Automatic processing of motion capture data

R.J. Schuddeboom, 3508552

1. Introduction

Despite the extensive use of motion capture systems, direct use of the data they produce is often prevented by a number of problems. First of all, due to hardware properties of the sensors used for motion capture, the signals that are measured will often be noisy. Noise may also be caused by the shifting of markers used for motion capture. Markers may shift with respect to clothing, which may shift with respect to the skin, which in turn may shift with respect to the underlying skeleton, to which recorded data is usually mapped.

Furthermore, in optical motion capture systems, where body movement is recorded by determining the 3D position of reflective markers with multiple cameras, a series of other problems may occur. Markers may be occluded for a period of time, by other markers, body parts, or objects, leading to gaps in the data. Also, if passive markers are used (markers that are not uniquely distinguishable), it may be unclear that when a marker reappears, this is actually the same marker that had disappeared before. Determining marker identities may also pose a problem as markers come too close, in which case their identities may be mixed up by the system. A final problem is the appearance of ghost markers, recording of reflections in the environment that do not belong to any of the markers.

Various methods have been developed to deal with one or some of these problems in an automated fashion, reducing the need for manual processing. Some of these methods can operate in real-time, while others are a means of post-processing. They also differ with respect to the use of an underlying skeleton to determine motion constraints, which may be non-existent, predefined, or estimated as part of the method.

In this report, the advantages and disadvantages of various methods will be discussed and an effort shall be made to combine a number of methods into a pipeline that is able to deal with most of the problems mentioned above, allowing it to process raw motion capture data.

2. Related work

There is a variety of research on automated processing of motion capture data, focused on a different subset of the problems discussed earlier.

Aristidou et al. [2] present a method for dealing with marker occlusion in real-time. It combines the use of the Kalman filter with inferred information from neighboring markers. No knowledge of a predefined skeleton is used, although the method requires three markers to be placed on each rigid bodypart, opposed to the other methods discussed next and the methods presented in this report.

Van Rhijn et al. [3] discuss a method that does not put such constraints on the amount of markers per body part. Skeletal constraints are determined in a so-called “model estimation phase”, a pre-processing step in which a set of body parts is put through all possible motions to determine the movement constraints imposed by the joints.

Sul et al. [4] present a method serving three purposes: (i) smoothing the jerky motion due to the sensor noise, (ii) satisfying the kinematic constraints of the human body, and (iii) generating a seamless motion transition between motion segments. This last part suggests it may also be useful as a post processing method for gap filling. They developed a method based on the Kalman filter approach that handles these problems in a single, unified framework.

Piazza et al. [5] have developed a real-time extrapolation algorithm that deals with missing markers and does not require any statistical data (in contrast to methods using the Kalman filter). It predicts the marker position based on previous position and velocity, using the assumption that movement consists of linear and circular movement. This prediction is corrected using a constraint matrix, storing rigid marker distance relations.
One of the issues that remains unaddressed in these papers is the problem of marker identification, either due to the assumption markers are uniquely identifiable or due to the restricted scope of the research.

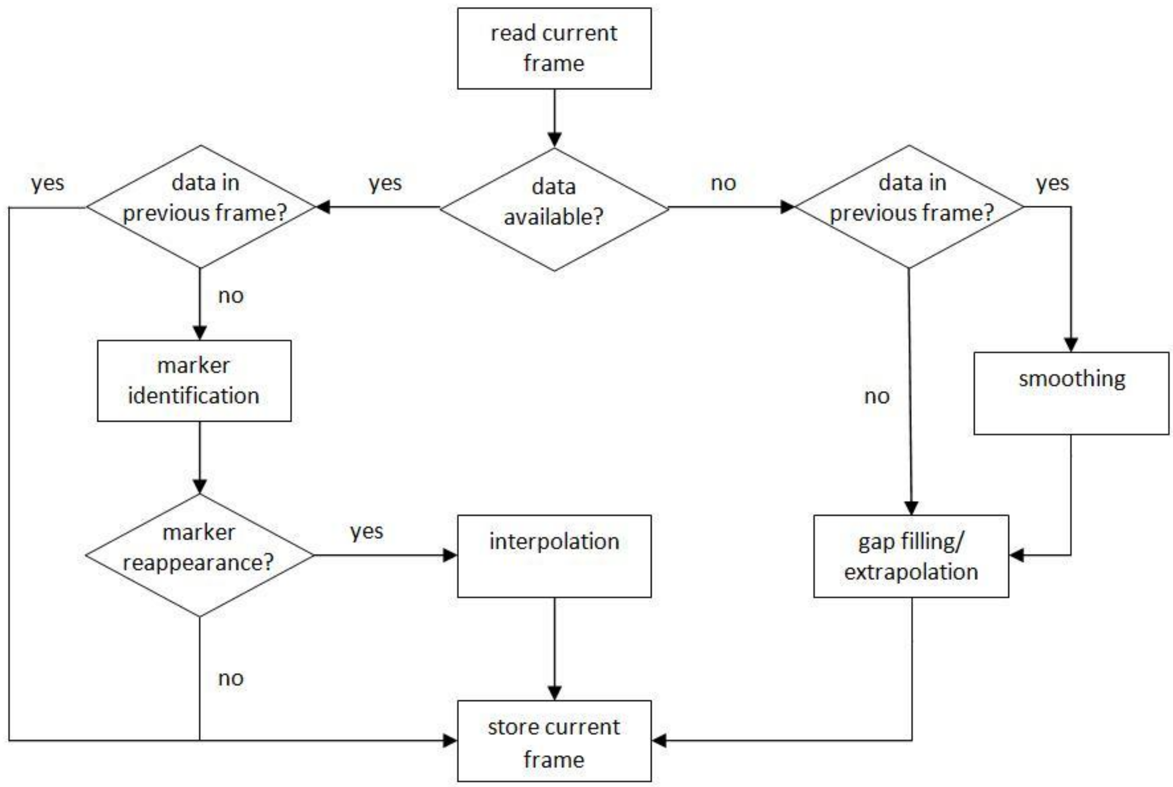
3. Pipelines & report overview

In this report, the essential steps of both real-time and post-processing pipelines for automatic processing of motion capture data are discussed. Based on the problems discussed in Section 1, the following processes, that may serve as pipeline components, can be identified:

- Smoothing: noise can be removed from available data segments using smoothing methods. This can be done to improve quality of playback at a later stage, or to obtain a more accurate estimation of position and velocity of a marker, which may allow for more accurate estimation of its position during periods of marker occlusion.
- Gap filling: as markers go missing, data gaps can be filled using extrapolation methods, predicting their positions based on previous data. In real-time applications, extrapolation should be used for all missing markers, in order to be able to display an estimate of their position during the current frame. During post-processing, gap filling may also be done using interpolation methods, allowing for the use of data both before and after the gap.
- Marker identification: marker identification is an issue for systems using passive markers. If a new data segment appears, checks should be made to see if it might belong to a marker that has disappeared at an earlier stage. Also, problems such as marker switches and the appearance of ghost markers might be resolved here, but these issues will remain largely unaddressed during this research.
- Interpolation: once a marker reappears, its predicted (extrapolated) value may differ from the actual data. In real-time processing, interpolation may be applied to create a smooth transition between predicted and actual position over the next couple of frames, reducing any visual distortions as the marker is displayed.

Figure 1 provides a decision chart illustrating how these processes may be combined into a pipeline. Both in real-time processing and post-processing, a frame by frame processing method may be followed.
The first step consists of obtaining the data that is available for the current frame. If data is available for a certain marker, and was available in the previous frame, the data for the current frame can be stored as it is. If no data was available in the previous frame, the identity of the marker to which the new data belongs is unknown, which means marker identification should take place. Based on the result of marker identification, the data can either be labeled as the reappearance of a missing marker, or as a new marker that has appeared for the first time during this frame.
If reappearance did occur, predicted position in the previous frame is likely to differ from the new positional data in the current frame, due to prediction inaccuracy. In a real-time pipeline, interpolation may be started for the next couple of frames, smoothing out the transition over a number of frames. In a post-processing pipeline, there is no need for this kind of interpolation, since data doesn’t have to be displayed for the current frame. Instead, the interpolation step may be used to fill the gap in the previous frames, now that the position of reappearance is known. After any possible interpolation, the data for the current frame can be stored.

Now, to be able to perform marker identification, a predicted position should be maintained and updated for each marker that goes missing. This process is depicted on the right side of Figure 1. If no data is available for a marker in the current frame, a check should be made to see if this is the first frame for which data is missing. If so, data in previous frames may be smoothed in order to obtain a more accurate estimation of the last known position and velocity of the marker, which could improve prediction results. As long as a marker remains missing, the “gap” in the data should be filled using extrapolation. The extrapolated position of a missing marker can now be stored for the current frame.



The remainder of this report is divided as follows: In Section 4, smoothing methods are discussed, followed by various methods that can be used for gap filling in Section 5. In section 6, the problem of marker identification is addressed and in section 7, results of adding neighboring marker constraint to gap filling methods and marker identification methods will be presented. Suitability the methods presented in each section as part of a real-time or post-processing pipeline will also be discussed.

4. Smoothing

Motion capture data is noisy for various reasons: the signals of the measurement equipment may be subject to noise, the clothing to which markers are attached may shift on the skin and the skin may shift with respect to the underlying skeleton.
There are various methods for filtering data in order to reduce such noise. This discussion will focus on Butterworth filters and the Kalman smoother filter.

4.1 Butterworth lowpass

The Butterworth lowpass filter [1] is used to remove high frequency behavior from a data signal. This can be useful for processing motion capture data, since signal fluctuations with very high frequencies are unlikely to correspond to natural human motion. (A fluctuation occurring over 4 frames in a signal captured at 200 frames/second would have to correspond to a movement taking no more than 0.02 seconds.) For this reason, it is safe to assume such signal changes are noise and may be removed.

Figure 1 illustrates the effect of the lowpass filter for various cutoff frequencies. Behavior with higher frequencies will be removed, while behavior with a lower frequency will be maintained. Based on experimentation, a cutoff frequency of 4 or 5 is recommended for this particular motion capture system, which captures data at 200 frames per second.

