isn't overly unreasonable, it would be better to speed this up for practical use; ideally it would take 1 or 2 minutes total.

5. Conclusion and Future Work

5.1. Conclusion

In this work, I created a climbing video analyzer that generates a video report with information about the COM trajectory, joint angles, specific climbing techniques detected at different moments, and scores that quantify the efficiency of the COM trajectory and elbow flexion over time. The technique classifier has high precisions (most 1.0, although 0.833 for high feet) but somewhat lower recalls, mostly due to occasional failures in pose estimation. The efficiency scores based on the COM trajectory and elbow flexion also work as expected based on the analysis of six attempts over two test climbs.

5.2. Future Work

I was limited by the time I had this quarter, but I plan on continuing this project after this quarter ends too.

I would give my climbing analyzer more climbing videos of varying levels of technique (different amounts of wasted movements and bent arms) to further refine the trajectory efficiency and the straight arm efficiency scoring systems, perhaps exploring more alternative ways to calculate the trajectory efficiency. I would also want to explore more ways of quantifying climbing technique that can somehow be captured using computer vision.

Due to the difficulty of obtaining a labelled dataset of climbing technique images/videos, I did not implement the technique classifier using machine learning methods. However, with more time, I will explore ways to build up such a dataset and use machine learning methods to do technique classification. While the if statements with several conditions worked decently well for classifying techniques, I think machine learning will work better in the long run with a large and varied enough dataset; a given climbing technique can have so many visual variations that it is hard to capture all the nuances with just a set of conditions like I tried to do. I'd also want to include more types of techniques that can be classified if possible (heel hooks, toe hooks, hand jams, foot jams, pogos, dynos, etc.), many of which would be difficult to identify using pose estimation keypoints alone.

I also want to find ways to optimize the process to cut down the time it takes to analyze videos, maybe by around half. Furthermore, I'd like to make my analyzer more flexible and robust: improving the object detection and pose estimation by looking more into what causes issues in the detection of a body, dealing with a slightly moving camera, dealing with not exactly head-on camera angles (maybe even a side-view angle; another important aspect of climbing technique is the distance of a climber's hips to the wall), etc.

A brief summary of other features I'm thinking of implementing: velocity and acceleration graphs over time; climbing hold detection for gym climbs as done in [7], which would also make it easier to identify whether a hand or foot is actually in contact with a climbing hold (I'd imagine this would be much more difficult to tell for outdoor climbs on real rock); a "beta suggester," which could give suggestions of the sequence of steps and techniques that could be used to complete a climb (which, as mentioned earlier, is called "beta" in the climbing community), etc. It would be nice to eventually turn this into a functional app that climbers can use to analyze their technique and that beginner climbers can use to learn more about technique.

6. Acknowledgements

I'd like to thank my friend and fellow climber Nathan Cuevas for helping me take climbing videos to analyze and providing useful suggestions about how to evaluate climbing technique. I'd also like to thank my friend Caitlyn Vergara for also helping me take climbing videos to analyze.

References

[1] AscentAI. Available at https://play.google.com/store/apps/details?id=com.jonasdeuchler.ascendai&hl=en&gl=US&pli=1.

[2] Climbalyzer. Available at https://climbalyzer.com/.

[3] Belay AI, 2024. Available at https://belay.ai/.

[4] R. Beltrán B., J. Richter, and U. Heinkel. Automated human movement segmentation by means of human pose estimation in rgb-d videos for climbing motion analysis. *VISIGRAPP 2022*, 5:366–373, 2022.

[5] R. Beltrán B., J. Richter, G. Köstermeyer, and U. Heinkel. Climbing technique evaluation by means of skeleton video stream analysis. *Sensors*, 23(8216), Oct 2023.

[6] A. D. and K. Lóránt. easy_vitpose, Apr 2024.

[7] S. Ekaireb, M. A. Khan, P. Pathuri, P. H. Bhatia, R. Sharma, and N. Manjunath-Murkal. Computer Vision Based Indoor Rock Climbing Analysis, Jun 2022.

