A Novel Approach Using Smoothing to Detect Errors in OpenPose Estimated Running Data

Jipp Krabbenborg, University of Twente, The Netherlands

OpenPose is a real-time, multi-person key point detection library that uses deep learning to identify and locate human body key points from images or videos and this can be used to calculate gait parameters. In this pose estimated data misdetections occur. However, the estimated data should be accurate to give correct advice to athletes and prevent injuries. In this research, a novel approach is developed to detect these errors using the moving average filter on OpenPose estimated running data. The data used in this research comprises datasets sourced from videos featuring individual runners from the sagittal plane. Performance metrics were calculated for two datasets using this approach. On the datasets the approach had an accuracy of 98% and 99% which means the approach has a very high ratio of correct predictions across both datasets.

Additional Key Words and Phrases: OpenPose, pose estimation, MATLAB, error detection, human gait, smoothing

1. INTRODUCTION

Nowadays, pose estimation technology entails the automated identification and tracking of key body points of individuals in videos. Gait parameters can be calculated based on this pose estimated data. These gait parameters can then be used to give athletes advice and prevent injuries [4, 8]. However the pose estimated data can contain misdetections [2, 3, 9] and therefore the data might not be accurate enough to give correct advice to athletes.

For this research OpenPose was used to do pose estimation. OpenPose is a real-time, multi-person key point detection library that uses deep learning to identify and locate human body key points from images or videos [1]. In this research, there was worked with already provided datasets, compromising raw OpenPose data. This OpenPose estimated data was extracted from different videos showing a single person running from a right sagittal view (right side view). The outcome of the use of OpenPose on videos comprises a sequence of coordinates for 25 joints over successive frames, essentially forming a set of 2D coordinates representing the motion of the joints across the frames. Figure 1 in appendix A.1 illustrates the 25 key body points tracked by OpenPose.

OpenPose is useful because videos represent the actual presentation of how someone runs which is practical for the advice that will be given to athletes. Besides with this video method, no extra equipment is needed to track the gait movement, however there are disadvantages. The algorithms should be able to identify human bodies in diverse environments, accommodating variations in lighting, occlusions, and different body types.

TScIT 40, February 2, 2024, Enschede, The Netherlands

Unfortunately, these challenges contribute to instances of misdetections [3, 9]. Ronchi and Perona [9] also concluded that there are four types of misdetections that can happen with pose estimation:

1. Jitter
2. Miss
3. Inversion
4. Swap

Jitter indicates a small localization error for example because of a frequent change in the position of a body joint. A *miss* indicates a large localization error for example when there is an occlusion and a person cannot be detected. *Inversion* indicates confusion between key points within an instance. This happens when for example two joints are detected as the same body part. *Swap* indicates confusion between different instances' key points when for example a left knee joint is seen as a right knee joint. To see what these types of errors look like in the videos, screenshots are added in appendix A.2. The figures 1.1, 1.2, 1.3 and 1.4 in appendix A.2 are screenshots taken of two consecutive frames in videos for each error. The term 'error' pertains to the misdetections within the estimated pose data.

Earlier research has been conducted and various filters are used to reduce and remove this noise in pose estimated data. Gauss et al. [2] utilized three filters on pose estimated data to smooth Skeleton Avatar Technology (SAT) animations, employing TensorFlow PoseNet for pose estimation. The research results indicate that various filters can be utilized depending on the emphasis placed on smoothness and validity. Smoothness refers to the continuity and regularity of the estimated body joint positions over the time, while validity relates to the accuracy of the true underlying movement and position of the key body joints. It is important to note that their focus was on the smoothness of the SAT animations rather than the validity. However, for the calculation of the gait parameters the validity is prioritized over the smoothness of the pose estimated data. Besides, smoothing techniques will smooth out the misdetections and not fix them. There should be looked into another way of fixing the errors and therefore error detection needs to happen first.

Nevertheless, the smoothing techniques can maybe still be used for the purpose of error detection. This would be an intriguing approach, since according to previous works, this approach is not used before. To determine the smoothing technique that can detect the errors the best, various filters should be investigated and tested on the data. The performance of this approach should also be evaluated to see how accurate this approach is. Therefore, the following three research questions will be explored:

1. *"How do various smoothing techniques contribute to the detection of errors in OpenPose estimated running data?"*
2. *""How can smoothing be effectively utilized to develop a method for detecting errors in OpenPose estimated running data?"*
3. *"How can the results of this method be evaluated to assess its performance?"*

Within this study, firstly there will be evaluated which smoothing technique is most effective on the OpenPose estimated running data used in this research. Next, there will be looked at how the results of this smoothing technique can be used for the purpose of error detection. Lastly, performance of this approach will be evaluated.

2. RELATED WORK

This paper aims to detect errors using smoothing techniques in OpenPose estimated running data. Therefore, this section delves into related research on using filters on pose estimated data, to use this as inspiration for this research.

In the research conducted by Stenum et al. [4] in 2021, spatiotemporal and kinematic gait parameters calculated by OpenPose were measured against simultaneously recorded three-dimensional motion capture from overground walking healthy adults. To remove noise from this OpenPose data, post-processing scripts were written in MATLAB. These scripts require manually selection of the misdetections and then the program reconstructs the datapoint according to the type of error. However, if these scripts are used on large datasets, it would take a lot of time to select all the misdetections and characterize them as a certain error type. Therefore, the process of error detection should be automated and these post-processing scripts cannot be used for this research.

He et al. [3] in 2023, proposes the noise-resistance pose estimation (NRPose) to improve the accuracy of multi person pose estimation. There are two types of noise regarding multi-pose estimation. Aleatoric noise and epistemic noise. The former represents the noise inherent in the observations, such as the background. The latter indicates the noise brought by the priori hypotheses, such as the inappropriate key point relations. NRPose is integrated with two modules that mitigate the effects of aleatoric and epistemic noise. This is a very interesting module that is used to remove noise from pose estimated data. However, the NRPose is focused on multi person pose estimation and therefore the data is trained on a different dataset than used for this research. Additionally, I found the module too hard to understand since it is a very complex module. Therefore I chose to not use this module for my research.

The research done by Sato et al. [5] in 2019, used OpenPose to make it easier for physicians and neurologists to examine people with the Parkinson's disease. Videos were taken to examine the patients, and therefore OpenPose was used to do pose estimation. To detect patterns and frequencies in the time series data, the short-time autocorrelation function (ACF) is used, an applied form of a short-time Fourier Transform (STFT). The ACF would have been interesting to use for the detection of errors in this research. However, the ACF is not evaluated in this research because I could not get any promising results out of it. If there was more time I could have looked into it further, to maybe get some better results out of it.

Nakano, Sakura, Ueda et al. [6] aims to develop a 3D marker less motion capture technique, using OpenPose with multiple synchronized video cameras, and examine its accuracy in comparison with optical marker-based motion capture. They used a zero-lag fourth order Butterworth low pass filter to smooth out the raw kinematic data OpenPose processed. The cut off frequency was determined using residual analysis [7]. The use of filters is applicable on large datasets and it is relatively easy to implement on pose estimated data. Therefore, the Butterworth filter is evaluated in this research.