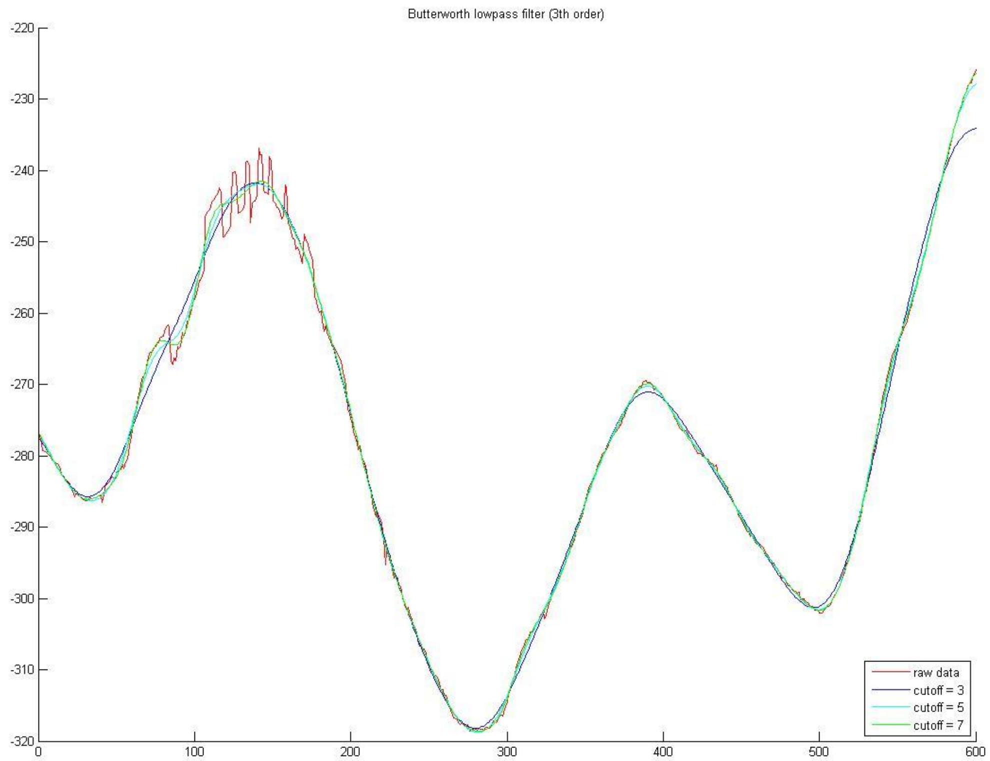


Figure 1. Butterworth filtering using various cutoff frequencies.

Figure 2 illustrates the differences of lowpass filters of various orders. These seem to be minimal, but for the second order filter, a deviation can be observed at some of the local maxima and minima. For this reason, the use of a third or fourth order filter is recommended.

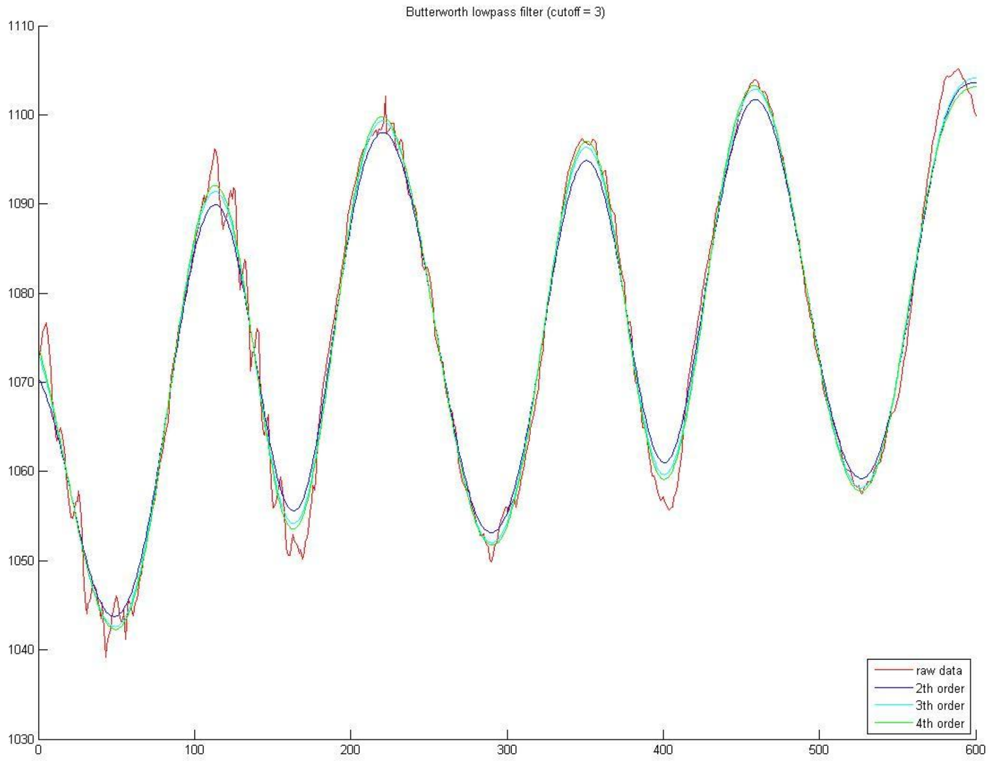


Figure 2. Butterworth lowpass filtering using second, third and fourth order filters.

By examining Figure 1 more closely, a drawback of the lowpass filter became evident. As the data set comes to an end, the smoothed data seemed to deviate from the real data. Further examination of this phenomena confirmed that this was indeed the case, demonstrating an increased inaccuracy for lower cutoff frequencies. Figures 3 and 4 illustrate this by showing the result of smoothing a subset of the dataset presented in Figure 1, showing deviations at the end of the subsets that do not appear in Figure 1.

Although this may not pose a problem to post-processing methods of data sets with a number of redundant frames at the end, it does pose a problem to real-time processing of a signal, where smoothing actually has to be done for these last couple of frames (and the last frame in particular, which needs to be displayed at that point in time). It also poses a problem to data sets which contain gaps, unless gap filling can be done before smoothing. However, in order to apply gap filling, smoothing may be a useful preprocessing step, providing a more accurate starting point and direction for extrapolation or interpolation.

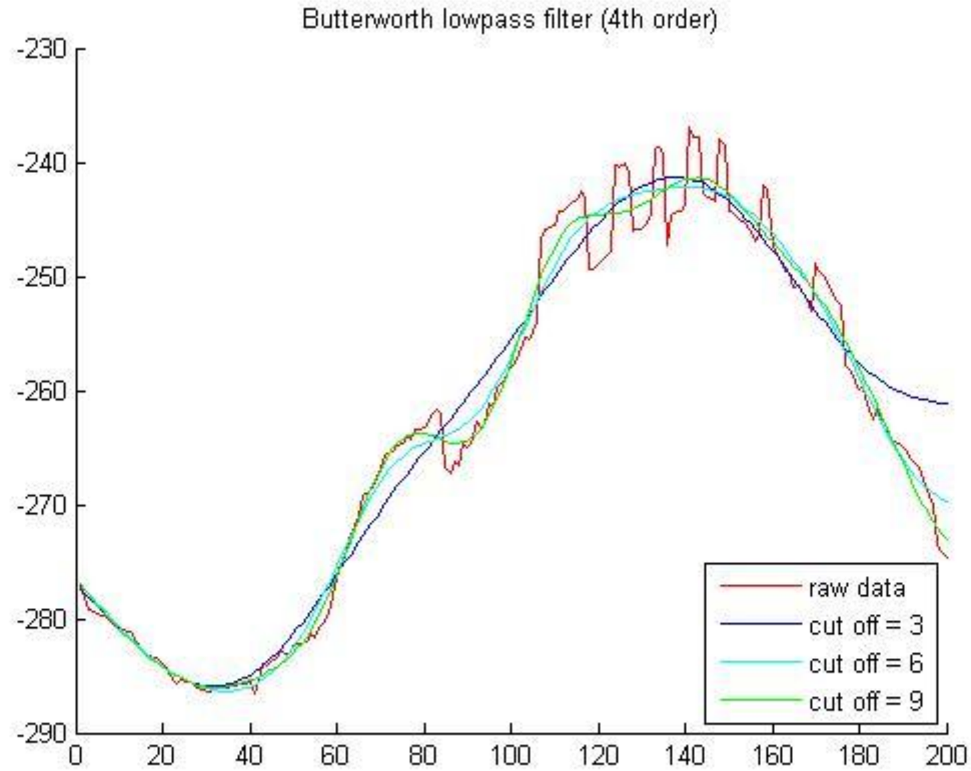


Figure 3. Smoothing the first 200 frames of the dataset presented in Figure 1, showing deviations at the end of the subset that do not appear in Figure 1.

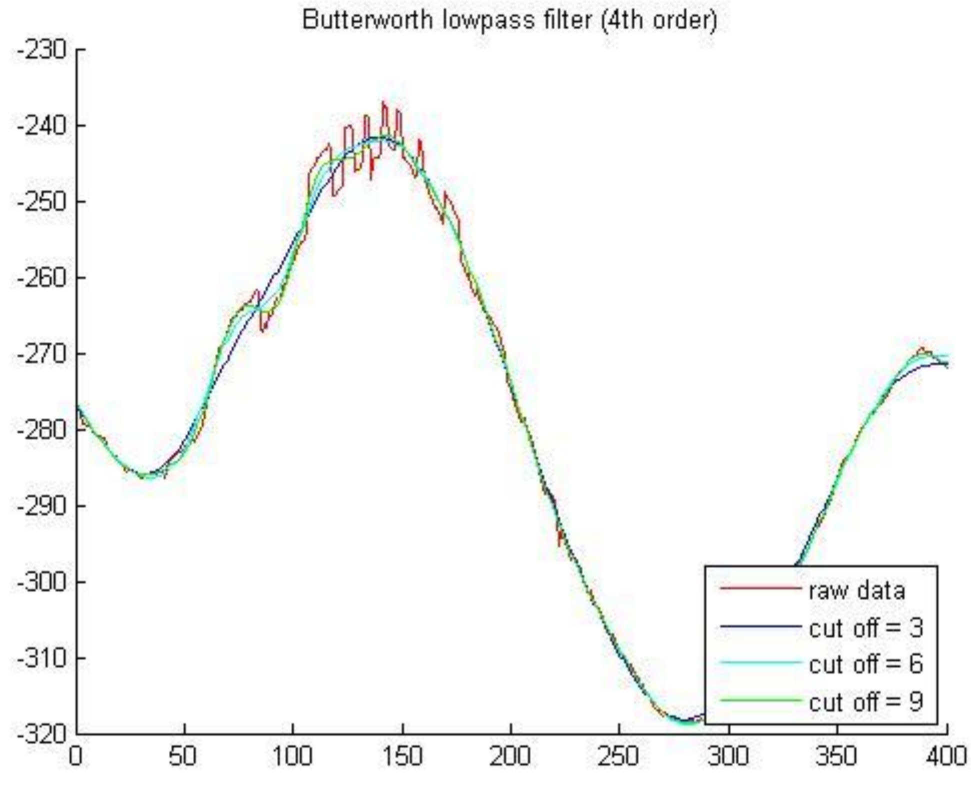


Figure 4. Smoothing the first 400 frames of the dataset presented in Figure 1, showing deviations at the end of the subset that do not appear in Figure 1.