[8] P. Elias, V. Skvarlova, and P. Zezula. Speed21: Speed climbing motion dataset. In *Proceedings of the 4th International Workshop on Multimedia Content Analysis in Sports*, MMSports'21, page 43–50, New York, NY, USA, 2021. Association for Computing Machinery.

[9] L. Hill, Feb 2023. Available at https://www.youtube.com/watch?v=XDuJqHTmSD4.

[10] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. *CVPR*, Jun 2015.

[11] J. Richter, R. Beltrán, G. Köstermeyer, and U. Heinkel. Human climbing and Bouldering Motion Analysis: A survey on sensors, motion capture, analysis algorithms, recent advances and applications. *VISAPP*, 5:751–758, 2020.

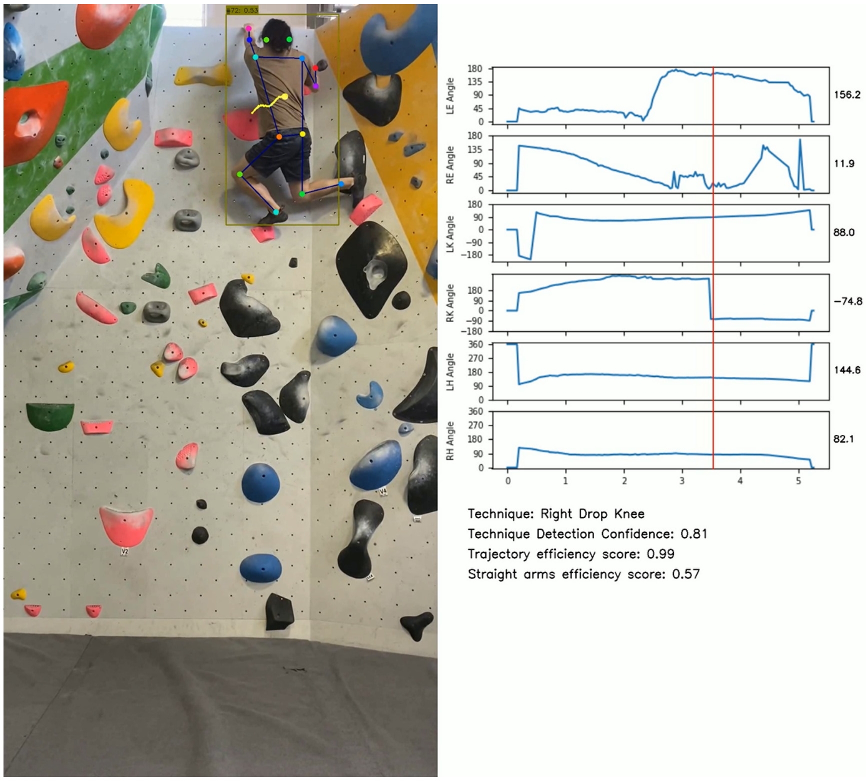
[12] K. Shiro, K. Egawa, T. Miyaki, and J. Rekimoto. Interposer: Visualizing interpolated movements for bouldering training. In *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI EA '19, page 1–6, New York, NY, USA, 2019. Association for Computing Machinery.

[13] J. Terven and D. Cordova-Esparza. A comprehensive review of yolo architectures in computer vision: From yolov1 to yolov8 and yolo-nas. *arXiv*, Jan 2024.

[14] J. XU, K. TASAKA, and M. YAMAGUCHI. [invited paper] fast and accurate whole-body pose estimation in the wild and its applications. *ITE Transactions on Media Technology and Applications*, 9(1):63–70, 2021.