In 2021 Gauss et al. [2] conducted research about using Skeleton Avatar Technology (SAT) to accurately present real life moments of persons moving. Pose estimation was conducted using TensorFlow PoseNet, a versatile tool capable of analyzing poses in videos. It generates 2D coordinates just like OpenPose. In this research three filters are used to remove noise from the PoseNet estimated data. The first filter is the moving average filter. This filter is a simple filter to smooth out noisy data. Next, a high-pass filter using the Fourier Transform is used to eliminate high-frequency signal components. Lastly, the Kalman filter is used. This filter is used to remove noise from the key-point estimators. This paper's findings suggest that when prioritizing validity, moving averages are a superior choice, whereas for smoothness, the Kalman filter stands out as a better option. The high pass filter scored higher on validity than the Kalman filter but demonstrated lower smoothness scores than both the moving average and the Kalman filter. In this research the moving average and the high pass filter are evaluated because the focus is on the validity of the data. Besides, these filters are relatively easy to apply on the OpenPose estimated data used for this research.

Due to promising results in prior research, the Butterworth filter, a band pass filter, and the moving average filter were selected for evaluation in this study. These filters were chosen for their relative ease of implementation, saving valuable time in addressing research question 1. Furthermore, the band pass and moving average filters are anticipated to improve the data's validity.

3. STUDY

In this section, the first two research questions will be explored. Firstly, there is a subsection for the data provided for this research. Then there is a subsection about various smoothing techniques that are tested on the estimated data used in this research. Next, there will be evaluated which smoothing technique provides the most promising results for error detection. Lastly, a method is developed that uses a smoothing technique to detect errors in the OpenPose estimated data.

3.1 Provided data

In this research, there was worked with six already provided datasets, compromising raw OpenPose data. The raw data consists of 2D coordinates generated by OpenPose per key body joint for each frame of the video. The 25 key body joints OpenPose detects are shown in figure 1 in appendix A.1. The pose estimated data was

extracted from six videos where in each video a person running was recorded from a right sagittal view. The recording process involved utilizing a DSLR camera and the videos were recorded at a frame rate of 60 frames per second (fps). The videos were between 15 and 30 minutes long, depending on when the participant fatigued.

It's important to note that the methodologies employed in this research were tested and executed on the OpenPose raw data.

3.2 Testing filters

As discussed before, various filters can be used to smooth out and remove noise from pose estimated data. In this subsection, various filters are tested on the OpenPose data of the right knee joint and the results are investigated. To do so, filters were applied and the results were compared to the estimated data of the right knee joint by OpenPose. This was done on a small window of the dataset of participant 6. The dataset of participant 6 is chosen because in this dataset a lot of misdetections occur, making it valuable for assessing the performance of the filters on a challenging dataset. The right knee is used due to its significant movement in the horizontal (x) direction for when a person is running, plus the right knee is always visible to the camera. In figure 3 in appendix A.2, the x coordinate values of three different joints are displayed, to show the significant movement in the x direction of the right knee joint.

As indicated in the previous section, I will be investigating the following three filters: the Butterworth filter, a band pass filter and the moving average filter. A MATLAB script has been developed for each filter to apply the filter on the x coordinate values of the right knee joint. These scripts allow for the flexibility to modify the window in which the filter should be applied. MATLAB is also used to display the results. Additionally, these scripts are tested on different datasets, to ensure reliability and validity.

The OpenPose estimated data of the right knee joint will henceforth be termed as the original data. In order to distinctly show the difference between the original data and the filtered data, the filters were tested on a small window of a dataset. The window was taken from 42800 to 42900 frames of the dataset of participant 6. Within this window a few misdetections are visible and therefore this is a useful span to see how effective the filters are on the OpenPose estimated data. In the figures 3.1, 3.2 and 3.3, the blue line represents the original data and the orange line represents the filtered data. The y-axis in the graph presents the x-coordinate values of the right knee joint and the x-axis corresponds to the frames in the video.

3.2.1 The Butterworth filter. For this research the Butterworth filter is used as a low-pass filter. This means only low frequencies are being let through. I determined a cut off frequency of 20 Hz using the Fourier Transform. In figure 3.0 in appendix B.1 the frequency spectrum is shown from where I determined the cut off frequency. Additionally, I selected an order of 4 based on a research paper mentioned in the related work. Figure 3.1, displays the results of the Butterworth filter used on the original data, within the specified window of the dataset of participant 6.

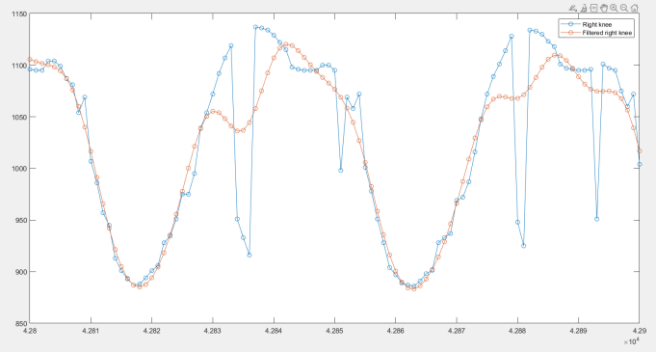

Figure 3.1. The original data vs the Butterworth filter used on the original data within the specified window of participant 6.

It seems like the Butterworth filter smooths out the errors as well. The data points surrounding the specific frame where the error occurs now also deviate from the original data. As a consequence, the validity of the data remains compromised. Which means that this data resulted from the application of the Butterworth filter, still contains errors.

3.2.2 Band pass filter. Next, a band pass filter is used to smooth out the pose estimated data. This band pass filter determines what frequencies should be let through. A cut off frequency of 20 Hz was used again based on the frequency spectrum shown in figure 3.0 in appendix B.1. Next, the total amount of frames of the dataset are divided by this frequency to compute a k-value. This k-value lets the first and last 'k' datapoints through which are the frequencies with the highest amplitude. Finally, the inverse of the Fourier Transform is used to see the results of this band pass filter. Figure 3.2, displays the results of the band pass filter used on the original data within the chosen window of the dataset of participant 6.

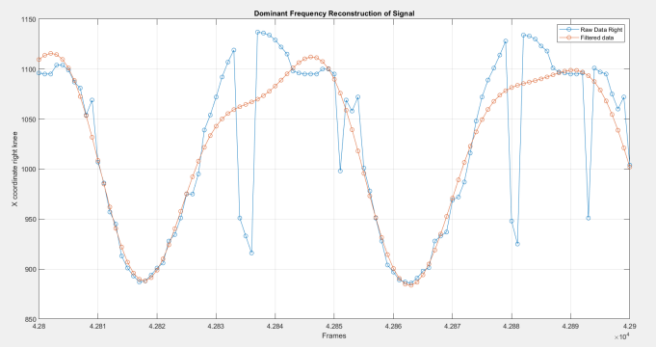

Figure 3.2. The original data vs the band pass filter used on the original data within the specified window of participant 6.