4.2 Butterworth highpass

The Butterworth highpass filter is the opposite of the lowpass filter. It preserves behavior with frequencies above the cutoff frequency, while it removes low frequency behavior. It may be used to examine the noise in a signal more closely. Figure 5 gives an example of the application of a highpass filter.

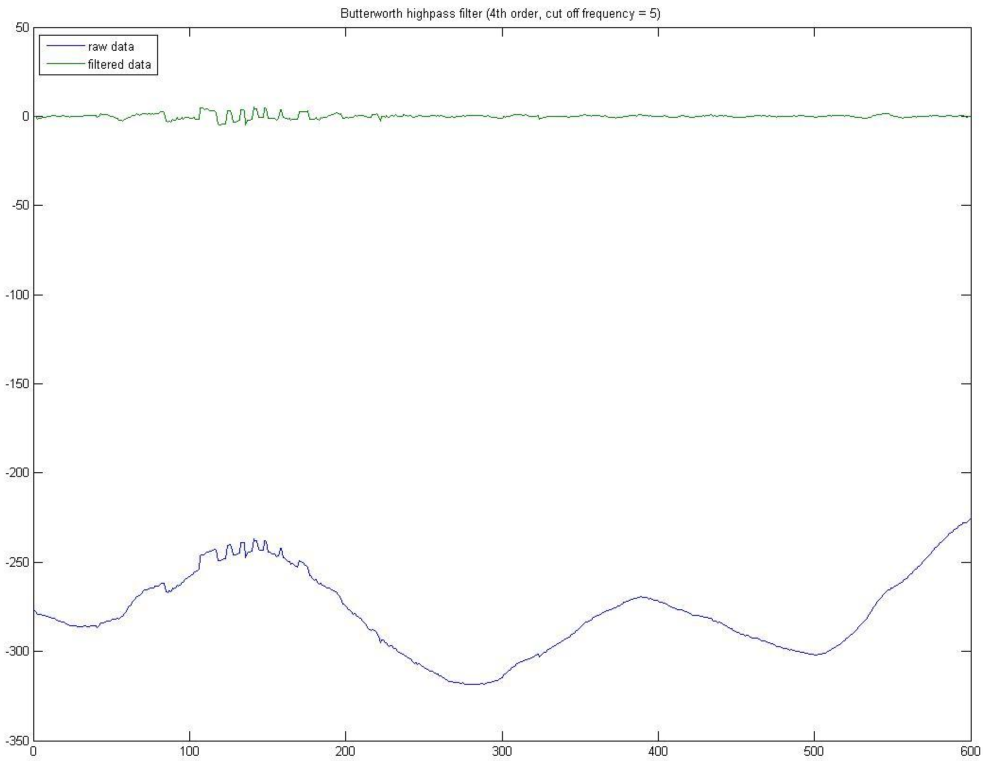


Figure 5. Butterworth highpass filtering, useful for gaining insight in the amount of noise in the data.

4.3 Kalman filter

The Kalman filter is a filter that is mentioned often in relation to processing motion capture data [1,3]. There exist a number of variations on the Kalman filter, such as the extended Kalman filter and the Kalman smoother filter. During this research, an implementation of the smoother Kalman filter by Murphy [6] was used.
Figure 6 and 7 illustrate the application of the smoother Kalman filter on a set of motion capture data. Figure 6 shows the result of smoothing using Kalman smoother filters of second, third and forth order. It illustrates lower orders have a stronger smoothing effect, filtering out more of the high frequency behavior. It also shows some inaccuracies of the second order filter at the local maxima and minima in the plot. Based on these observations, third order filtering is recommended for the use of this particular implementation of the filter on data from this particular motion capture system.
Figure 5 illustrates the differences between different values of experimental noise in the filtering, a value of 10e7 is recommended, which seems to generate an appropriate level of smoothness.

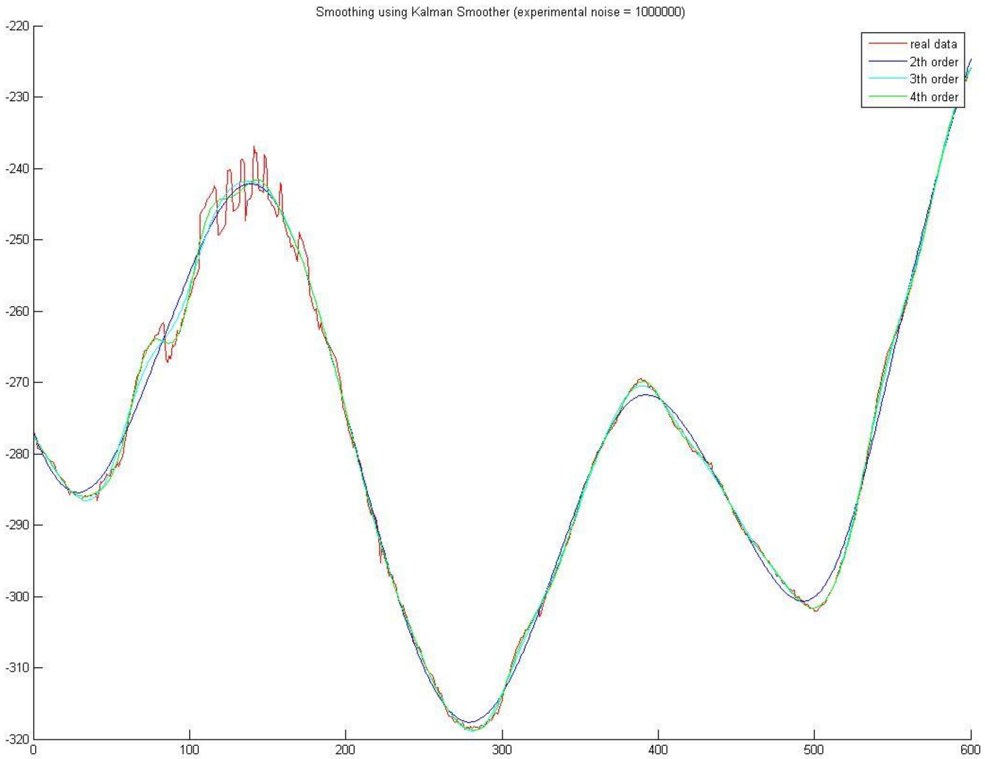


Figure 6. Kalman smoothing using second, third and fourth order filters.

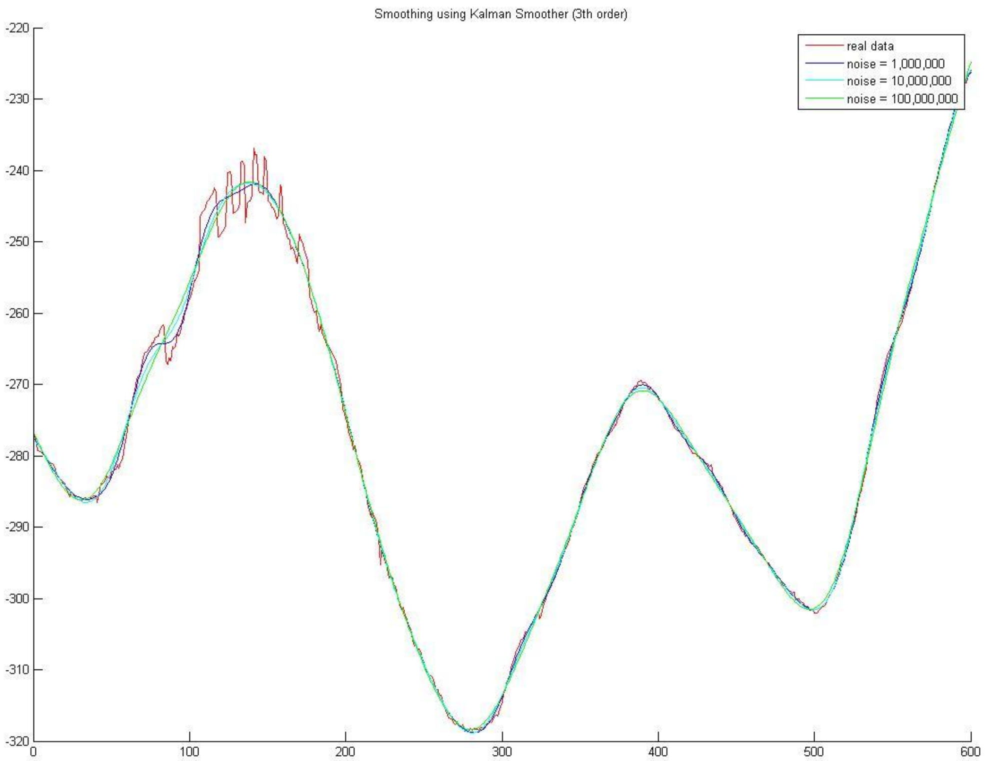


Figure 7. Kalman smoothing using different amounts of experimental noise.

Unlike the lowpass filter, the Kalman filter does not cause inaccuracies at the end of a data set. This makes it more suitable for application in real-time processing and gap filling methods.