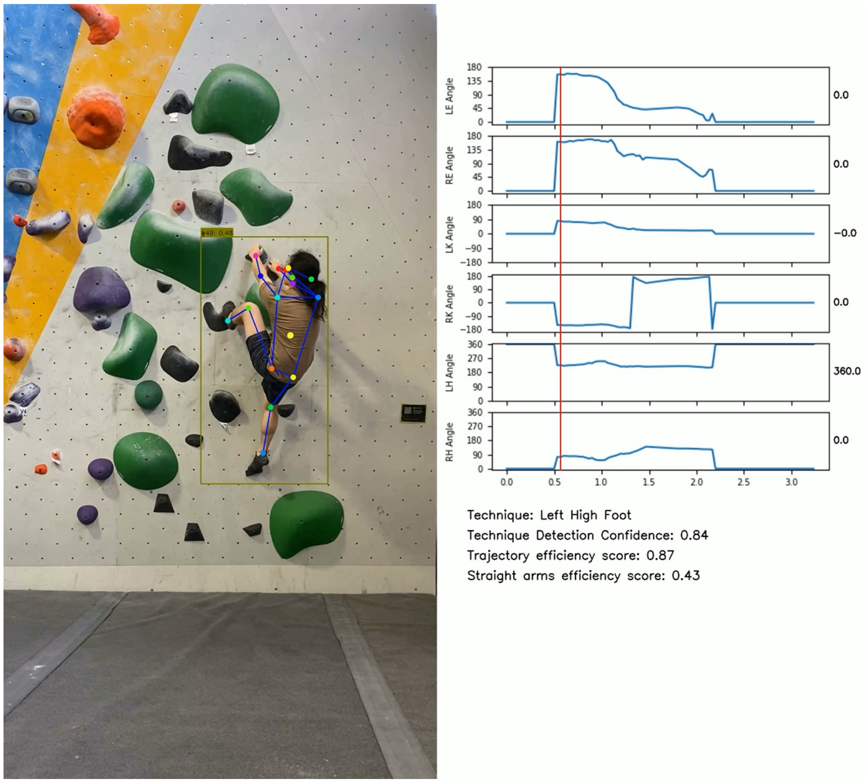
[15] Y. Xu, J. Zhang, Q. Zhang, and D. Tao. Vitpose: Simple vision transformer baselines for human pose estimation. *NeurIPS*, Oct 2022.

A. Appendix



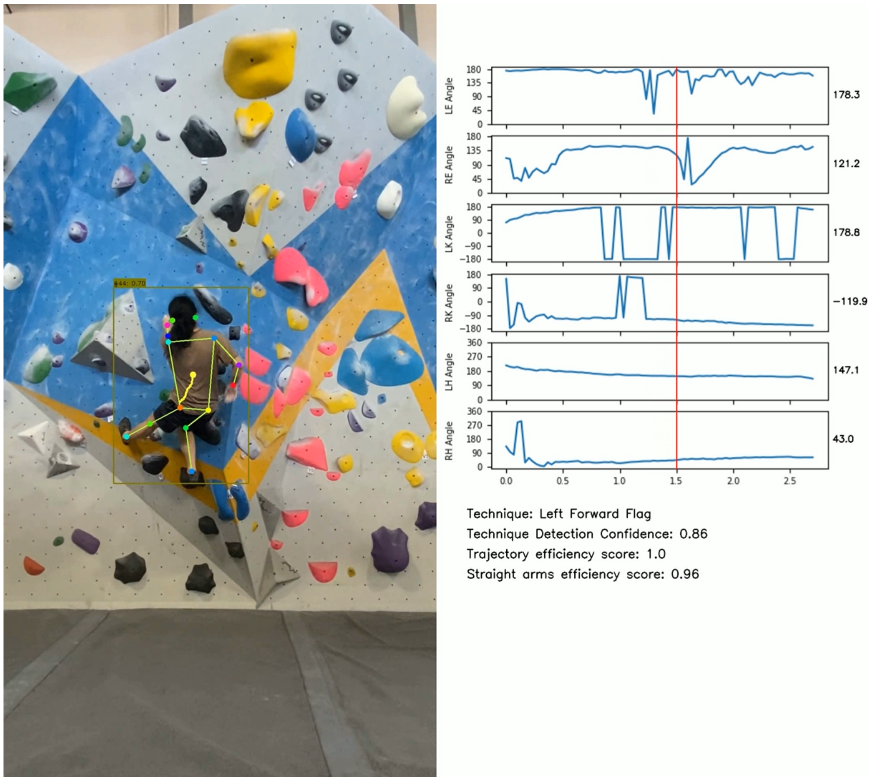
```
Technique: Right Drop Knee
Technique Detection Confidence: 0.81
Trajectory efficiency score: 0.99
Straight arms efficiency score: 0.57
```

Figure 4: Example of technique detector correctly classifying a right drop knee.



