Final Report

Spasticity Findings

When researching new techniques in the detection of spasticity about half of the articles I read were focused on the lower limbs, while the other half was focused on the upper limbs. However, this is not an accurate representation of spasticity detection research. Most of the research focus has been dedicated to the detection of spasticity in the upper limbs. While the lower limb has gained more focus in the past few years, there remains a large divide in the amount of attention given to spasticity detection of the lower limbs

The two most common scales of clinically measuring spasticity are the Modified Ashworth Scale (MAS) and the Modified Tardieu Scale (MTS). The MAS is performed by extending the patients' limb from maximal flexion to maximal extension and then by moving the limb from maximal extension to maximal extension. Spasticity is then scored on a scale from 0 to 4 based on muscle tone and resistance. The MTS is performed by extending the patients' limb at 3 different velocities: as slow as possible(V1), the speed of the limb falling with gravity(V2), at a faster rate than gravity(V3). Two angles of the limb are measured: angle of catch seen at V2 or V3(R1) and full range of motion achieved with muscle at rest at V1(R2). Spasticity severity is then assessed on a range from 0 to 5 depending on the muscle resistance and joint angle. These two scales are the two most common scales used as a control to compare findings from proposed detection methods.

The most used technology in new methods to identify spasticity are sEMG sensors, IMU sensors, and motion capture systems. Using these systems, they tracked and analyzed joint angles, angular velocity, sEMG signals, tonic stretch reflex threshold, and other variables to help identify spasticity.

Many recent studies have analyzed sEMG signals in some way to aid spasticity detection. Wu et al. (2024) analyzed the time-domain and frequency-domain features of the sEMG signals [1]. They trained an SVM classifier on these features and were able to achieve a recognition accuracy of 95% for MAS 1+, but for MAS 2 and 3, the accuracy fell below 70% which could possibly be due to their limited sample size. Overall, they achieved an average classification accuracy of 78.85%. Becker et al. (2019) measured muscular coherence between antagonistic muscles using sEMG data and found that muscular coherence largely depended on angular velocity [2]. They then proposed that spasticity is primarily affected by an increasing co-activation of antagonistic muscles. Gurluk et al.

(2024) used IMU sensors in combination with sEMG data to find the safe range of motion before muscle co-contraction to aid in rehabilitation of spasticity [3]. Guo et al. (2023) also used IMU and sEMG sensors to measure muscle activation and found that a decreasing sEMG frequency can be used to identify severe spastic responses [4]. Another variable that is evaluated from sEMG signals is the Tonic Stretch Reflex Threshold (TSRT) which is the joint angle at which muscle activation occurs. Levin et al. (2024) explored the TSRT method and explained that in spastic patients, the TSRT lies within the normal biomechanical joint range [5]. Frenkel-Toledo et al. (2021) measured TSRT to show that it can be a reliable way to detect spasticity [6].

Other studies have looked closely at the angle of the joint to determine when spasticity occurs. Tan et al. (2018) used motion capture to determine spasticity from angular velocity and acceleration [7]. They proposed that when spasticity occurs, the extension speed and acceleration of the limb greatly decreases. De Santis et al. (2024) measured knee kinematics during extension and flexion using IMU sensors and a motion capture system [8]. By analyzing the first swing angle (FSA) they were able to determine that a FSA <= 80 degrees is a conservative threshold for spasticity.

Human Pose Estimation

Human pose estimation is a task within the field of computer vision that aims to identify the location of human joints from images or videos. It has gained significant attention in computer vision research due to its wide range of applications such as healthcare, sports, surveillance, video games, and more.