The band pass filter smooths the data pretty well and it reconstructs the datapoints around the misdetections a bit better. In comparison with the Butterworth filter, the filtered data resembles the expected behavior of the right knee's motion more closely. However, the validity of the filtered data is compromised as the curves in the filtered data are too flattened to correspond to the motion of the x coordinate values of the right knee. The bandpass filter outcomes imply that offering athletes advice based on these results may not be valid or reliable.

3.2.3 Moving Average. Finally, a moving average filter is used to reduce noise. This is a simple filter to smooth out data. The moving average uses one variable, the window size. The window size indicates the amount of datapoints that is used to calculate the moving average at any given position in the dataset. A window size of 5 is used to calculate the moving average. Different values were examined for the window size. Using a window size between 5 and 10 causes the valid data points to shift in a way that the filtered data deviates a lot from the expected behavior of the motion of the right knee. While employing a window size between 1 and 5 fails to sufficiently smooth out the misdetections in the filter because the most of the misdetections are still clearly visible. Therefore a window size of 5 was used. Figure 3.3 displays the results of the application of the moving average filter used on the original data within the specified window of the dataset of participant 6.

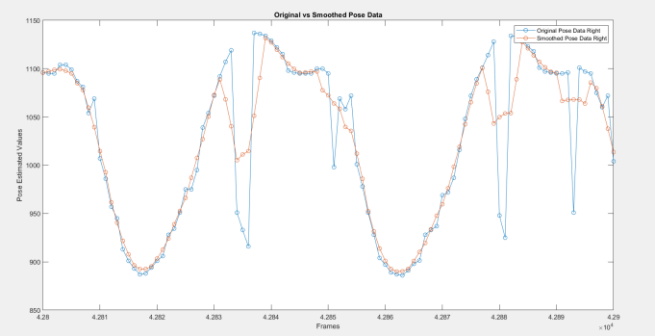

Figure 3.3. The original data vs the moving average filter used on the original data within the specified window of participant 6.

Figure 3.3 shows that the big misdetections are still clearly visible after applying the moving average. Although the figure shows that the moving average fixes small misdetections. Therefore, this filter is not valid enough since the filtered data still contains errors. However, the moving average does correspond to the motion of the right knee.

In conclusion, the results of the application of the three filters are still compromised and therefore not valid for calculating the gait parameters. Either the misdetections are still visible in the filtered data or the filtered data does not follow the expected motion of the right knee anymore. As a consequence, a way of doing error detection is still necessary. In the next subsection, there is looked at how these smoothing techniques still can be used for the purpose of error detection.

3.3 Error detection

3.3.1 Difference. Even though the results of the applicated filters in the previous section are not valid, the smoothing techniques can still be used for the purpose of error detection. After testing and investigating these filters, I looked at the difference between the original data and the filtered data for each filter. This difference should be close to zero if the datapoints are correctly reconstructed by the filter and if there are misdetections the datapoints should deviate from zero. As a consequence, high values in this difference should indicate the misdetections.

The filtered data was subtracted from the original data for the three filters and the difference per filter were compared with each other. To remain consistent, the dataset of participant 6 is used again with the same window from 42800 to 42900 frames. In figure 3.4, the red stems represent the original data minus the filtered data. The top graph represents the difference between the original data and the output of the Butterworth filter, the middle graph represents the difference between the original data and the output of the band pass filter and the bottom graph represents the difference between the original data and the output of the moving average filter.

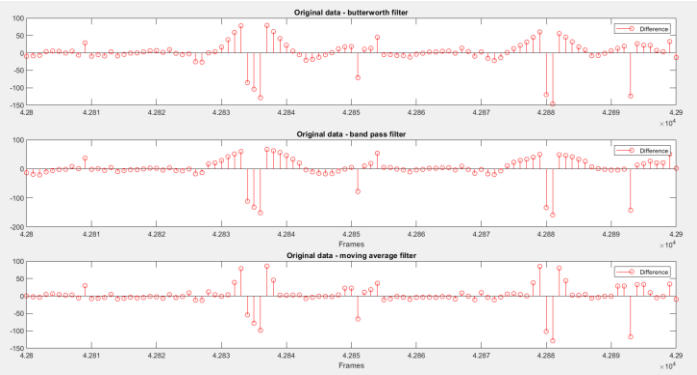

Figure 3.4. the difference between original data and the filtered data for each filter within the specified window.

As is shown in figure 3.4, if the datapoints deviate a lot from zero it actually gives an indication of where a misdetection occurs. The original data (the blue line) in the figures 3.1, 3.2 and 3.3 from the previous section, showed where the misdetections occur within the window. These misdetections stand out in figure 3.4. Therefore, this difference gives an indication of where the misdetections are located. However, the three outputs of the filters display the errors a bit differently from each other.

When looking at the figures, the difference between the Butterworth filter and band pass filter look a lot like each other. However, it seems like the band pass filter smooths out the datapoints around the actual misdetection more than the Butterworth filter does. By looking at the figures 3.1 and 3.2, there is also shown that the band pass filter smooths this area more than the Butterworth filter.

Then there is the difference between the original data and the moving average filter. This difference does not show a lot of deviating datapoints around the actual misdetection. Although, it still detects the errors. This graph also shows that most of the datapoints are closer to zero than the other filter differences. Therefore the difference between the original data and the moving average filter might be the most accurate because it detects less false positive errors than the differences with the other filters.

3.3.2 Thresholds. To make the misdetections easily identifiable, thresholds can be taken on this difference. As a consequence the datapoints that deviate a lot from zero will be detected. Figure 3.5 displays the results of when thresholds are taken on the difference per filter. The same window of 42800 to 42900 frames within the dataset of participant 6 is used again. I used a positive threshold of 20 and a negative threshold of -20. A positive and a negative threshold are used because sometimes the pose estimated data minus the filtered data would result in a negative value. This happens when misdetections are pointed downwards, like the misdetections within this window, instead of upwards. Different thresholds were tested but these thresholds are chosen because higher thresholds would not show all the misdetections anymore and lower thresholds would show more false positive datapoints. The results of the use of higher and lower thresholds are shown in figures 3.5.1, 3.5.2 and 3.5.3 appendix B.2.

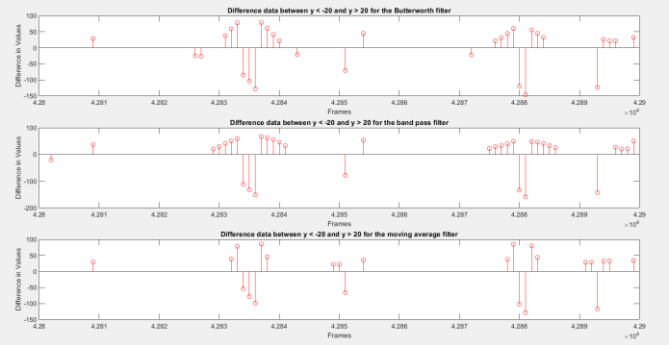

Figure 3.5. The thresholds -20 and 20 used on the original data minus the filtered data within the specified window of the dataset of participant 6 for each filter.