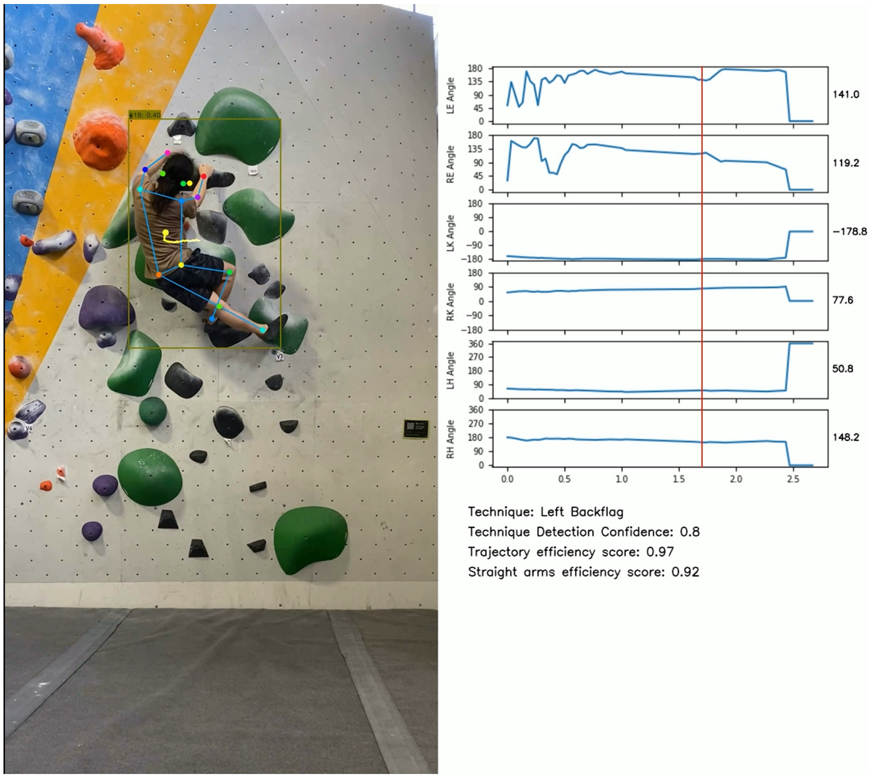
Technique: Left High Foot
Technique Detection Confidence: 0.84
Trajectory efficiency score: 0.87
Straight arms efficiency score: 0.43

Figure 5: Example of technique detector correctly classifying a left high foot.



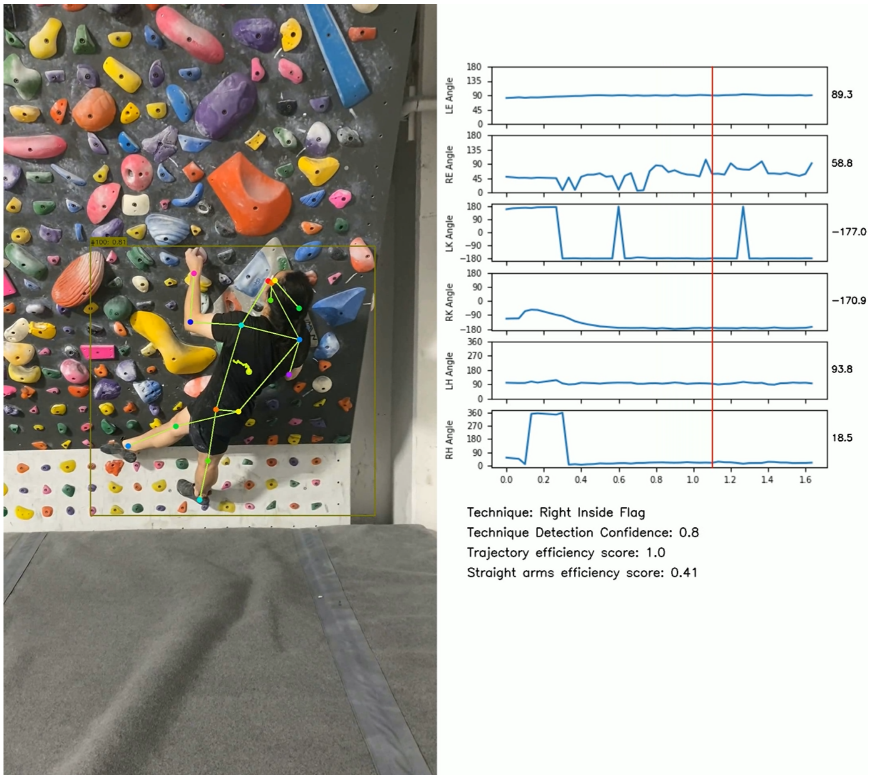
Technique: Left Forward Flag
Technique Detection Confidence: 0.86
Trajectory efficiency score: 1.0
Straight arms efficiency score: 0.96