Many studies have compared the accuracy of human pose estimation models, such as OpenPose, by comparing joint angles with motion capture data or radiography data. Saiki et al. (2023) measured the hip-knee-ankle angle in knee osteoarthritis patients and compared it with radiography data [9]. After analyzing the data, they found that OpenPose had a large correlation with radiography and produced a fixed error of 1.077°, an absolute error of 1.799°, and a random error of 2.171°. Helmstetter et al. (2021) aimed to compare OpenPose with Xsens motion capture for biomechanical analysis of human movement [10]. To do this they measured the joint angle of the right elbow while the participant hammered a nail. They found that both signals from the Xsens data and OpenPose data had the same shape, but with an offset of about 25°. Kim et al. (2021) measured 20 joint angles across various tasks from different angles to test the accuracy of OpenPose against the Xsens motion capture [11]. They discovered that 16 and 20 out of 20 joint angles had an error smaller than 10% and 20% with a total mean error of 8.4° from frontal views. They also found that from a non-frontal view, 9 and 20 out of 20 joint angles had errors smaller than 10% and 20% with a total mean error of 10.4°. For the left and right knee flexion angles from a frontal view they measured a root mean square error (RMSE) of about 9° for the left knee and a RMSE of about 11° for the right knee. When measuring from a non-frontal view they measured an RMSE of about 11° for the left knee and an RMSE of about 5° for the right knee.

Mroz et al. (2021) conducted a study comparing the quality of human pose estimation between OpenPose and BlazePose [12]. Their findings showed that OpenPose was better at predicting accurate key points than BlazePose in videos of movements that are relevant to clinicians. While OpenPose was more accurate than BlazePose, they did note that BlazePose offers other advantages such as real-time video processing, a visibility score, and a 3D-coordinate for each key point.

Model Comparison

BlazePose is a human pose detection model with Lite, Full, and Heavy versions available for use. The model uses 33 key points, with 10 focused on the lower limbs, the hips (2), knees (2), ankles (2), heels (2), and foot index (2). The input format is video frames represented as a 256x256x3 array with normalized RGB values. The output is a 33x5 tensor that contains an x-coordinate, y-coordinate, z-coordinate, a visibility score, and a presence score for each key point. BlazePose does single person pose estimation and tracking and the person must be closer than 14 feet with the head visible. If there are multiple people in the image an object detector will be needed to separate the humans into bounding boxes and then fed into BlazePose for key point detection. BlazePose is compatible with the Android OS.

MoveNet is a human pose detection model with Lightning and Thunder variants available. There is a total of 17 key points with 6 focused on the lower limbs which are the hips (2), knees (2), and ankles (2). The input format is a video frame or image represented as an int32 tensor with shaped 192x192x3 for Lightning and 256x256x3 for Thunder with RGB values from 0 to 255. The output is a float32 tensor of shape [1, 1, 17, 3] containing the 17 key points, each with a x-coordinate, y-coordinate, and a confidence score. MoveNet only does single person pose estimation, but if more than one person is in the image the model will try to focus on the person closest to the image center. It is also recommended for best performance that the person targeted in the image is 3-6ft from the camera. MoveNet is compatible with the Android OS.

PoseNet is a human pose detection model with MobileNetV1 and ResNet50 architectures available. There is a total of 17 key points with 6 focused on the lower limbs which are the hips (2), knees (2), and ankles (2). The input format is RGB images or videos with a default size of 257x257x3 but can be changed. The output is a float32 tensor of shape [1, 1, 17, 3] containing the 17 key points, each with a x-coordinate, y-coordinate, and a confidence score. Besides single person pose estimation/tracking, PoseNet can also do multi-person tracking. PoseNet is compatible with the Android OS.

OpenPose is one of the most popular human pose detection models that offers 2D real-time multi-person key point detection and 3D real-time single-person key point detection and tracking. They offer 15, 18, and 25 key point estimation for the body and feet. With 25 key points there are 13 focused on the lower limbs which are the hips (3), knees (2), ankles (2), big toes (2), small toes (2), and the heels (2). Input can be an image or video from a webcam, but they also support Flir/Point grey cameras. The output is a float array that contains an x-coordinate, y-coordinate, and a confidence score for each key point. OpenPose is not compatible with the Android OS.

Ultimately, BlazePose proved to be the most suitable model for lower limb spasticity detection due to its inclusion of additional key points in the lower limbs. Furthermore, its use of an XYZ coordinate system, rather than just XY, offers more comprehensive spatial data, which enhances the accuracy and depth of spasticity analysis.