As is shown in figure 3.5, all the filters detect datapoints within this window but the filters detect different datapoints. For example the three swap errors that are happening between 42830 and 42840 also contain some neighbor datapoints for the three filters. The same for the swap errors happening around 42880 frames. In figure 2 appendix A.2 is shown what the different types of errors look like in OpenPose estimated data which can help to classify a misdetection in the estimated data. Noticeable is that every filter detects a few different datapoints within this fragment.

Like is displayed in figure 3.5, the band pass filter and the Butterworth filter contain more neighbor datapoints next to the actual misdetection than the moving average filter. To select the filter that detects misdetections most effectively, it is essential to assess the number of false positives and true positives. The less false positives the more accurate a method is since less instances are incorrectly identified as misdetections. However, the true positives and true negatives are most important because the method should be as accurate as possible.

Since the amount of correct predictions are the most important, the accuracy of the results per filter were calculated. To do so, the amount of true positives, true negatives, false positives and false negatives needed to be counted. A false positive is a datapoint that is detected by the method but is not an actual misdetection and a false negative is not predicted by the method but it was a misdetection. A true positive is a datapoint detected by the method and it is an actual misdetection and a true negative is not detected by the method and it was not a misdetection. I counted them for the window of 42800 to 42900 frames per filter. In table 1, the accuracy is calculated based on confusion matrices per filter for the specified window.

Table 1. the accuracies calculated per filter over the window of 42800 to 42900 frames in the dataset of participant 6.

	Accuracy percentage
Butterworth filter	81.0%
Band pass filter	74.0%
Moving average filter	82.2%

The moving average filter slightly performs better than the Butterworth filter. Noticeable is that for the moving average filter there is a pattern occurring for swap errors. If one or more negative datapoints are enclosed by two positive datapoints in consecutive frames, the negative datapoints are misdetections. The Butterworth filter and the band pass filter do not show a consistent pattern regarding the errors.

The swap pattern showing for the moving average filter also occurs for swap errors in other datasets plus, the pattern is also occurring for other types of errors. Figure 3.6.1 in appendix B.2, demonstrates the detection of the swap pattern within the dataset of participant 4.

Figure 3.6 shows the results of thresholds taken on the difference between the original data and the moving average filter for a time frame of 22700 to 23200 frames for the dataset of participant 7. The two high values at the start are two miss errors and the high value at the end of this window relates to an inversion error. In appendix A.2, figure 2 illustrates the appearance of these errors in the OpenPose estimated data.

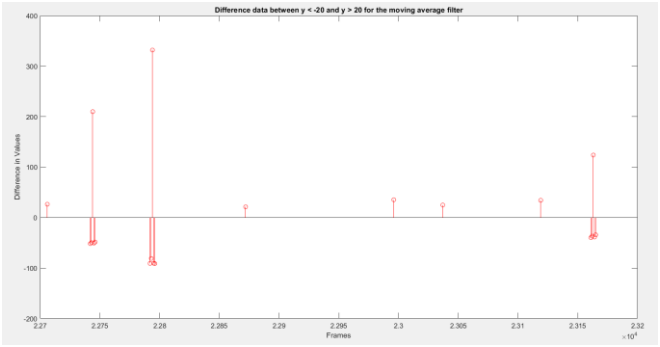

Figure 3.6. The thresholds -20 and 20 used on the original data minus the moving average filtered data for the specified window of the dataset of participant 7.

In figure 3.6 the same pattern, as for the swap errors, occur for the miss and inversion errors. Noticeable is that the pattern occurs when at the misdetection, the difference between the original data and the moving average filtered data is very large. Which is logical because with a window size of 5, the moving average keeps taking an average over five datapoints. If one of the 5 datapoints has a very large value, the average will be taken over the five datapoints and

the neighbor datapoints will deviate more from zero as well. This is why the neighbor datapoints are being detected and why this pattern is five datapoints long.

If these false positive datapoints within this pattern can be removed for the moving average filter, this would leave only the true positives and the accuracy would be even higher. Additionally, this would make the overview of the detected errors more clear and it is easier to identify where the misdetections occur.

3.3.3 Script. MATLAB is used to write the scripts, to remove the false positive neighbor datapoints within the pattern. Like explained before, the pattern has the following sequence: when two positive data points are succeeded by one or more negative data points, and then followed by another sequence of two positive data points, the negative data points represent true positive misdetections, while the positive data points correspond to false positives. This can also happen vice versa for when a misdetection is enclosed by two consecutive negative datapoints. The script should be able to recognize this pattern and then delete the false positive data points but it should also keep the misdetections besides these patterns.

I wrote two scripts in MATLAB. One checks the negative misdetections within the pattern and the other one the positive misdetections with in the pattern. This means the neighbor datapoints within the pattern should be removed and the other misdetections are visible as well. In figure 3.7 the top graph shows the estimated data of the x coordinate values of the right and left knee joint plus the moving average filter used on the right knee joint. The middle graph shows thresholds of 20 and -20 taken on the difference between the original data and the moving average filter. The bottom graph displays the results of the thresholds used together with the scripts to remove the false positives in the window of 42800 to 42900 frames.

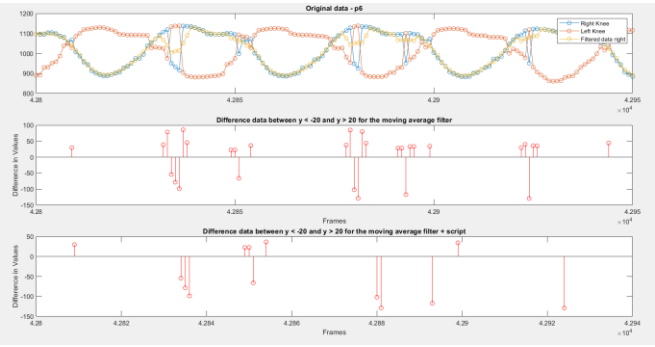

Figure 3.7. the results of the thresholds together with the scripts to remove the false positive datapoints.

As is displayed in the bottom graph, the script with the use of thresholds gets rid of the false positives for the errors. It also shows the other correctly identified misdetections. This window now only contains two false positives and two false negatives which are very good results. The accuracy within the specified window of the dataset of participant 6 has significantly improved reaching 96.0%. This represents a notable enhancement compared to the previous accuracy of 82.2%.

Like is shown in figure 3.7, the method shows promising results and it is detecting the misdetections within the window in the dataset of participant 6.