4.3.1 Minimal amount of frames

It is also useful to determine the minimal amount of data frames for which smoothing becomes useful for prediction, since some markers may disappear and reappear frequently, leading to small fragments of data. In order to be able to apply a Kalman smoother filter at all, the amount of available data frames should at least be larger than the order of the filter. This means at least four data frames are needed to apply the recommended third order Kalman filter. However, in this case, the smoothed data is still identical to the original data, increasing the minimum amount of frames to five. Whether this is a useful minimum is discussed next.

Smoothing data can be used to remove noise in the data before a gap, allowing for more accurate estimations of position and velocity. Figure 8 shows the effect of applying a Kalman smoother filter over a very small number of frames. As the figure illustrates, smoothing over five data frames fails to establish any useful noise reduction and results in an inaccurate ending velocity. Smoothing over seven or nine frames seems to result in a more significant amount of noise reduction, but still fails when it comes to ending velocity. It should also be noted that nine subsequent frames may well be part of the same noise peak, as can be seen by looking back at Figure 6 or 7. For these reasons, a larger amount of frames seems desirable.

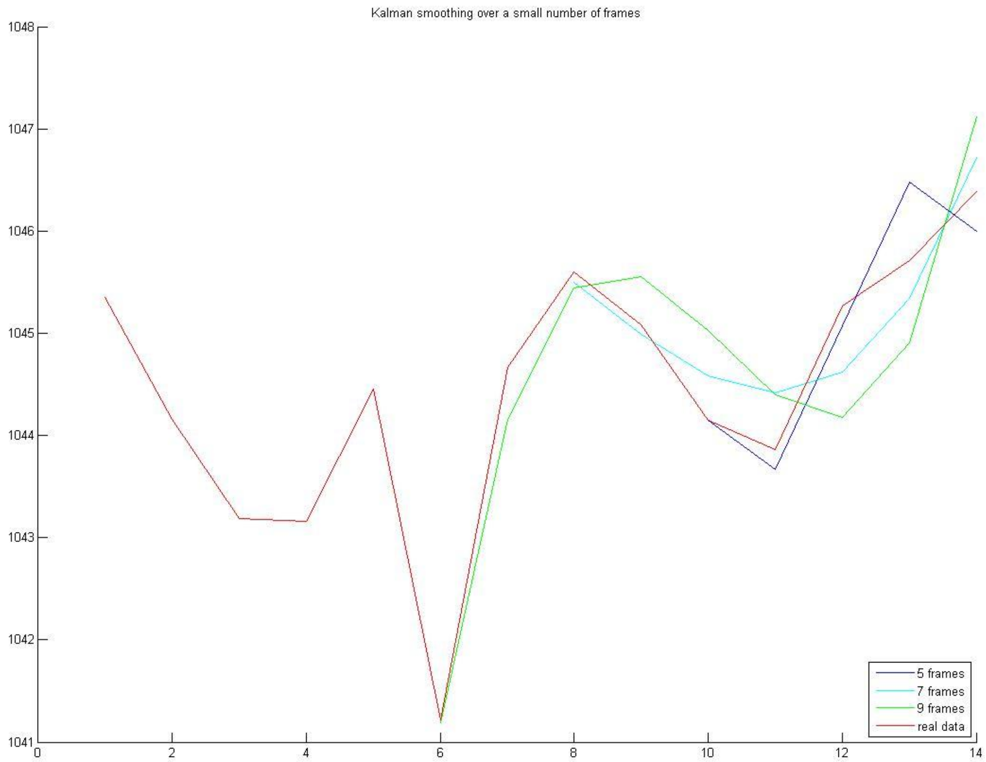


Figure 8. Kalman smoothing over a limited amount of frames.

Figure 9 shows the effects of smoothing segments of 30, 40 and 50 frames with respect to smoothing the entire data set (consisting of 100 frames). The figure shows the ending velocity after smoothing over 50 previous frames is close to that of smoothing over the entire dataset. Table 1 shows the mean differences between ending velocities after smoothing over the entire set of previous data points and over various limited amounts of previous data points. (The data that was used to generate these results consisted of six different datasets, corresponding to the x, y and z coordinates of two different markers. This data will also be used for experiments in other sections of this report. Differences between ending velocities were calculated for each frame of each dataset, starting at frame 101, since a number of previous frames were needed.)

Although Table 1 illustrates that the differences keep decreasing as the segment size increases, these improvements seem to become less substantial after increases above 50 frames. For this reason, 50 previous frames seems to be a useful minimum for smoothing intended to improve prediction accuracy.

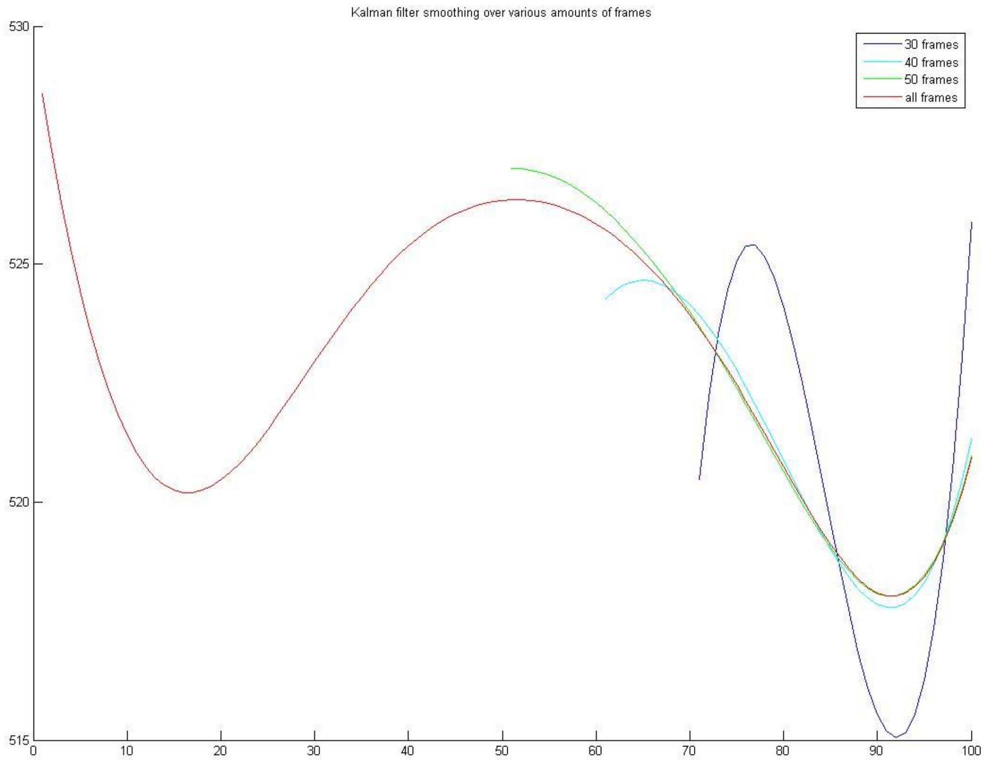


Figure 9. Kalman filter smoothing over a limited amount of previous frames, compared to smoothing over all previous frames.

#previous frames	20	30	40	50	60	70	80	90	100
mean difference	0.3784	0.3227	0.2751	0.2424	0.2318	0.2237	0.2149	0.2106	0.2094
standard deviation	0.7490	0.5699	0.4544	0.3911	0.3832	0.3805	0.3777	0.3793	0.3784
maximum difference	6.5003	4.9709	3.7740	3.2628	3.1662	3.1397	3.1326	3.1347	3.1285

Table 1. Differences between ending velocities after smoothing over various amounts of previous frames (mm/s).

If a segment of data does not contain at least this amount of frames because two gaps in the data follow each other in quick succession, it may be helpful to interpolate over the first gap in order to obtain more data points for the prediction used to fill the second gap. Interpolation methods will be discussed in Section 5.3.

5. Gap filling

Data gaps are a result of marker occlusion in optical motion capture, indicating a period of time in which a marker was not visible for a sufficient amount of camera's, or was outside of the acquisition volume, making it impossible to determine its position.
There are two ways of dealing with gaps in the motion capture data of a specific marker: interpolation and extrapolation. Since interpolation requires both data before and after the gap its use is limited to post-processing methods. It also requires that segments of data can be identified as belonging to the same marker. Unless markers are uniquely distinguishable, this is a non-trivial task. Therefore, the main focus of this section will be on extrapolation methods, which may be used in real-time processing and may also help in matching data segments. After this, spline interpolation will also be discussed as a post-processing method.

5.1 Average velocity prediction

This method predicts the position of markers that have gone missing based on the average velocity over the n previous frames (where the velocity of a marker at frame i is calculated as the position of the marker at frame i minus its position at frame $i-1$). Although one would expect the ending velocity to be the most accurate one to use for prediction, this may not be the case due to noise in the data. Figure 10 illustrates the results of predicting ten frames ahead using average velocity prediction for $n = 10$, $n = 20$ and $n = 30$. To establish whether there is an optimal value for n, average deviation of these predictions were determined for different values of n and different sizes of 'gaps'. (The dataset was the same as in Section 3.3.1.)
Prediction deviations (from the actual, smoothed data) were determined after predicting one to thirty frames ahead, using prediction based on $2 \leq n \leq 22$ previous frames. Since prediction accuracy can be expected to decrease as one looks further ahead, for each gap size, these deviations were normalized by dividing them by the minimum deviation over all values of n. This means the most accurate prediction for a certain gap size will have a value of 1, and every other prediction will have a value larger than 1. The average of these results over all gap sizes was then computed for each of the six datasets and normalized again, the results of which are shown in Table 2.
As the table shows, this optimal value is not consistent among the different data sets, although predictions using $n = 12$ previous frames seem to perform quite well on average. An exception to this is the fifth data set, which can be most accurately predicted using only two or three previous data frames (resulting in an approximation of the ending velocity). On closer inspection, this dataset turned out to be almost noiseless, in which case the most recent velocity could indeed be expected to be the most accurate one.