Figure 6: Example of technique detector correctly classifying a left forward flag.



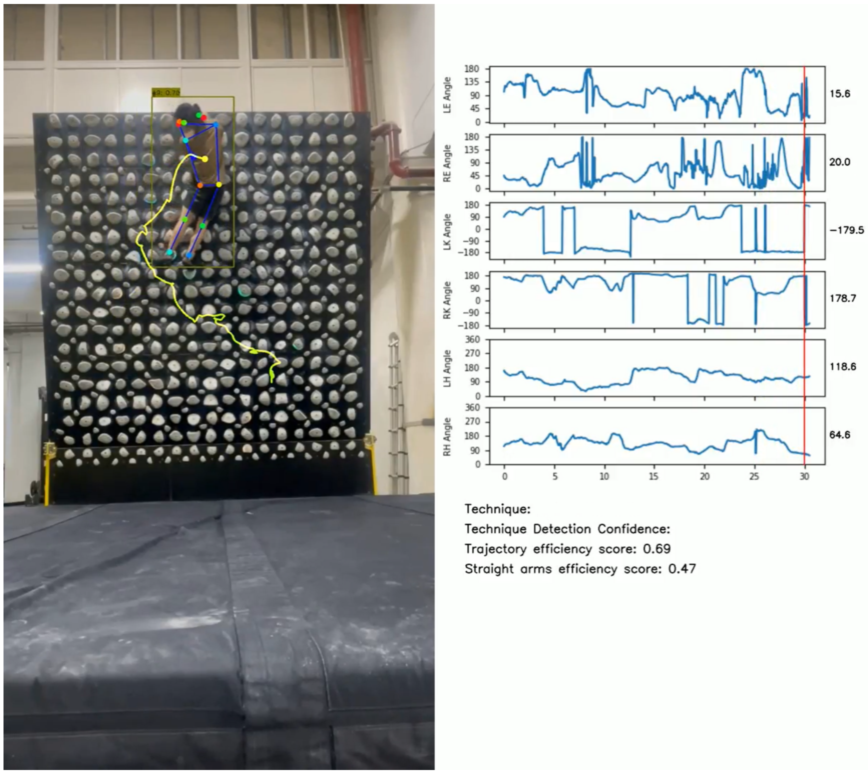
Technique: Left Backflag
Technique Detection Confidence: 0.8
Trajectory efficiency score: 0.97
Straight arms efficiency score: 0.92

Figure 7: Example of technique detector correctly classifying a left backflag.