In the end this error detection approach using smoothing contains the following 4 steps:

1. Determine the window size for the moving average filter
2. Subtract the moving average filter data from the OpenPose estimated data.
3. Determine thresholds to let certain datapoints through.
4. Run the remaining datapoints through the scripts to remove false positives within the swap error pattern.

The performance of this approach is evaluated in the next section.

4. EVALUATION

The third goal of this research was to evaluate the results of the method to assess its performance. To do so, the method is tested on different windows within the datasets of participant 6 and participant 9. To show how accurate the method is, a performance evaluation based on confusion matrices was done on the datasets. Additionally, per type of error, true positives were counted and evaluated. This in regards to future work to optimize the method because it is not able to detect all the misdetections yet. Since, the dataset of participant 6 is a challenging dataset, testing the method on this dataset, would assess the method's robustness and sensitivity. The dataset of participant 9 is also evaluated because it has different types of errors than the dataset of participant 6.

4.1 Overall model performance

The performance evaluation method used was calculating metrics based on confusion matrices. Three windows were selected within the dataset of participant 6 and the dataset of participant 9. For the dataset of participant 6 the following windows were selected: 3750-4000 frames, 42750-43000 frames and 81250-81500. For the dataset of participant 9 the following windows were selected: 1000-1500 frames, 18200-18700 frames and 41100-41600 frames. These windows were selected because they contain all the different types of errors. For each window within these datasets the true positives, false positives, true negatives and false negatives were counted and summed up into one confusion matrix per dataset. Based on this confusion matrix, the precision, accuracy, recall and the f1-score were calculated for the two datasets. Table 2 shows the calculated metrics for the datasets of participant 6 and 9.

Table 2. results of the performance metrics calculated for the dataset of participant 6 and the dataset of participant 9.

	Participant 6	Participant 9
Accuracy	0.98	0.99
Precision	0.85	1.00
F1 score	0.80	0.65

The accuracy is rounded 98% for participant 6 and 99% for participant 9. These values conclude the method has a very high ratio of correct predictions across both the datasets. Since true positives are crucial for my method, the precision was also

calculated. The precision indicates that when the model predicts a misdetection, there is a certain chance this is an actual misdetection. A precision of 85% was calculated for the dataset of participant 6 and a precision of 100% was calculated for the dataset of participant 9. Lastly, the F1 score was calculated. The F1 score provides the balance between the precision and the recall, where a higher score provides a better performance. The F1-score of 0.80 for the dataset of participant 6, indicates a well-balanced model, so the method effectively identifies a substantial portion of actual positive instances while maintaining high precision in positive predictions. However, the F1-score for the dataset of participant 9 is only 0.65 which means this is a worse balanced model than the model of the dataset of participant 6.

4.2 Error type evaluation

Since, the method is not able to detect all the types of errors yet an evaluation was conducted to see what types of errors the method excels at identifying and those where its performance is less effective. Regarding future work this can be used to optimize the method so it will be capable of detecting all the misdetections.

For the same three windows in the dataset of participant 6 and the dataset of participant 9, per error type the amount of misdetections the method predicted were counted and the actual amount of misdetections were counted. The misdetections are labeled as the following:

1. Miss
2. Jitter
3. Small swap
4. Big swap
5. Small inversion
6. Big inversion

The swaps and inversions are split into small and big errors because sometimes their ranges differ significantly from each other. In figure 2 in appendix A.2, is displayed what these misdetections look like in the OpenPose data and how they differ from each other. The datapoint that is brushed red in this figure, indicates the actual misdetection.

Table 3 shows a table with the misdetections counted per error type and summed up for the three windows of the dataset of participant 6. Table 4 shows a table with the misdetections counted per error type and summed up for the three windows of the dataset of participant 9.

Table 3. the counted misdetections per error type for the dataset of participant 6.

	actual	method	percentage
Miss	0	0	100%
Jitter	2	0	0%
Small swap	13	8	62%
Big swap	19	19	100%
Small inversion	11	7	64%
Big inversion	1	1	100%
Total	46	35	76%

Table 4. the counted misdetections per error type for the dataset of participant 9.

	actual	method	percentage
Miss	5	5	100%
Jitter	0	0	0%
Small swap	11	4	36%
Big swap	3	3	100%
Small inversion	10	1	10%
Big inversion	1	1	100%
Total	30	14	47%

The method fails to detect all of the small swaps and small inversions for both datasets. The same for the jitter errors. These errors are small and therefore do not lie within the thresholds, as a consequence they are not detected by the method. In the dataset of participant 6 the small errors were detected better than in the dataset of participant 9. Although, the method does successfully detect all of the big swaps and inversions for both datasets. This is because these errors have a larger range and lie within the thresholds.

For the overall performance of the method, metrics were calculated based on confusion matrices for the datasets of participant 6 and participant 9. An accuracy of 98%, a precision of 85% and a F1 score of 0.80 was calculated for the dataset of participant 6. An accuracy of 99%, a precision of 100% and a F1 score of 0.65 was calculated for the dataset of participant 9. Besides, there was investigated what types of errors the method can detect more effectively and which types not. All of the miss errors, big swap errors and big inversion errors were successfully detected for both datasets but the method fails to detect all the small swaps, small inversions and jitters. The detection percentages for small swaps and small inversions remained above 60% in the dataset of participant 6. However, for participant 9, the detection of small errors was even less accurate.

5. DISCUSSION

In this section the results of the method are being discussed since the method still has its limitations even though it performs well.

The method is tested on different datasets, and it seems to work on all of them but there are still a lot of exceptions that happen. Datapoints can lie outside the thresholds so they do not pop up as misdetections or the pattern is not complete so the scripts do not remove false positives. For example, if there are many misdetections happening in a short time frame, the pattern does not show up and it keeps the false positives. In appendix B.2, figure 5 provides examples within a small window of the dataset of participant 1, illustrating instances where misdetections occur. Notably, the scripts are not effectively removing false positives in these cases.

Besides, the method might not be reliable if it is used on another body joint than the right knee. Other body joints might have a lower or higher range in their values and therefore the method needs different thresholds to detect the errors. However, the thresholds can be adapted according to the dataset.

Then there is the pattern. This pattern was discovered in the x coordinate values of the right knee but this pattern might not be there in the data of other joints. This can be a positive thing because the method can just detect the errors based on thresholds or there is another pattern that cannot be recognized by the scripts.

Another limitation is that the performance evaluation is done just on two datasets. This is not enough evidence to proof that the method works well on other datasets.

6. CONCLUSIONS

The first goal of this research is to investigate the results of the application of smoothing techniques on the OpenPose estimated running data for the purpose of error detection. The Butterworth filter, a band pass filter and the moving average filter were tested on a window in the dataset of participant 6. As the results still included misdetections, there was explored how the application of smoothing techniques can still be used for error detection. This exploration aims to provide accurate advice to athletes based on the estimated data.