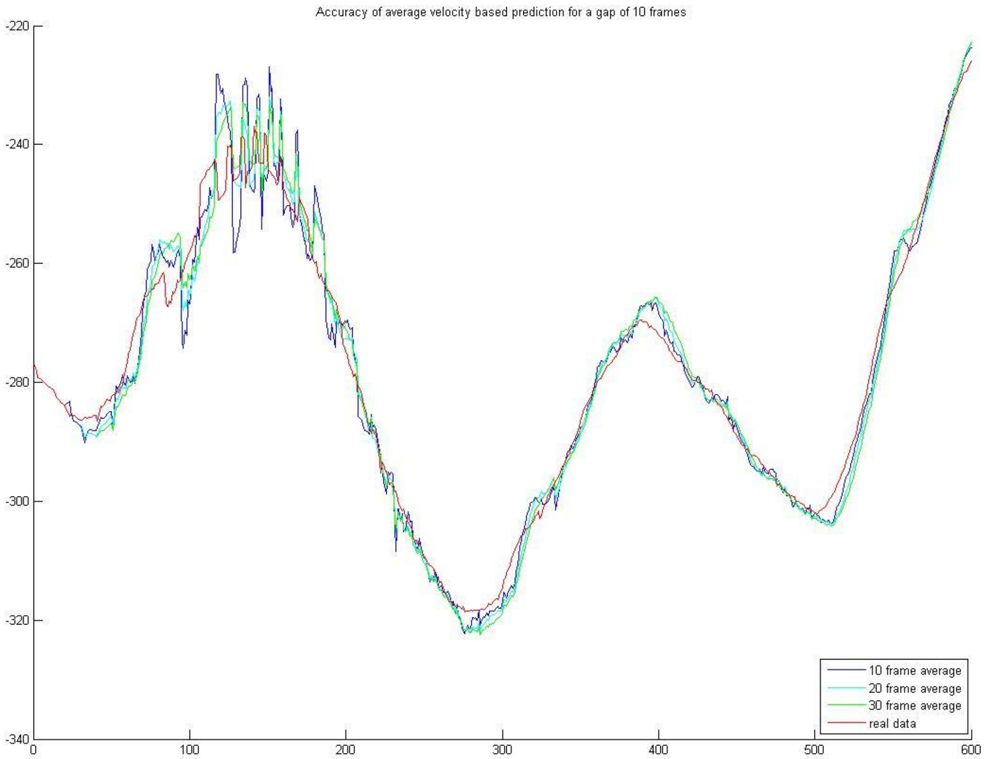


Figure 10. Predicted values after a 10 frame gap using average velocity prediction over 10, 20 and 30 previous frames.

#frames used	marker1x	marker1y	marker1z	marker2x	marker2y	marker2z	average
2	1.872542	1.648686	1.358519	2.190731	1.002761	1.34884	1.458502
3	1.625096	1.518983	1.244206	1.695642	1	1.14571	1.273905
4	1.499177	1.433857	1.166096	1.51331	1.019998	1.069129	1.192176
5	1.415986	1.360005	1.109053	1.392531	1.04578	1.03344	1.138824
6	1.330127	1.292002	1.065074	1.304951	1.072415	1.015133	1.095957
7	1.25738	1.228737	1.032028	1.24636	1.101562	1.007404	1.064062
8	1.197334	1.170599	1.01137	1.188625	1.132118	1.002125	1.037573
9	1.144079	1.119318	1.000856	1.142664	1.164269	1	1.017325
10	1.105811	1.085203	1	1.105273	1.197474	1.000092	1.005377
11	1.079664	1.063387	1.008502	1.073874	1.231724	1.002302	1.000071
12	1.061871	1.051915	1.022202	1.05008	1.266598	1.006223	1
13	1.049029	1.043263	1.037827	1.031969	1.302055	1.012291	1.002732
14	1.037459	1.035107	1.053137	1.017451	1.337796	1.02071	1.006653
15	1.027571	1.025952	1.067838	1.008326	1.373793	1.030889	1.011733
16	1.018769	1.016348	1.082895	1.002453	1.410154	1.042057	1.01768
17	1.010304	1.007508	1.0991	1.000097	1.446729	1.053998	1.024674
18	1.003268	1.000931	1.117251	1	1.48338	1.066206	1.032942
19	1	1	1.137092	1.001855	1.52017	1.077913	1.043174
20	1.001263	1.003489	1.157612	1.004467	1.557121	1.089192	1.05497
21	1.004609	1.008701	1.178613	1.007537	1.594092	1.100235	1.067466
22	1.008896	1.013638	1.199774	1.010513	1.630896	1.110763	1.07997

Table 2. Average (normalized) prediction deviation based on average velocity prediction over 2-22 previous frames.

To see whether smoothing could improve the accuracy of predictions or could reduce the number of previous frames over which average velocity needs to be calculated, the experiment was repeated, but this time, the 50 previous frames were smoothed before prediction. The results are shown in Table 3. For the third and fifth data set, using the ending velocity (based on the last two frames) turns out to be the most accurate prediction method indeed. For the fifth data set, this is expected to be due to the low level of noise, although for the third data set (depicted earlier in Figure 2), it may also be due to the frequent change in direction, favoring a prediction method that reacts quickly. The other data sets do not follow this trend and in these cases prediction performs better using a larger amount of previous frames after smoothing. On average, according to Table 3, using $n = 13$ previous frames provides the most accurate prediction after smoothing. This is close to the optimal value $n = 12$ of prediction without smoothing.

#frames used	marker1x	marker1y	marker1z	marker2x	marker2y	marker2z	average
2	1.217908	1.11786	1	1.285279	1	1.066909	1.053392
3	1.199145	1.10441	1.003812	1.262529	1.001227	1.05506	1.043712
4	1.181215	1.091786	1.009017	1.240544	1.003453	1.044582	1.035008
5	1.164186	1.080052	1.015754	1.219368	1.006724	1.03541	1.027327
6	1.148008	1.069223	1.023889	1.198711	1.011081	1.027426	1.020585
7	1.132613	1.059271	1.033229	1.178571	1.016614	1.020467	1.014723
8	1.117979	1.050109	1.043762	1.159133	1.023388	1.014491	1.009756
9	1.104311	1.041795	1.055566	1.14095	1.031418	1.009874	1.005886
10	1.091375	1.034268	1.068492	1.12408	1.040666	1.006481	1.003024
11	1.079142	1.027448	1.082839	1.10837	1.05123	1.004056	1.001152
12	1.067783	1.021345	1.098313	1.09367	1.063039	1.002267	1.000165
13	1.057183	1.016106	1.114675	1.079876	1.076138	1.000988	1
14	1.047338	1.011593	1.131985	1.067044	1.09046	1.00027	1.000651
15	1.038416	1.007805	1.149932	1.055392	1.106202	1	1.002143
16	1.030254	1.004767	1.168574	1.044668	1.123369	1.000195	1.004426
17	1.023025	1.002421	1.187691	1.034789	1.142397	1.000751	1.007524
18	1.01668	1.00085	1.207314	1.02575	1.162733	1.001682	1.011361
19	1.01128	1.000072	1.227391	1.01775	1.183991	1.002892	1.015895
20	1.006653	1	1.247803	1.010837	1.205819	1.004415	1.021024
21	1.002912	1.000721	1.268302	1.004916	1.228067	1.006193	1.026692
22	1	1.002207	1.288884	1	1.25057	1.008127	1.032846

Table 3. Average (normalized) prediction deviation based on average velocity prediction over 2-22 previous frames after smoothing the previous data..

Finally, to compare the different prediction methods, prediction deviation was determined for prediction using 12 unsmoothed previous frames, 13 smoothed previous frames and 2 smoothed previous frames. The results are displayed in Table 4, showing the first prediction (using 12 unsmoothed previous frames) method is the most accurate one. Although the second method (using 13 smoothed previous frames) performs better than the third (using two smoothed previous frames), it can be observed that the third prediction method becomes more accurate as gap size increases.

gap size	5	10	15	20	25	30	average
method 1	1	1	1	1	1	1	1
method 2	1.147294	1.095726	1.064244	1.051551	1.044527	1.042599	1.08748
method 3	1.231998	1.111382	1.054874	1.038751	1.036007	1.041173	1.105797

Table 4. Comparison of prediction accuracy for various gap sizes. Method 1: 12 unsmoothed frames, method 2: 13 smoothed frames, method 3: 2 smoothed frames.

5.2 Polyfit prediction

This method tries to find a polynomial function that fits a dataset as good as possible, using the Matlab method “polyfit”. [7] This function may then be used to predict values at different points in time, possibly making it useful as a gap filling or prediction method.
Figure 11 shows the result of applying a polyfit to all previous data points and then using this polynomial function to predict the value for the current frame. Note that this is not a continuous function, a new function was created for each frame. As the figure shows, the polyfit function does not appear capable of fitting accurately to the entire set of data as the number of frames increases. For this reason, the accuracy of predictions based on a smaller amount of previous frames was researched next.
Figure 12 shows the result of filling a number of artificially created gaps using polyfit prediction based on twenty previous frames. Figure 13 displays the same predictions using smoothed data as input. As can be seen in the figures, some predictions seem to be reasonably accurate, but at other points the difference between predicted position and real position seems to increase rapidly, making it an unreliable method. Such results were also found for predictions based on smaller and larger amounts of frames. It can also be noted that the prediction does not always start at the last (smoothed) data point, providing a relatively large deviation even for a small number of missing data frames. Based on these observations, the polyfit method was discarded as a potential method for gap filling.