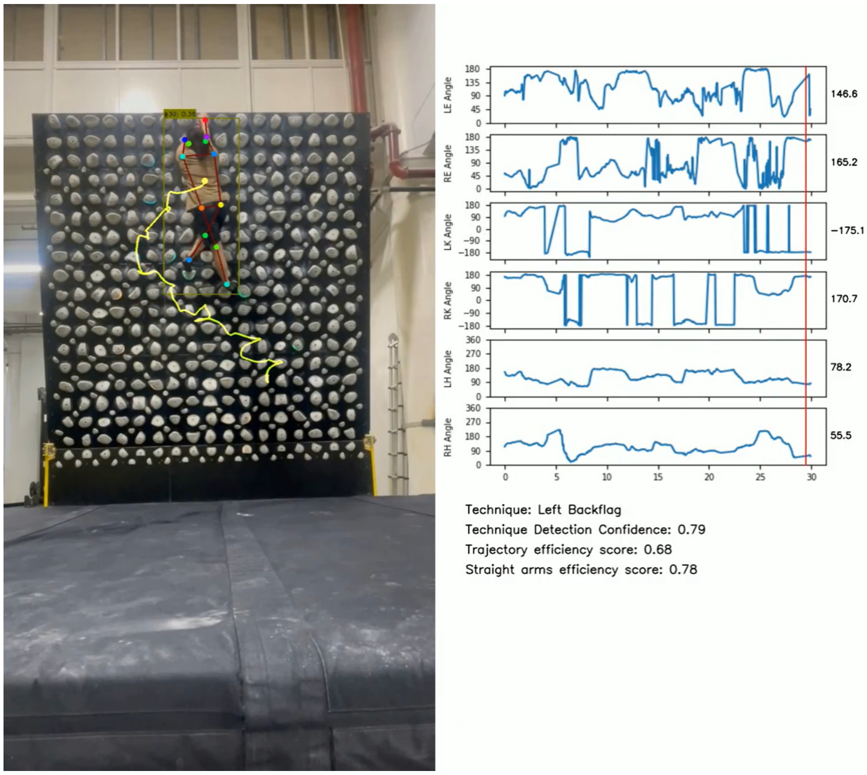
Technique: Right Inside Flag
Technique Detection Confidence: 0.8
Trajectory efficiency score: 1.0
Straight arms efficiency score: 0.41

Figure 8: Example of technique detector correctly classifying a right inside flag.



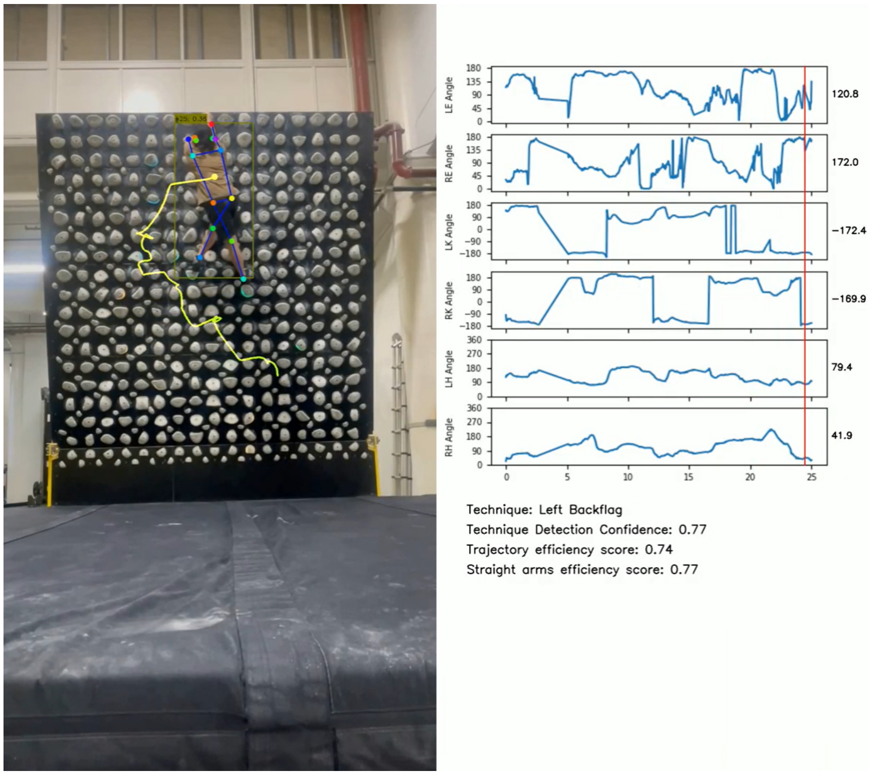
Technique:
Technique Detection Confidence:
Trajectory efficiency score: 0.69
Straight arms efficiency score: 0.47

Figure 9: Snapshot of generated climbing report. Boulder problem example 1. Climbing with more bent arms.