Preliminary Dataset Acquisition

To gain hands-on experience with BlazePose and to evaluate the optimal conditions for pose estimation, I designed a trial aimed at collecting a preliminary human gait analysis dataset. The primary goal was to capture videos of individuals walking in order to support pose estimation, joint angle calculations, and lower limb movement assessments. Additionally, these trial videos were used to analyze which video recording conditions most effectively enhance the accuracy of the BlazePose model.

In the trials, five subjects were asked to walk back and forth under varying conditions to evaluate how different factors affect pose estimation accuracy. Each subject walked with different combinations of clothing, pants/trousers with shoes, pants/trousers with no shoes, shorts with shoes, and shorts with no shoes. Then to assess the influence of camera positioning, each walking sequence was recorded from four distinct viewpoints: oblique (45-degree angle), sagittal (side view), frontal (facing the camera), rear (walking away from the camera). Recordings were conducted in a well-lit environment with a background that provided strong contrast against the subject to enhance key point visibility. The walking path was marked to capture at least three full gait cycles to ensure sufficient motion data for analysis. An Android phone with default camera settings was used for video capture, mounted on a tripod positioned approximately at the subject's hip height to maintain a consistent perspective across recordings.

After analyzing the recorded videos, several insights emerged regarding factors that influence pose estimation quality. There was no noticeable difference in performance between tight-fitting trousers and shorts, except in cases where long pants partially obscured the ankles or feet which occasionally led to key point detection errors. Footwear had a more subtle impact: while there was no significant difference between socks and shoes themselves, wearing socks or shoes that contrasted in color with the pant leg improved key point visibility and overall estimation accuracy. Among the different camera angles, the sagittal view consistently yielded the most accurate results for both join angle calculations and pose estimation. The oblique view also performed well, though with slightly reduced accuracy. Frontal and rear views produced reliable key point detection overall, but the angle of observation provided limited value for analyzing joint angles and lower limb movement. Across all trials the foot index key point was identified as the least reliable, often exhibiting the highest error.

OptiTrack Angle Comparison

To verify the accuracy of joint angle calculations obtained from BlazePose coordinates, I recorded videos simultaneously with the OptiTrack motion capture system. This allowed for a comparison between the key points detected by BlazePose and the precise coordinates for the OptiTrack system. Using these coordinates from BlazePose and OptiTrack I calculated two key angles: the knee angle, which was measured as the angle formed behind the knee, specifically between the thigh and the shin, and the ankle angle, which is the angle between the shin and the foot. These calculations enabled a direct assessment of BlazePose's accuracy in estimating joint angles, using the high-precision data from OptiTrack as a reference.

After calculating the joint angles from both BlazePose and the OptiTrack system, the next step was to synchronize the data. Since the recordings did not start simultaneously, I aligned the angle time series by identifying a common motion pattern and shifting the data accordingly to achieve temporal alignment. Once synchronized I compared the angles using several statistical metrics to evaluate accuracy. I used the Root Mean Square Error (RMSE), Mean Absolute Error (MAE), Mean Error (ME), and the Pearson Correlation Coefficient.

For a representative sagittal-view video, with the subject's left side facing the camera, the calculated RMSE between BlazePose and OptiTrack revealed key trends in joint angle accuracy. The RMSE for the knee angles was 6.8° for the left knee and 7.3° for the right knee, indicating a relatively strong agreement. In contrast, the ankle angles showed significantly higher error, with RMSE values of 22.8° for the left ankle and 19.8° for the right ankle. The knee angle plots from BlazePose closely resembled those from OptiTrack in overall shape, though some variation in magnitude was observed. Meanwhile, the ankle angle graphs showed less similarity in both shape and value. These trends, stronger agreement in knee angle estimation and greater error in ankle angle estimation, were consistent across all trials between BlazePose and OptiTrack.