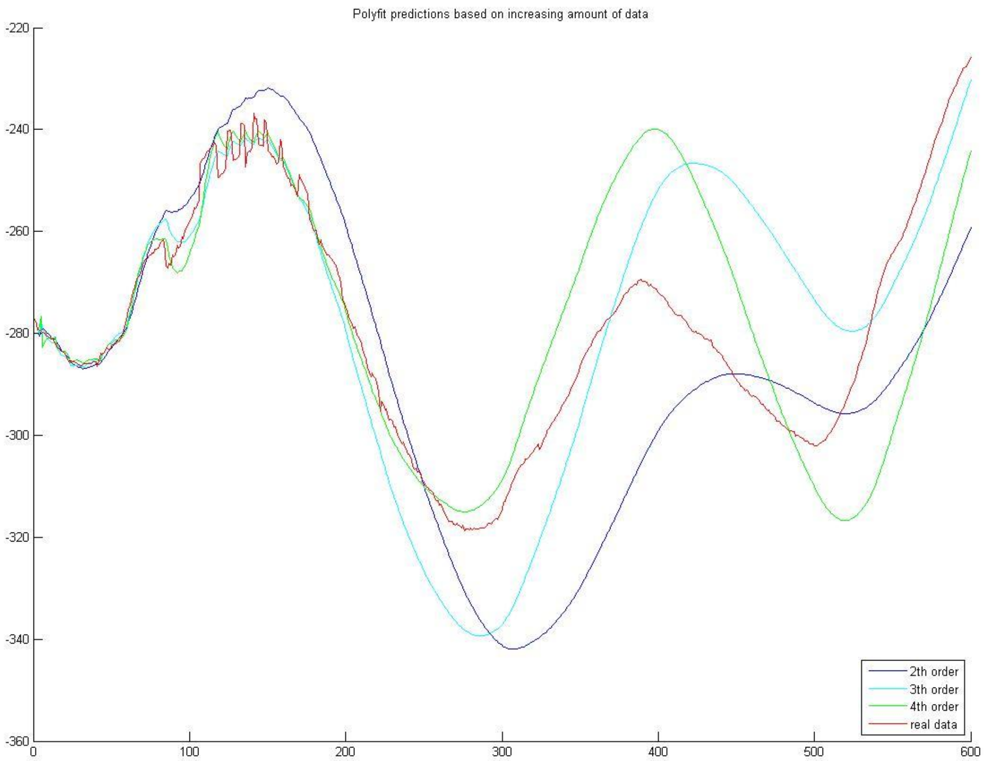


Figure 11. Polyfit approximation of data based on an increasing amount of previous frames.

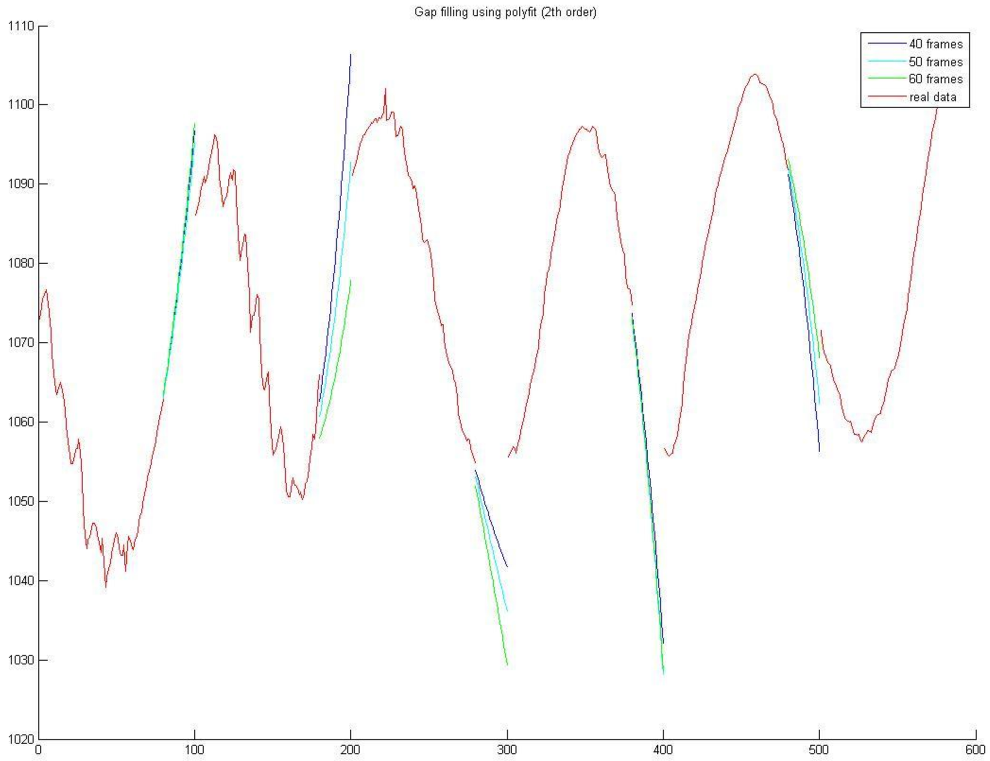


Figure 12. Gap filling using polyfit prediction on 40, 50 and 60 frames of raw data.

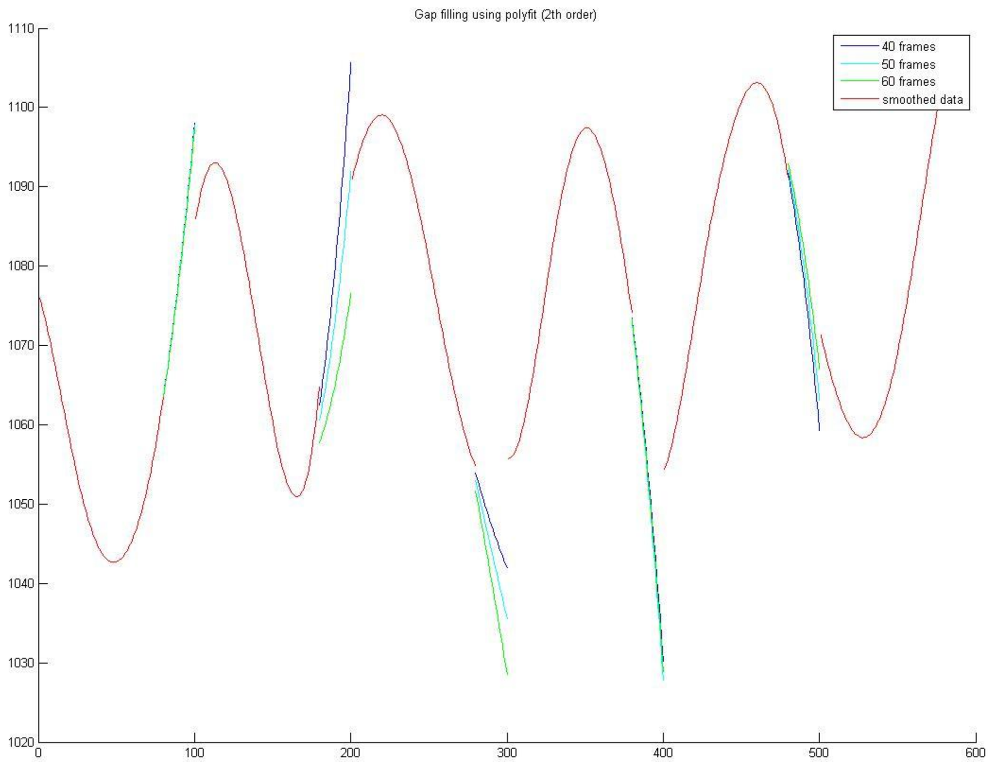


Figure 13. Gap filling using polyfit prediction on 40, 50 and 60 frames of smoothed data.

5.3 Spline interpolation

During post-processing, gaps may also be filled using interpolation methods rather than extrapolation methods, allowing for the use of data captured after the gap.
There are numerous interpolation methods, a number of which have been included in Matlab. [8]
Experimentation was done with Hermite spline interpolation, using the “pchip” function. [9] This method fills a gap using position and velocity at both ends of the gap to create a C2 continuous spline.
Figure 14 shows an example of a gap filled using the pchip function. Although it does not appear to be C2 continuous, this is due to the large sampling size which was used for sampling interpolated points on the spline. (Samples taken along the spline were one frame apart, since there is no for sampling inbetween frames for data processing purposes. Using a higher sample rate would show C2 continuity.)

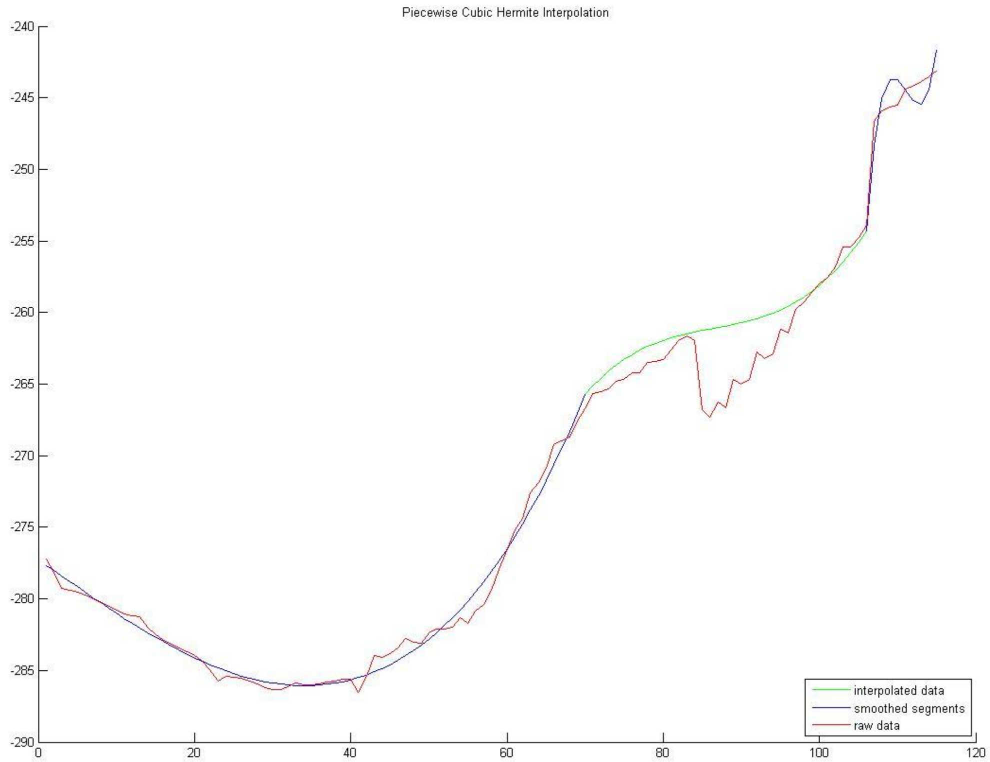


Figure 14. Gap filling using Hermite spline interpolation.