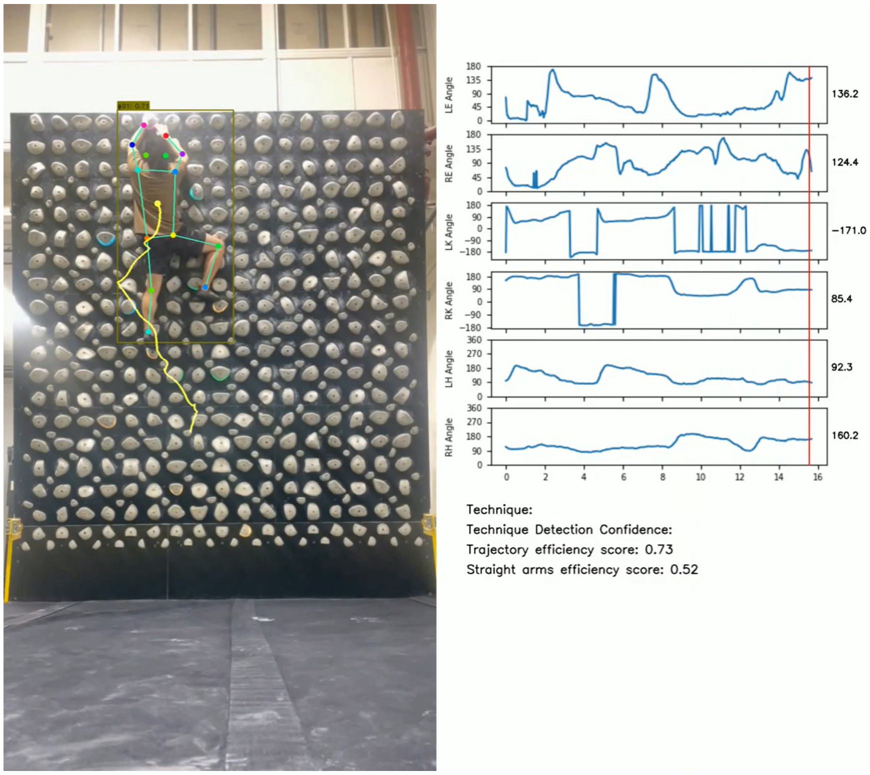
Technique: Left Backflag
Technique Detection Confidence: 0.79
Trajectory efficiency score: 0.68
Straight arms efficiency score: 0.78

Figure 10: Snapshot of generated climbing report. Boulder problem example 1. Climbing with more wasted movements.



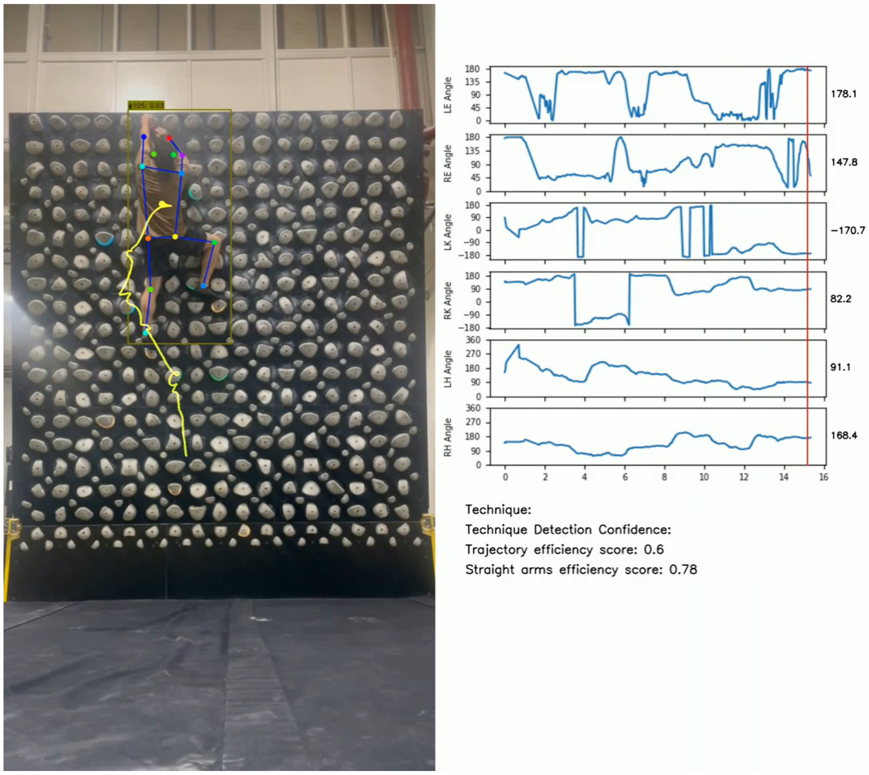
Technique: Left Backflag
Technique Detection Confidence: 0.77
Trajectory efficiency score: 0.74
Straight arms efficiency score: 0.77