The relatively high error values, particularly in ankle angle estimation, may be attributed to several factors. One possible source is estimation errors from BlazePose, especially for key points near the feet which tend to be less accurate than key points further up the leg. Another possible source for the high error values is the discrepancies between the BlazePose key point definitions and the physical marker locations used in the OptiTrack system. BlazePose estimates joint centers based on visual landmarks, while the OptiTrack system relies on reflective markers placed according to anatomical reference points. In this study, I used the conventional lower body marker set in OptiTrack, which defines joint locations slightly differently than BlazePose. These differences in anatomical assumptions and marker placement can lead to systematic offsets in angle calculations between the two systems.

To contextualize my findings, I reviewed a related study that compared joint angle measurements from BlazePose to those obtained using Kinovea, a 2D video sports analysis tool. The researchers evaluated BlazePose's accuracy in estimating hip, knee, and ankle angles at three key phases of the running gait cycle: initial contact (IC), midstance (MS), and toe-off (TO). Their results showed that BlazePose performed well at IC and TO, but less so at MS. For the ankle the RMSE was 2.29° at IC, 2.05° at MS, and 2.54° at TO. The hip angle errors were relatively low across all phases – 2.29° at IC, 2.09° at MS, and 2.59° at TO. Knee angle errors were slightly higher with RMSE values of 2.77° at IC, 3.58° at MS, and 3.52° at TO [13]. While the study focused on running rather than walking and reported lower error values overall, its findings support the conclusion that BlazePose is generally reliable for measuring joint angles, particularly at the hip and knee, even though my results showed higher errors, especially at the ankle.

BlazePose Errors

In analyzing BlazePose's output across various trials, I identified two recurring categories of error. The first type is swapped limb errors, where BlazePose incorrectly assigns key points from one side of the body to the over, for example, predicting the right leg as the left and vice versa. The second category involves localized key point inaccuracies, where one or more key points are placed incorrectly. This sometimes involves just a single point, such as the heel or foot index, but in other cases, the entire foot may be mispositioned. These errors can significantly affect joint angle calculations, especially when evaluating asymmetrical gait patterns. Notably, these errors tend to occur in clusters, where multiple incorrect key points appear together in clusters of frames across a short window.

Detecting these types of errors Is crucial for improving the overall accuracy and reliability of joint angle calculations using BlazePose. Swapped limb errors can drastically skew results if left uncorrected, as they invert the anatomical context of the walking motion. Similarly, inaccurate or misplaced key points, especially when affecting joints like the ankle or knee, can introduce large deviations in angle measurements. By identifying frames where these errors occur, those frames can be corrected for further analysis.

Error Detection

To improve the reliability of joint angle calculations and mitigate the impact of key point estimation errors, I explored several automated methods for detecting anomaly frames in BlazePose output. These methods focused primarily on analyzing the x-coordinate trajectories of key points over time, as lateral inconsistencies often come from pose estimation errors. The approaches ranged from basic statistical techniques to more advanced machine learning and signal comparison strategies. Specifically, I tested: (1) a basic z-score method applied to frame-to-frame x-coordinate differences, (2) a sliding window approach to capture local anomalies, (3) an Isolation Forest model for unsupervised anomaly detection, (4) a smoothed signal residual method that compares original signals to their smoothed counterparts, and (5) a sigmoid curve matching approach that fits step-like transitions to segments of the x-coordinate signal. Each method was designed to flag frames where key point trajectories deviated from expected motion patterns, enabling further analysis and correction.

The first method I implemented for error detection was a basic-z-score analysis based on x-coordinate changes over time. I focused this method on the right and left ankle, heel, and foot index key points, since any errors I had observed involved inaccuracies in these lower extremity key points. Notably, when the knee key point is incorrect, errors were always present in the foot, but not vice versa, which is the reason for the narrowed focus. For each key point I calculated the frame-to-frame difference in x-coordinates and then computed the z-score for each of these differences across the entire video. A frame was labeled as erroneous if all three key points (ankle, heel and food index) on either foot had z-scores greater than a threshold of three. A threshold of three was chosen, as it is a standard cutoff for identifying outliers in z-score analysis, and initial testing showed that it provided a good balance between sensitivity to large errors and a resistance to false positives from typical gait variation. This method was effective at detecting large, isolated errors where sudden jumps in x-coordinate values produced high z-scores. However, it struggled to identify large groups of consecutive errors, where abrupt transitions may only appear between the first and last frame of the segment. It also performed poorly at catching smaller deviations, particularly in videos where natural walking speed changes or variations in step distance occurred. This issue was especially pronounced in oblique view recordings, where the camera angle and positioning introduced perspective distortion, causing apparent step sizes to vary throughout the video depending on the subject's distance from the camera. While this method provides a simple and interpretable baseline for error detection, its limitations highlight the need for more adaptive techniques capable of capturing subtle or sustained key point inaccuracies.