Table 5 shows the results of comparing the accuracy of average velocity prediction (using thirteen smoothed previous frames) and Hermite spline interpolation by comparing their average deviation from the actual, smoothed data over gaps of different sizes. Data before the gaps was smoothed for both methods. Data after the gap was not smoothed. The results suggest average velocity prediction is more accurate than spline interpolation. Still, spline interpolation can be expected to be more accurate toward the end of the gap, since it will converge to the point at which data reappears. Further experimentation could be done to confirm this and other interpolation methods may provide different results as well.

gap size	5	10	15	20	25	30	average
interpolation	1.159234	1.203646	1.355903	1.556118	1.84818	2.227502	1.55843
extrapolation	1.104334	1.100329	1.096698	1.093716	1.090358	1.086264	1.095283

Table 5. Comparison of interpolation and extrapolation accuracy (normalized), calculated for gaps of different amounts of frames.

6. Marker identification

During optical motion capture using passive markers, markers are not uniquely distinguishable. This means markers have to be identified based on their expected position and behavior over subsequent frames. When markers are occluded for a relatively long period of time, motion capture software may not be able to identify markers correctly, resulting in data segments of the same marker that are stored as if they belong to multiple separate markers. In this section, a method will be presented that deals with the following marker identification problems:

- Identifying whether a new data segment belongs to a marker that has disappeared earlier.
- Identifying ghost markers, reflections in the environment that are incorrectly identified as markers.

6.1 Method overview

An implementation of the marker identification method was made in Matlab, which used the following input format: data was provided in a table/matrix format, where rows were used to store data for subsequent frames and columns were used for separate markers, storing x, y and z-coordinates in three subsequent columns. Due to the problems with occlusion described earlier, data belonging to one marker may be stored in a number of columns.
The method works on a frame-by-frame basis. For each frame, processing was divided between dealing with existing columns (columns for which data was available in previous frames) and new columns (columns in which data appeared for the first time during this frame). For all existing columns, a status was maintained, which was one of the following:

- Available: data was available in previous frames
- Missing: the marker has gone missing in one of the previous frames
- Interpolating: the marker has reappeared and interpolation is going on between the predicted position used to fill the gap and the new data
- Continued elsewhere: the marker has reappeared in another column and processing should continue at that point

For each existing marker (group of three columns), this status information can be combined with the available data in the current frame to determine an appropriate action.
If data was available before and is available for this frame as well, the data for the current frame can be read and stored normally. If no data is available for the current frame, the status can be changed to missing and extrapolation can be started to predict marker position.
If the marker was missing and continues to be missing (markers may also reappear in the same column), extrapolation can be continued. If the marker reappears, interpolation can be started to close the gap between prediction and actual data.
If interpolation is going on and data is available for the current frame, it can continue, possibly changing the status to available when interpolation is finished. If no data is available, the status should be changed to missing again and the position for the current frame should be extrapolated. As

will be discussed later, prediction based on a small set of previous frames may be inaccurate and ways may be found to obtain a larger dataset for use in prediction methods.

Now, for each new data column in a frame, a check should be made to see whether the data might match one of the existing markers that has gone missing. This can be done by comparing the predicted position of the existing marker with the position in the new set of columns, allowing for a certain error margin. By doing this for all missing markers, a best match can be obtained. If no match is found, the data in the new columns is assumed to belong to a marker that was not visible before. If a match is found, the status of the old column can be set to continued elsewhere, while interpolation can be started for the new column.

6.1.1 Prediction method

In the implementation in Matlab, extrapolation was done using the average velocity prediction discussed in Section 5.1, or rather, the most recent velocity after smoothing the available data for the previous points using the Kalman smoother filter. Smoothing of the previous data points was done as soon as a marker went missing, using all data points up to the previous gap. If this set of data was too small, prediction might become unreliable, as was discussed in Section 4.3.1. For this reason, the pchip method, discussed in Section 5.3, was used to fill the previous gap in these cases, providing more data points for smoothing.

6.1.2 Best match measurement and error threshold

When the predicted position of a missing marker is compared with the data in a new column, one should allow for a certain error threshold for two reasons: prediction may be inaccurate and the newly obtained data may be subject to noise.
To determine an appropriate error threshold, noise was measured by calculating the difference between a raw dataset and a dataset that was smoothed using the Kalman smoother filter. Table 6 displays the results of measuring noise for six different coordinates from two different markers over 600 frames.
The other part of the error threshold should be based on prediction accuracy. Since prediction accuracy can be expected to decrease as gap size increases (which is confirmed by the results presented in Section 5), it may be useful to create a variable error threshold for this, which is dependent on gap size. One should, however, pay attention that the error threshold does not become so large that any other signal is accepted as a reappearance over time.

	marker 1x	marker 1y	marker 1z	marker 2x	marker 2y	marker 2z
mean	0.885759	1.957587	0.976544	0.746959	1.230555	1.173431
standard deviation	1.050888	2.61296	0.947095	0.699195	1.218669	1.198924
maximum	5.442585	12.00994	5.566206	4.975042	8.877608	7.027465

Table 6. Noise measurements for different signals, obtained by taking the absolute difference between raw data and data smoothed using the Kalman smoother filter (in mm).

6.1.3 Real-time processing: Interpolation after marker reappearance

Once a marker reappears, the predicted position of that marker may differ from the new measurements. For this reason, a simple interpolation method was created that can be used in a real-time processing pipeline, smoothing the correction that needs to take place, illustrated in Figure 15 . Assuming interpolation takes place over n frames, the first interpolated coordinates can be obtained by following the vector from the predicted position in the previous data frame toward the new data position for $1/n$ of its length. After this, the i-th interpolated value can be obtained by following the vector from the previous interpolated position toward the new data position for i/n of its length, where $2 \le i \le n$.

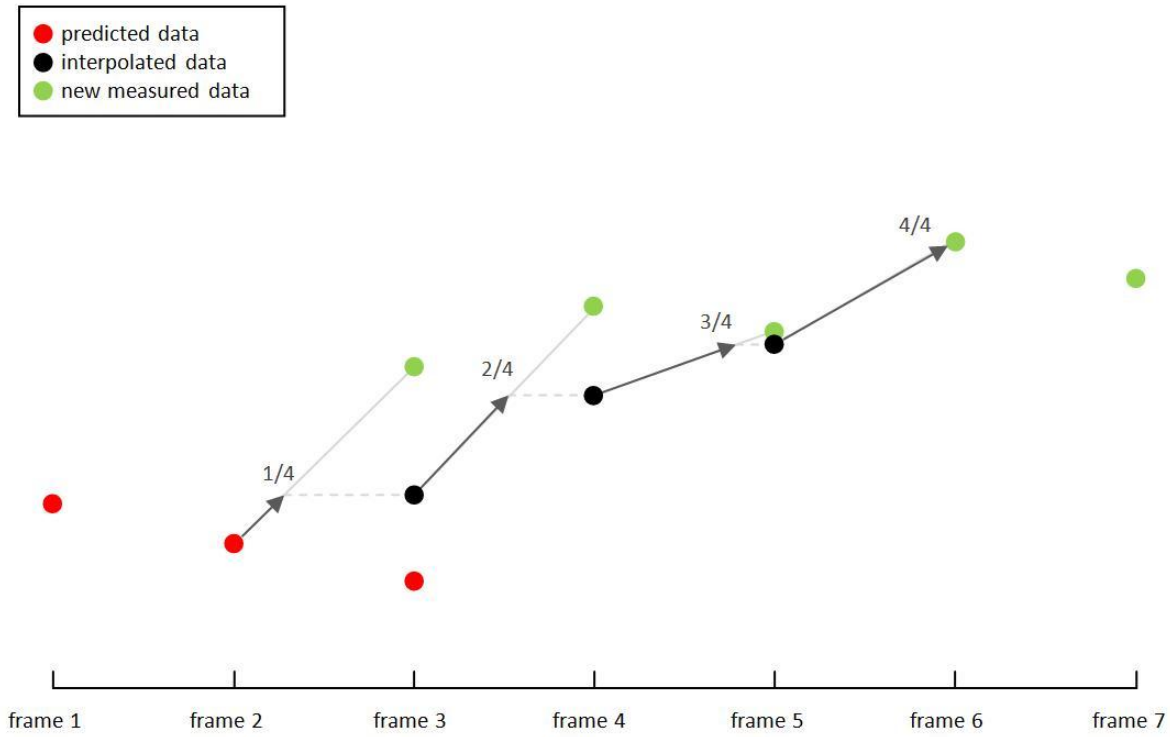


Figure 15. Interpolation over four frames after marker reappearance. As the marker reappears in frame 3, the interpolated position is determined by following the vector from the predicted position in frame 2 to the data position in frame 3 for ¼ of its length. In the next frame, the vector from the previous interpolated position and current real data position is followed for ½ of its length. This process is repeated until interpolation reaches the actual data position in frame 6.

An example of the result of this method is shown in Figure 16, showing the result of interpolating over 10 frames after marker reappearance. This appears to be a reasonable amount of time, finding a balance between an smooth transition and getting back to the actual position as quickly as possible. Note that this form of interpolation is only necessary in a real-time application. In a post-processing method, gaps could presumably be filled more accurate using interpolation methods such as the Hermite spline. Note that in this case, extrapolation is still necessary to perform marker identification at the point of reappearance.