The second goal of this research was to use smoothing techniques to detect errors in the OpenPose estimated running data. Performance metrics were calculated to say that the moving average performed best. Therefore, a method was developed using the moving average filter to detect errors in OpenPose estimated data. This approach contains the following steps. Firstly, a window size is taken for the moving average filter. Then this filtered data is subtracted from the OpenPose estimated data. The next step is taking thresholds on this difference to easily identify the misdetections. Lastly, the remaining datapoints should be ran through MATLAB scripts to remove false positives within a pattern.

The third and last goal of this research was to evaluate the results of the method and assess its performance. A performance evaluation was done for the method on two datasets. The method has an accuracy of 98% for the dataset of participant 6 and an accuracy of 99% for the dataset of participant 9. The method is able to detect all the miss, big swap and big inversion errors but not the small swap, small inversion and jitter errors within the small windows in these two datasets. The method has more limitations as is stated in the discussion.

7. FUTURE WORK

This method to detect misdetections is just the start of the end goal to fix the errors. To actually fix the errors, the detected errors should be characterized in the four types of errors: miss, jitter, swap and inversion. Then for example when a swap error occurs in the x coordinate value of the right knee, the datapoint can be swapped with the x coordinate value of the left knee. The other types also have their own properties and in this way it is possible to reconstruct them. For instance, by trying different thresholds it might be possible to detect different types of errors. Miss errors are clearly larger than the other errors. However, the other types of errors are harder to identify, since the range of those errors are more inconsistent. So for future work this issue should be investigated further.

Lastly, the method is not perfect because it does not detect all the misdetections yet. The method should also be able to detect the jitters, small swaps and small inversions. Thresholds can be explored further since the thresholds determine what misdetections are being let through. Plus, maybe there are even more patterns within the detected errors which I did not discover yet.

REFERENCES

[1] H. G. S. T. W. S. S. Y. Cao Z, "OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields," p. 14, 2018.

[2] B. C. H. A. L. W. Gauss J, "Smoothing Skeleton Avatar Visualizations Using Signal Processing Technology," *SN Computer Science,* p. 17, 2021.

[3] J. S. Q. L. S. P. Jianhang He, "NRPose: Towards noise resistance for multi-person pose estimation," *Pattern Recognition,* 2023.

[4] R. C. R. R. Stenum J, "Two-dimensional video-based analysis of human gait using pose estimation," *PLoS Computational Biology,* vol. 17, no. 4, 2021.

[5] N. Y. M. T. I. A. T. T. Sato K, "Quantifying normal and parkinsonian gait features from home movies: Practical application of a deep learning–based 2D pose," *PLoS ONE,* vol. 14, no. 11, 2019.

[6] S. T. U. K. O. L. K. A. I. Y. F. S. Y. S. Nakano N, "Evaluation of 3D Markerless Motion Capture Accuracy Using OpenPose With Multiple Video Cameras," *Frontiers in Sports and Active Living,* vol. 2, 2020.

[7] D. A. WINTER, "Biomechanics and motor control of human movement," in *Biomechanics and motor control of human movement,* Waterloo, Wiley, 2009, pp. 77-88.

[8] T. H. H. T. K. T. O. Y. Y. M. I. N. Ota M, "Verification of reliability and validity of motion analysis systems during bilateral squat using human pose tracking algorithm," *Gait and Posture,* vol. 80, pp. 62-67, 2020.

[9] P. P. M.R. Ronchi, "Benchmarking and Error Diagnosis in Multi-Instance Pose Estimation," 2017.

APPENDIX A.1

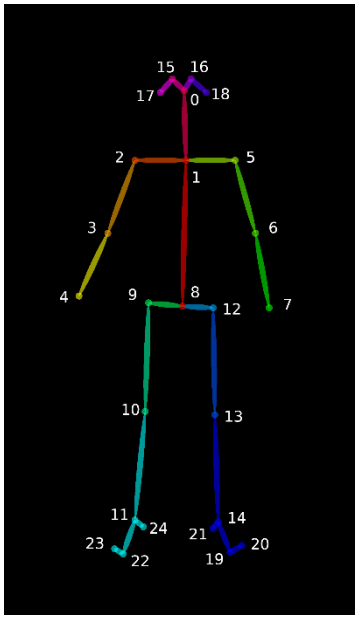

Figure 1. the 25 key body points tracked by OpenPose.
https://github.com/CMU-Perceptual-Computing-Lab/openpose/blob/master/doc/02_output.md

APPENDIX A.2

Figure 1.1. two consecutive frames of a jitter error happening.

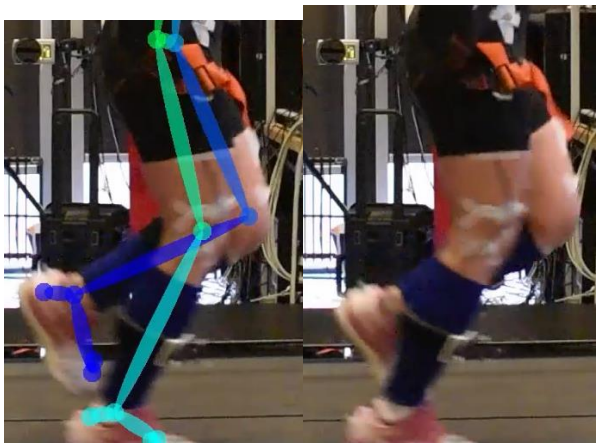

Figure 1.2. Two consecutive frames of a miss error happening.

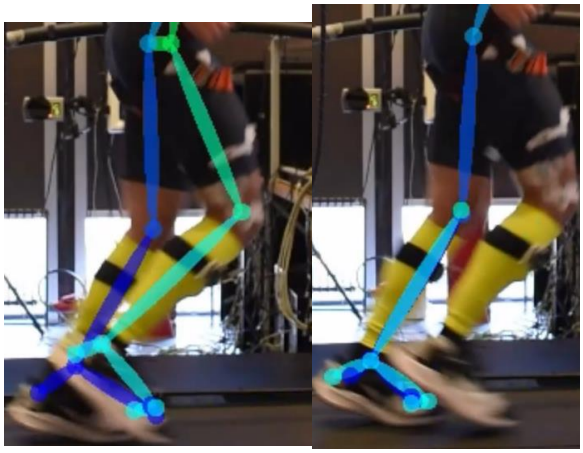

Figure 1.3. two consecutive frames of an inversion error happening.