To better capture smaller, localized errors that the basic z-score method often missed, I implemented a sliding window z-score approach using frame-to-frame x-coordinate differences. After testing several window sizes, a length of 30 frames was found to be the most effective for detecting anomalies. Within each window, I calculated the z-score for each key point's x-difference, focusing again on the ankle, heel, and foot index key points for both sides of the body. I used the same z-score threshold of 3 but adjusted the error classification criteria: a frame was labeled as erroneous if more than one of these three key points on either foot exceeded the threshold, or if the left or right foot index key point alone exceeded it. This adjustment was designed to improve the detection of subtle or partial foot errors, especially since the foot index key point consistently showed the lowest accuracy during testing. The sliding window method retained the basic z-score's strength in identifying large, isolated errors and improving on the detection of smaller errors. However, it still struggled to consistently identify extended sequences of erroneous frames, where sudden transitions were absent, and error patterns were sustained over multiple frames. Overall while these statistical approaches provided a solid foundation for detecting obvious and some localized anomalies, their limitations in handling more complex or sustained error patterns highlighted the need for more advanced detection techniques.

As a more advanced approach, I experimented with using an isolation forest model to detect anomalous frames based on the x-coordinate time series of key points. The goal was to leverage a model-based method that could learn patterns of normal motion and flag deviations without relying on fixed thresholds. I used x-coordinate data from hip, knee, ankle, heel, and foot index key points for both lower limbs, and extracted features such as mean, standard deviation, min, max, average velocity, and velocity variability. The isolation forest works by constructing decision trees that are recursively split to isolate observations; points that require fewer splits to isolate are more likely to be anomalies. The contamination parameter, which estimates the expected proportion of anomalies in the data, influences how the model flags unusual points. I tested different contamination levels, including no fixed contamination rate, but since the number of anomalies across videos varied with some videos having no errors, the results varied greatly. For labelling anomalies, I used the same criteria as the sliding window method, requiring multiple key points to be flagged or just either foot index alone. In practice, the isolation forest consistently returned too many anomalies and often failed to detect the true errors, suggesting that the model struggled to differentiate between normal walking variations and true key point estimation errors. I was unable to identify a reliable set of parameters that worked well across all videos. However, this approach could be an area for future refinement, especially with better-tuned features or possibly supervised learning strategies.

Another method I explored was based on analyzing the residuals between the original x-coordinate signal and a smoothed version of that signal. The goal was to highlight deviations by comparing the raw, noisy signal to a cleaner, more expected version. I tested two different smoothing techniques: the Savitzky-Golay filter and a Kalman Filter. After smoothing the data, I computed the residual for each key point by taking the absolute difference between the original and smoothed signals. I then calculated the z-score for each residual data point using the whole residual signal mean and standard deviation. A frame was labeled as erroneous if at least one key point had a residual z-score above a threshold of three. This method used all lower body key points, not just the key points on the feet. However, it proved to be less effective in testing. Both smoothing algorithms were influenced too heavily by the original errors, which distorted the smoothed signal and reduced their usefulness as a baseline. As a result, the method frequently generated too many false positives, particularly in frames surrounding true errors. The reliance on a perfect smoothed signal limited the accuracy of the smoothed residual approach.