Figure 11: Snapshot of generated climbing report. Boulder problem example 1. Climbing while trying to keep arms straight as often as possible and with fewer wasted movements.



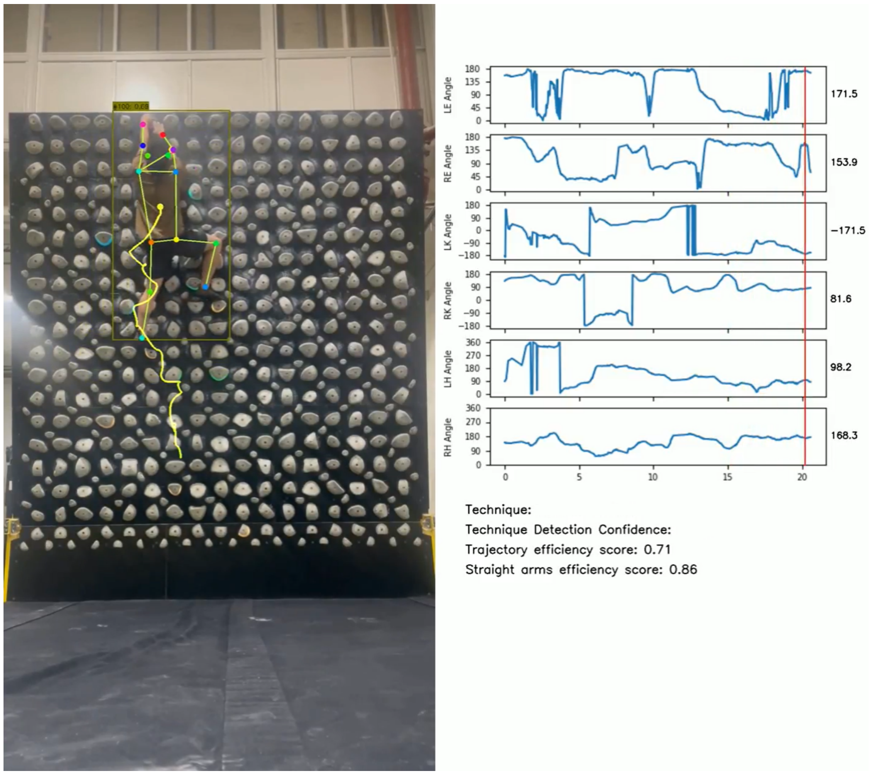
Technique:
Technique Detection Confidence:
Trajectory efficiency score: 0.73
Straight arms efficiency score: 0.52

Figure 12: Snapshot of generated climbing report. Boulder problem example 2. Climbing with more bent arms.



Technique:
Technique Detection Confidence:
Trajectory efficiency score: 0.6
Straight arms efficiency score: 0.78

Figure 13: Snapshot of generated climbing report. Boulder problem example 2. Climbing with more wasted movements.



Technique:
Technique Detection Confidence:
Trajectory efficiency score: 0.71
Straight arms efficiency score: 0.86

Figure 14: Snapshot of generated climbing report. Boulder problem example 2. Climbing while trying to keep arms straight as often as possible and with fewer wasted movements.