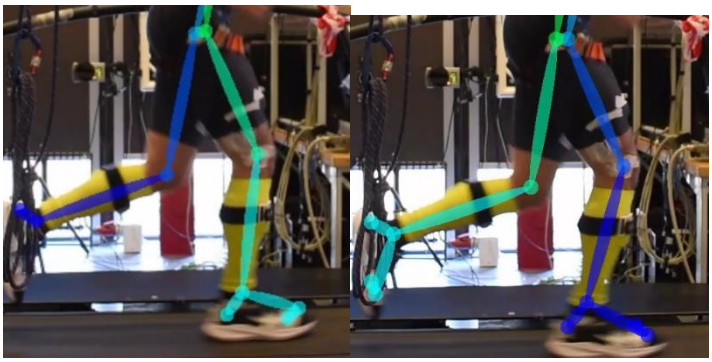

Figure 1.4. Two consecutive frames of a swap error happening

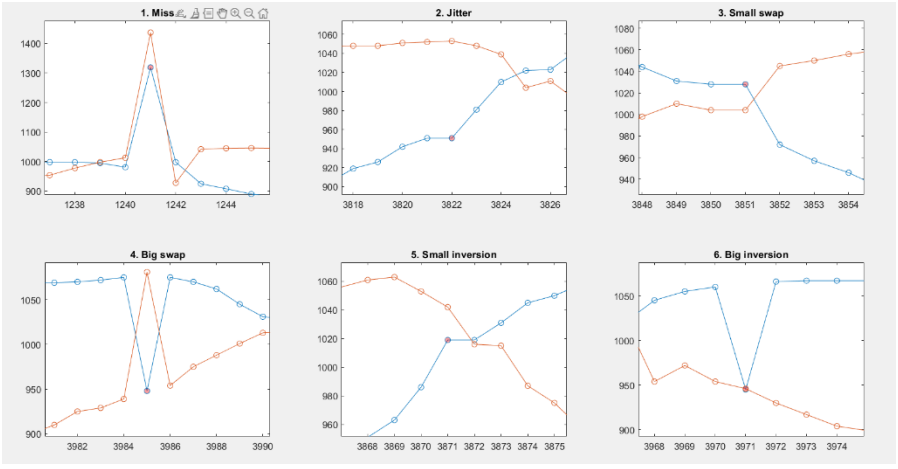

Figure 2. different types of errors in OpenPose estimated running data.

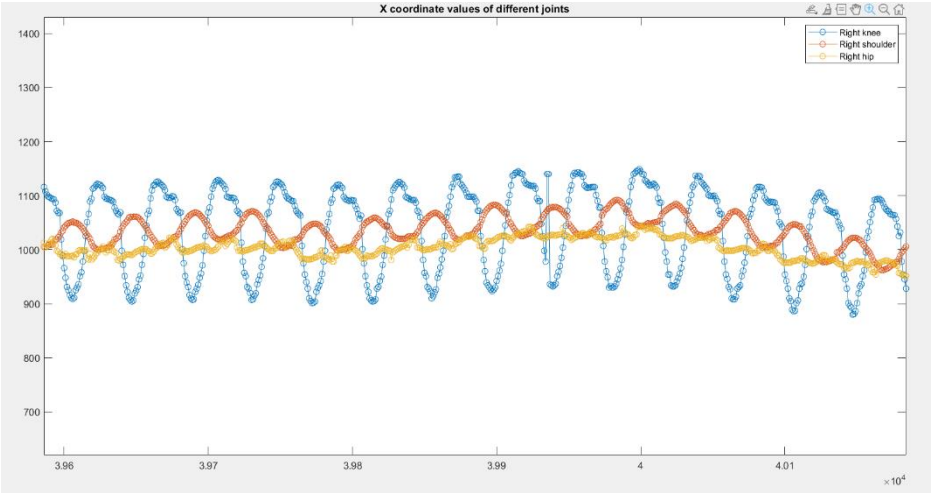

Figure 3. x coordinate values of the right knee, right shoulder and right hip joints estimated by OpenPose.

APPENDIX B.1

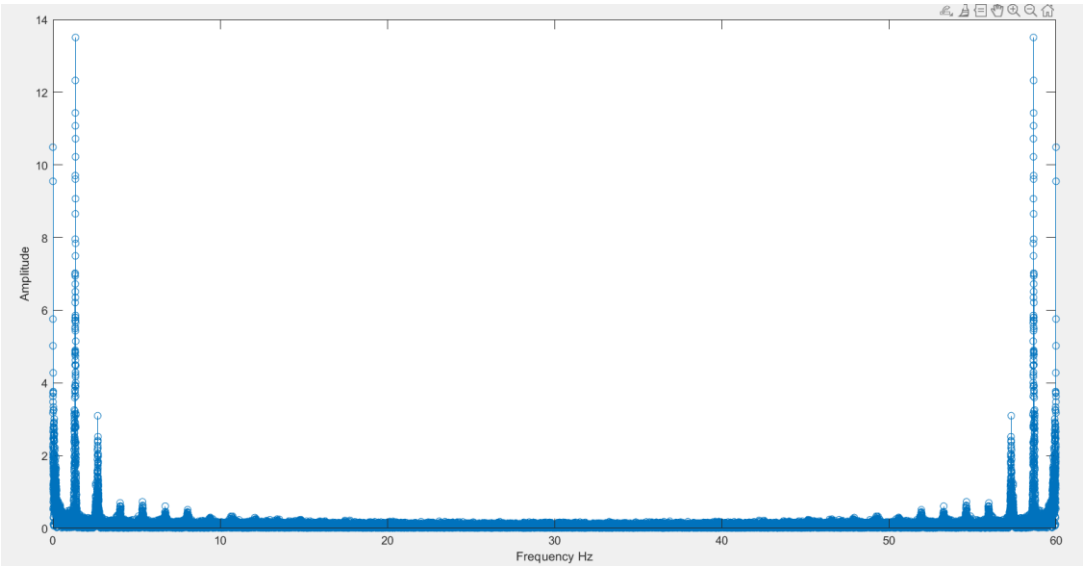

Figure 3.0. The frequency spectrum of the dataset of participant 6.

APPENDIX B.2

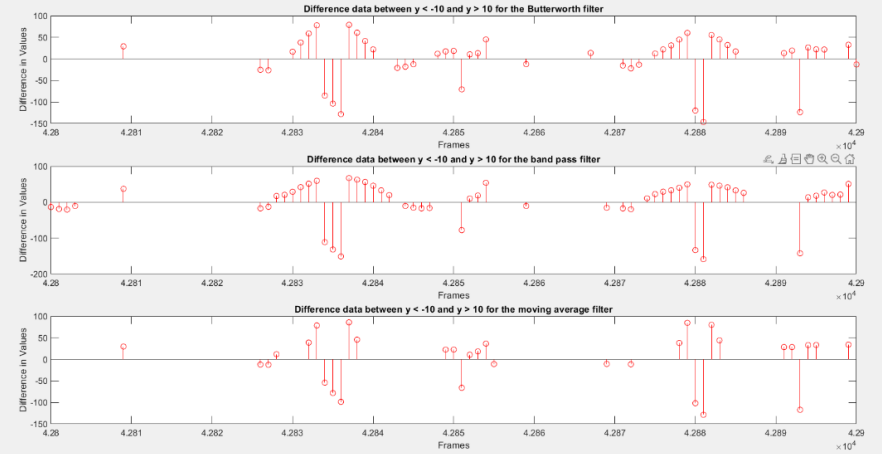

Figure 3.5.1. thresholds of -10 and 10 taken on the difference between the original data and the filtered data for each filter on a window of the dataset of participant 6.