The sigmoid curve matching method was developed to detect errors by modeling the expected motion patterns of the foot key points during walking. First, the x-coordinate signal was segmented by identifying regions where the gradient was less than 0.005, then splitting those regions at their midpoints to isolate the flat regions between steps. For each segment, a sigmoid step curve was fit using the sigmoid function: $f(x) = \frac{L}{1+\exp(-k*(x-x\theta))}$. where L represents the maximum value of the curve, k controls the steepness of the curve, x is the input variable, xθ is the midpoint of the transition. These parameters were optimized using the scipy.curve_fit method to best align the sigmoid curve with the x-coordinate data of each segment. The residuals between the original signal and the fitted sigmoid were then calculated as the absolute difference between the two. A z-score was computer from these residuals, and frames were marked as errors if three or more of the ankle, heel, and foot index key points exceeded a threshold of three. To better capture clusters of smaller errors that often occur adjacent to larger ones, the threshold was halved for frames directly next to already-detected anomalies. After this first pass, a post-processing step was used to re-include any nearby frames to anomaly frames that may have been filtered out by not having enough occurrences of key points above the threshold. This method proved to be the most effective among all tested approaches, with strong performance in identifying both isolated and grouped anomalies.

Error Correction

After identifying frames with potential BlazePose key point errors, the next step was to explore methods for correcting those inaccuracies to improve the reliability of joint angle calculations. I tested two different correction strategies: (1) a Kalman filter approach, which aimed to smooth and adjust the noisy key point data using filtering, and a (2) reprocessing method, which attempted to correct errors by re-running BlazePose predictions on the error frames. Each method was designed to improve the predictions for the identified error frames.

The first method I tested to correct errors in BlazePose key point predictions was a Kalman filter based method implemented with the pykalman library. This approach updated the values of all lower limb key points at frames that had been detected as errors. The goal was to generate better position estimations for anomaly predictions from BlazePose.

The Kalman filter was configured by setting the initial state mean and defining custom transition and observation matrices, along with observation and transition covariances (see appendix A for parameter values). Before filtering, anomaly values were removed from the x-coordinate and y-coordinate time series and replaced using linear interpolation. The interpolated signal was then passed through the Kalman filter to produce a smoothed trajectory. Finally, the original x and y-coordinate signals were updated with the filtered values, but only at the frames that were flagged as errors/

This method proved to be highly effective for correcting small, isolated anomalies or brief gaps in the signal. It produced visually smoother trajectories that better aligned with the expected motion. However, its performance dropped in cases where errors went undetected or when there were large groups of consecutive errors. Nonetheless, this method provided a strong correction method for many common error scenarios.

The second correction method I tested involved reprocessing individual error frames using BlazePose I image mode. Unlike the initial pass, which used BlazePose's video mode to process the full video sequentially, this method loaded and reanalyzed only the specific frames labeled as anomalies. By running BlazePose in image mode, the model was not influenced by prior or subsequent frames, allowing it to make more accurate predictions. Each reprocessed frame was used to update the corresponding frame values in the original dataset. This approach was effective at correcting most errors in large groups that may have resulted from temporal dependencies in the video mode. While some errors still occurred, it offered a noticeable improvement over initial estimations in many cases.

Both correction methods, the Kalman filter smoothing and reprocessing frames with BlazePose in image mode, offered useful improvements to key point accuracy in anomaly frames. The Kalman filter approach was very effective at correcting small gaps or single errors by using linear interpolation and smoothing, but it struggled with large consecutive groups of errors and undetected errors. On the other hand, reprocessing anomaly frames with BlazePose in image mode provided better results overall. Since the model was not influenced by adjacent frames, it often produced more accurate predictions over large gaps even if adjacent errors went undetected.

Final Detection and Correction Pipeline

As shown in the pseudocode in Appendix B, the final error detection and correction algorithm begins by continuously identifying anomalies using the sigmoid curve matching algorithm in a loop. Once anomalies are detected, either the reprocessing with BlazePose method or the Kalman filter method is applied to correct the anomalies. If the anomalies are new detections and have not been detected before then the algorithm attempts to re-estimate the errored frames using BlazePose in image mode. But if algorithm has already attempted to use the BlazePose approach on the anomalies then the Kalman filter approach is used. After each iteration the z-score threshold is increased slightly to reduce false positives and refine the set of detected anomalies. This loop continues until no new anomalies are found. The combination of the two correction methods works incredibly well together as the BlazePose method can sometimes leave small gaps or a single error, which the Kalman filter approach excels at correcting. The two methods working together compensate for the other's weakness which results in consistently accurate and reliable key point data.

Possible Improvements

While the final error detection and correction pipeline produced strong results, limitations and areas for enhancement remain. A major constraint was the absence of ground truth data for the preliminary dataset. None of the videos I had recorded and used contained labeled or verified key point data, making it difficult to quantitatively assess the accuracy of key point estimations or correction performance. This limited evaluation to qualitative and visual inspection.

One promising direction for future work would be to incorporate a learning-based model, such as an LSTM or another time-series-aware machine learning model. These models could leverage temporal patterns in the coordinate data to better distinguish between natural walking patterns and anomalies.

Currently, the algorithm operates entirely in a post-processing context. An exciting improvement would be adapting the algorithm for real-time, live-video error detection and correction. This would broaden its applications in clinical assessments.

Finally, there are opportunities to enhance the existing methods. Combining y-coordinate data with x-coordinate data during the anomaly detection could increase robustness of the algorithm, especially for key points that shift both laterally and vertically during normal walking motions. Additionally, revisiting the isolation forest approach with a more comprehensive feature set, potentially including more velocity-based features, might yield better performance and more accurate anomaly detection.

Appendix A: Kalman Filter Parameters

[x, y, x velocity, y velocity]

- Initial State Mean: [0, 0, 1, 1]
- Transition Matrices: [1, 0, 1, 0], [0, 1, 0, 1], [0, 0, 1, 0], [0, 0, 0, 1]
- Observation matrices: [1, 0, 0, 0], [0, 1, 0, 0]
- Transition Covariance: $\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$
- Observation Covariance: $\begin{bmatrix} 0.01 & 0 & 0 & 0 \\ 0 & 0.01 & 0 & 0 \\ 0 & 0 & 0.01 & 0 \\ 0 & 0 & 0 & 0.01 \end{bmatrix}$

Appendix B: Final Error Detection and Correction Pipeline (Pseudocode)

```
Initialize:
  anomaly_indices ← ∅
  previous_anomalies ← ∅
  z_score_threshold ← initial threshold (e.g., 3.0)
  x_data, y_data ← initial coordinate data (deep copies)
Repeat until no new anomalies are found:
  1. Detect anomalies using sigmoid matching:
    new_anomalies ← detect_sigmoid_anomalies(x_data, z_score_threshold)
    anomaly_indices ← anomaly_indices ∪ new_anomalies
  2. Increment threshold to reduce false positives:
    z_score_threshold ← z_score_threshold + 0.5
  3. If anomalies are detected:
    a. If new anomalies ≠ previous anomalies:
      - Update previous_anomalies ← copy of anomaly_indices
      - Apply BlazePose image-based reprocessing to affected frames
      - Update x_data, y_data with new predictions
    b. Else if new anomalies exist (but match previous):
      - Apply Kalman filter correction to new_anomalies
      - Update x_data, y_data with smoothed predictions
    c. Else:
      - Break loop (no new anomalies)
  4. If no anomalies:
    - Break loop
Return final corrected x_data and y_data
```

References

1. Wu, L., Bao, T., Zou, B., Liu, J., Sun, F., Wang, C., Zhou, P., Chakrabarty, S., & Xie, S. Q. (2024). Quantitative Assessment of Lower Limb Spasticity Using sEMG Data and Bayesian-Optimised SVM. *2024 International Convention on Rehabilitation Engineering and Assistive Technology and World Rehabilitation Robot Convention, WRRC 2024 - Proceedings*. https://doi.org/10.1109/WRRC62201.2024.10696863