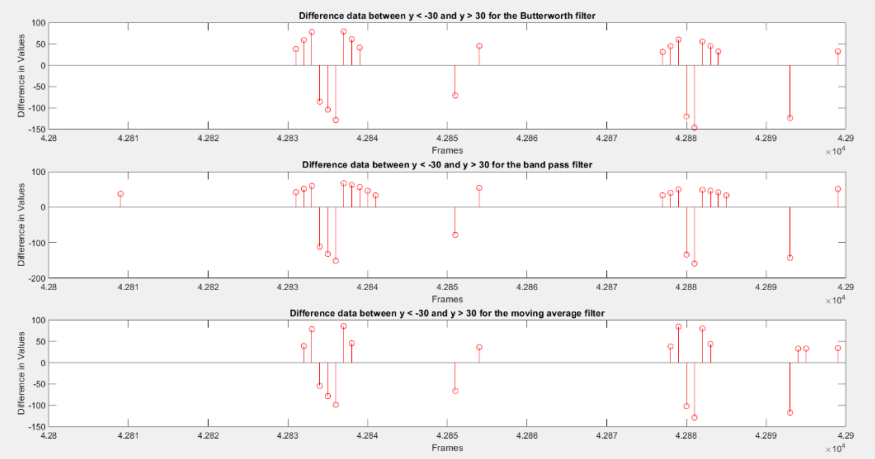

Figure 3.5.2. thresholds of -30 and 30 taken on the difference between the original data and the filtered data for each filter on a window of the dataset of participant 6.

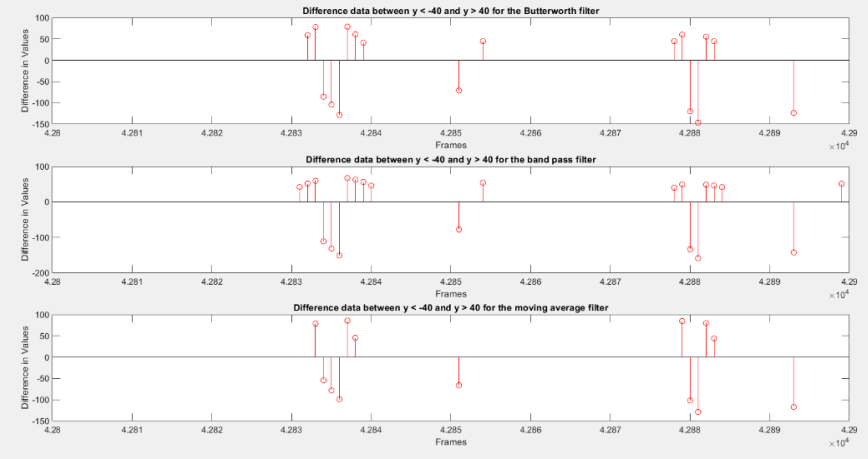

Figure 3.5.3. thresholds of -40 and 40 taken on the difference between the original data and the filtered data for each filter on a window of the dataset of participant 6.

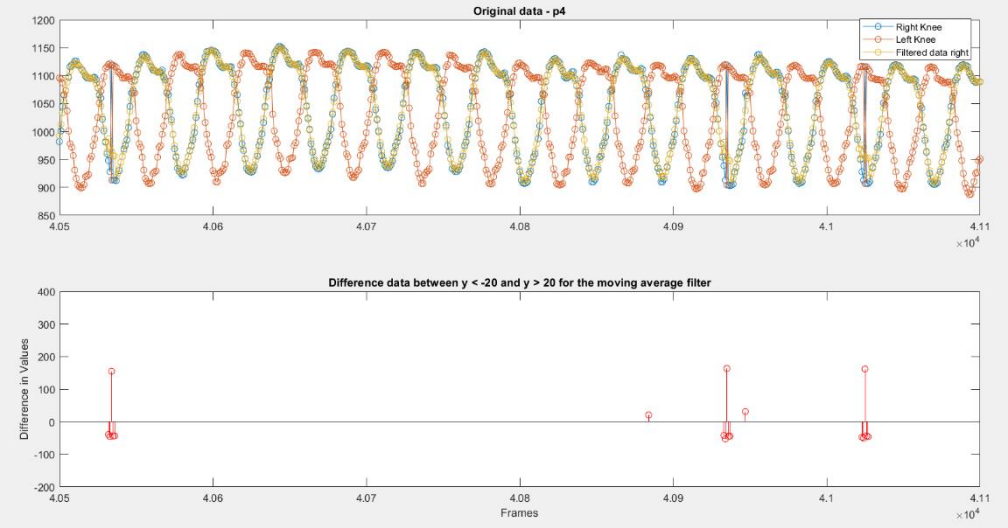

Figure 3.6.1. the results of the thresholds -20 and 20 taken on the original data and the moving average filter on a window of the dataset of participant 4.

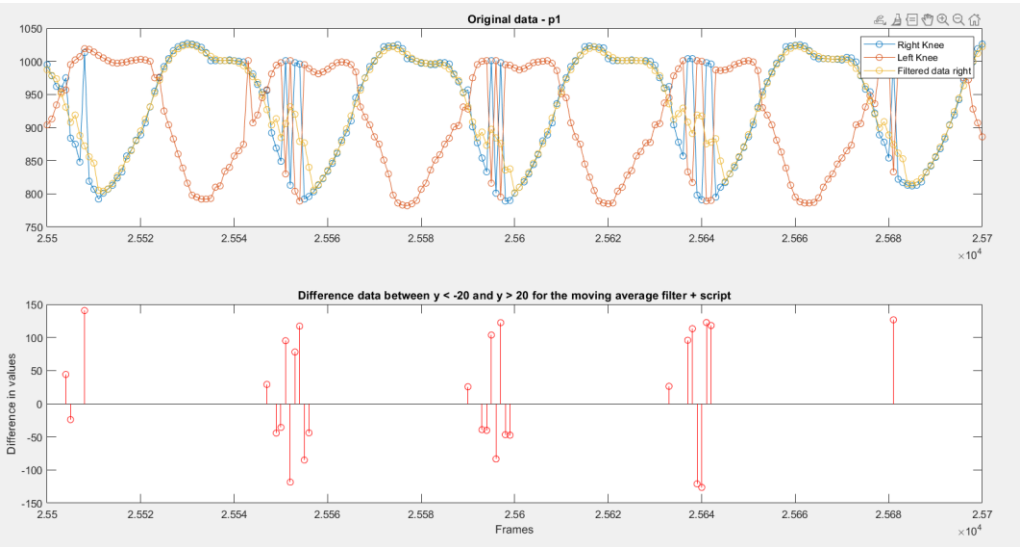

Figure 5. the results of the thresholds taken on the difference together with the scripts between the original data and the moving average filter on a small window of the dataset of participant 1.