2. Becker, S., von Werder, S. C. F. A., Lassek, A. K., & Disselhorst-Klug, C. (2019). Time-frequency coherence of categorized sEMG data during dynamic contractions of biceps, triceps, and brachioradialis as an approach for spasticity detection. *Medical and Biological Engineering and Computing*, *57*(3), 703–713. https://doi.org/10.1007/s11517-018-1911-3

3. Gurluk, A., Bilal, H. M., & Hocaoglu, E. (2024). Quantitative Analysis of Ankle Spasticity Through Electromyography Signal. *TIPTEKNO 2024 - Medical Technologies Congress, Proceedings*. https://doi.org/10.1109/TIPTEKNO63488.2024.10755260

4. Guo, X., Tang, J., Crocher, V., Klaic, M., Oetomo, D., Xie, Q., Niu, C. M., & Tan, Y. (2023). Using sEMG Signal Frequency to Evaluate Post-Stroke Elbow Spasticity. *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS*. https://doi.org/10.1109/EMBC40787.2023.10340707

5. Levin, M. F., Piscitelli, D., & Khayat, J. (2024). Tonic stretch reflex threshold as a measure of disordered motor control and spasticity – A critical review. In *Clinical Neurophysiology* (Vol. 165, pp. 138–150). Elsevier Ireland Ltd. https://doi.org/10.1016/j.clinph.2024.06.019

6. Frenkel-Toledo, S., Solomon, J. M., Shah, A., Baniña, M. C., Berman, S., Soroker, N., Liebermann, D. G., & Levin, M. F. (2021). Tonic stretch reflex threshold as a measure of spasticity after stroke: Reliability, minimal detectable change and responsiveness. *Clinical Neurophysiology*, *132*(6), 1226–1233. https://doi.org/10.1016/j.clinph.2021.02.390

7. Tan, T. J., Saito, M., & Mitobe, K. (2018). Detection of spasticity angle of occurrence in upper limb using magnetic MoCap and construction of a severity evaluation method. *IEEJ Transactions on Electrical and Electronic Engineering*, *13*(6), 851–857. https://doi.org/10.1002/tee.22638

8. de Santis, D., & Perez, M. A. (2024). A portable system to measure knee extensor spasticity after spinal cord injury. *Journal of NeuroEngineering and Rehabilitation*, *21*(1). https://doi.org/10.1186/s12984-024-01326-9

9. Saiki, Y., Kabata, T., Ojima, T., Kajino, Y., Inoue, D., Ohmori, T., Yoshitani, J., Ueno, T., Yamamuro, Y., Taninaka, A., Kataoka, T., Kubo, N., Hayashi, S., & Tsuchiya, H. (2023). Reliability and validity of OpenPose for measuring hip-knee-ankle angle in patients with knee osteoarthritis. *Scientific Reports*, *13*(1). https://doi.org/10.1038/s41598-023-30352-1

10. Helmstetter, S., Sänger, J., Germann, R., & Matthiesen, S. (2021). How to use human pose estimation to measure the hand-arm motion in craft application with no influence on

the natural user behavior. *Procedia CIRP*, *100*, 631–636. https://doi.org/10.1016/j.procir.2021.05.135

11. Kim, W., Sung, J., Saakes, D., Huang, C., & Xiong, S. (2021). Ergonomic postural assessment using a new open-source human pose estimation technology (OpenPose). *International Journal of Industrial Ergonomics*, *84*. https://doi.org/10.1016/j.ergon.2021.103164

12. Mroz, S., Baddour, N., McGuirk, C., Juneau, P., Tu, A., Cheung, K., & Lemaire, E. (2021). Comparing the Quality of Human Pose Estimation with BlazePose or OpenPose. *BioSMART 2021 - Proceedings: 4th International Conference on Bio-Engineering for Smart Technologies*. https://doi.org/10.1109/BioSMART54244.2021.9677850

13. Srinivasan, H. K., Mathunny, J. J., Devaraj, A., Gogoi, H., Govindasamy, K., & Karthik, V. (2025). Can BlazePose Accurately Assess Joint Angles in Outdoor Running Environments? *Biomedical Human Kinetics*, *17*(1), 13–25. https://doi.org/10.2478/bhk-2025-0002