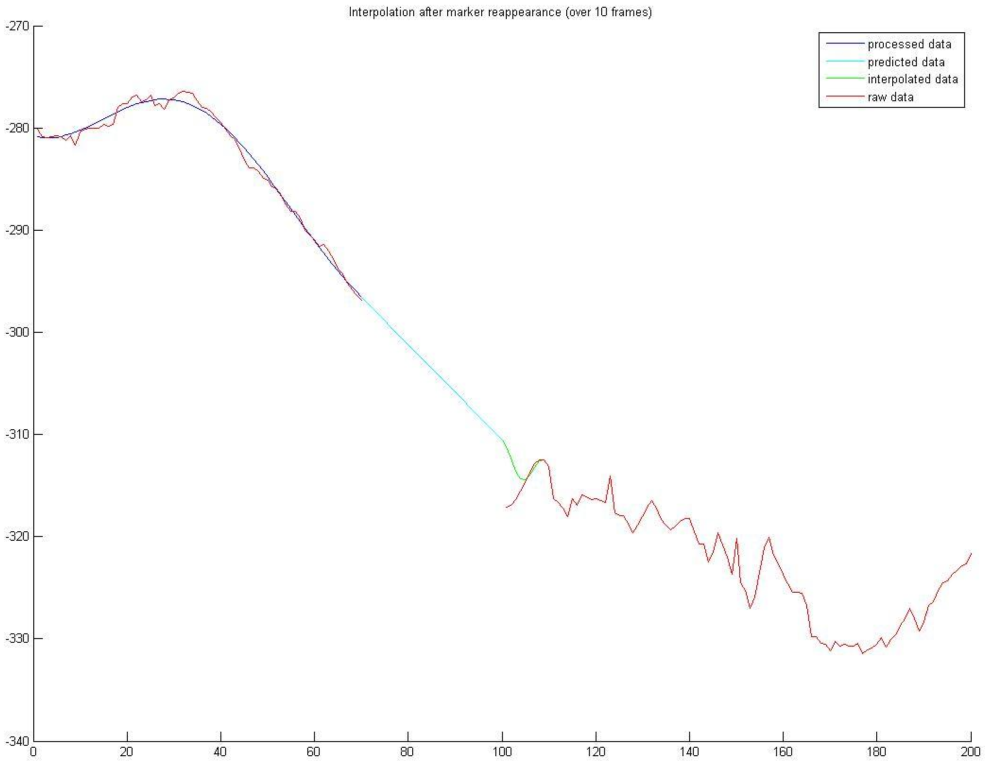


Figure 16. Interpolation after marker reappearance over a period of ten frames.

6.2 Remaining issues and possible extensions

6.2.1 Detecting and resolving incorrect marker identification in previous frames

Marker identification is not a flawless process and it could happen that a data column is labeled incorrectly at some point. Such errors may become evident at a later point in time and could be corrected. For instance, a new data column may appear containing positional data that is close enough to the predicted position of a missing marker to be considered its reappearance. A few frames later, however, another new data column appears, containing a significantly better match than the one before. It is likely that this new data column belongs to the missing marker and the other column belongs to another one, or to a new marker. The new data column can now be labeled as the reappearance of the missing marker, while the other data column can be reevaluated.
To be able to detect such mistakes, prediction could be continued for a number of frames after a marker is labeled as having reappeared in another column. If a significantly better match is found during this time, labeling adjustments can be made.

Motion capture systems may also perform incorrect marker identification. If two markers come very close to each other, the signals may be mixed up and the markers are switched. Such switches could be detected by identifying markers that come close to each other and comparing predicted position with the position found in the data during the subsequent frames. If the prediction is found to match the data of the other marker more accurately than its own data, this could indicate the markers have been mixed up by the system.

6.2.2 Dealing with ghost markers

Ghost markers are recordings of an optical motion capture system that do not belong to any marker, such as reflections on certain surfaces. Ghost markers should be ignored, but need to be identified as such before this can be done. One way to do this is to use knowledge about the total number of markers that is used. If there are ten markers of which none are missing and an eleventh signal appears, it can be assumed that one of the signals belongs to a ghost marker.

Also, if the total number of markers is assumed to be known, and all markers have been visible in earlier frames, a signal may be identified as a ghost marker if its position does not correspond to the predicted position of any of the missing markers.

Ghost markers may also be identified by considering their position with respect to those of other markers. If positional data is far away from any other marker positions, it is likely that this is some reflection in the environment, rather than the observation of an actual marker.

7. Neighboring marker constraints

In the gap filling and marker identification methods discussed so far, it was assumed that there was no knowledge on the placement of markers and the underlying skeleton of the subject. Such knowledge, however, may be used to determine constraints on the position and movement of markers. Such constraints may aid in the processes of extrapolation and marker identification.

This section will describe the results of applying skeleton based constraints by taking into account the fixed distances that exist between marker pairs placed on the same body part during these processes. For these experiments, a different dataset was used than the one used in the previous sections. It consisted for 1280 frames of motion capture data for six markers, placed on the shoulder, upper arm, elbow, inner wrist, top wrist and outer wrist of the right arm. The markers and their fixed distance relations are indicated in Figure 17. The motion consisted of picking up an object from the floor.

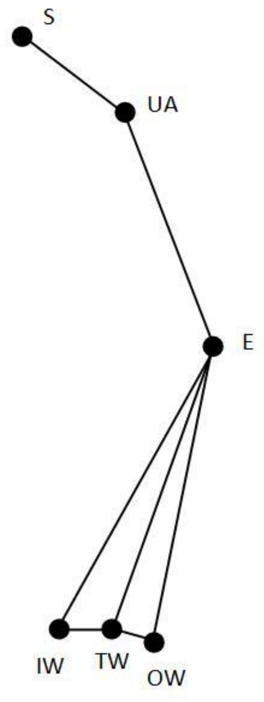


Figure 17. Markers and their fixed distance relations. S = shoulder, UA = upper arm, E = elbow, IW = inner wrist, TW = top wrist, OW = outer wrist

To determine the fixed marker distances, average distance between all pairs of markers with a relatively fixed distance relation was determined over all frames in the dataset (although usually, this could also be determined during a 'static pose' trial). Since markers may shift with respect to the underlying skeleton and since captured data may be subject to noise, the measured marker distances can be expected to differ slightly over time. For this reason, the standard deviation in marker distance was also determined for each pair with a fixed distance relationship (called neighbors from now on). The average distance and standard deviation for all pairs of markers is shown in Table 7.

	S-UA	UA-E	E-IW	E-TW	E-OW	IW-TW	IW-OW	TW-OW
average distance	14.39545	22.22558	24.29905	23.25881	22.59154	5.830325	8.675259	5.91047
standard deviation	1.35672	0.657855	0.151324	0.253707	0.52324	0.064943	0.158074	0.0837

Table 7.Average marker distance and standard deviation in distance(in cm) for all pairs of markers with a fixed distance relationship. S = shoulder, UA = upper arm, E = elbow, IW = inner wrist, TW = top wrist, OW = outer wrist.

7.1 Gap filling using distance constraints

The fixed distances between pairs of markers may be used to improve prediction methods if one of them goes missing. A method was developed in which prediction was done as usual as a first step, after which its predicted position was evaluated and possibly corrected based on distance constraints in a second step.

For the initial prediction average velocity prediction was used based on the last two frames, after smoothing the previous data at the start of a gap. When a predicted position for the missing marker was obtained, its position was compared to that of all of its neighbors. For each neighbor, the distance between predicted position of the missing marker and the current position of the neighboring marker was not allowed to exceed the sum of the expected distance (stored in a distance matrix) and the allowed error margin (stored in a distance error matrix). The error margin for each pair of related markers was set to twice the standard deviation reported in Table 7.

If the distance at the current frame exceeded the sum of the expected distance and error margin, its position was corrected as follows: a vector pointing from the position of the neighbor to the predicted position was obtained, which was then normalized and multiplied by the expected distance stored in the distance matrix. This process is illustrated in Figure 18. Distance correction was applied for one neighbor at a time, which means one correction might cause the constraints of a previously evaluated neighbor to be violated. For this reason, all neighbors were evaluated iteratively until no more corrections occurred.

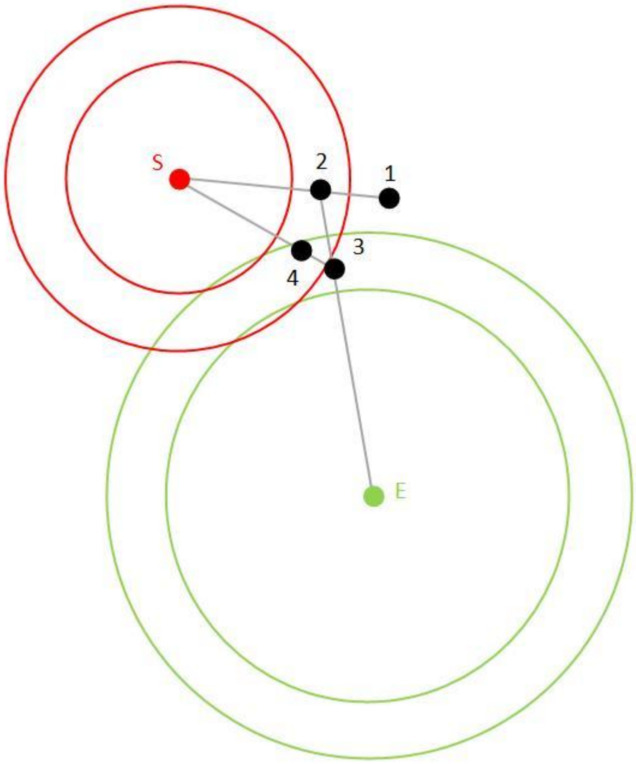


Figure 18. A sequence of positions of the upper arm marker during distance correction. 1. Initial predicted position, outside the error margin of the shoulder marker. 2. New position after distance correction for the shoulder marker, outside the error margin of the elbow marker. 3. New position after distance correction for the elbow marker, outside the error margin of the shoulder marker. 4. New position after distance correction for the shoulder marker, inside the error margin of all neighbors.

The method was tested on the available dataset, to which a number of artificial gaps had been introduced. The results are displayed in Figure 19 and 20, corresponding to marker placed on the upper arm and the top wrist respectively. As the figures show, the application of distance correction seems to improve accuracy, especially for the top wrist marker, for which prediction follows the actual data very closely. This may be attributed to the fact that the top wrist marker has four neighbors, for which there is only a small deviation in distance. Figure 19 illustrates prediction the corrected prediction can become quite noisy after the first correction is applied. This is due to the fact that once a correction is made, the velocity used during the prediction in the next frame is based on the difference between the (uncorrected) position in the previous frame and the corrected position in the current frame, which is in no way related to the expected direction in which the marker is moving. This increases the likelihood of more corrections in the future, resulting in the high frequency noise that can be observed in Figure 19. Fortunately, such high frequency noise can easily be reduced in a post-processing step, as was discussed in section 4.

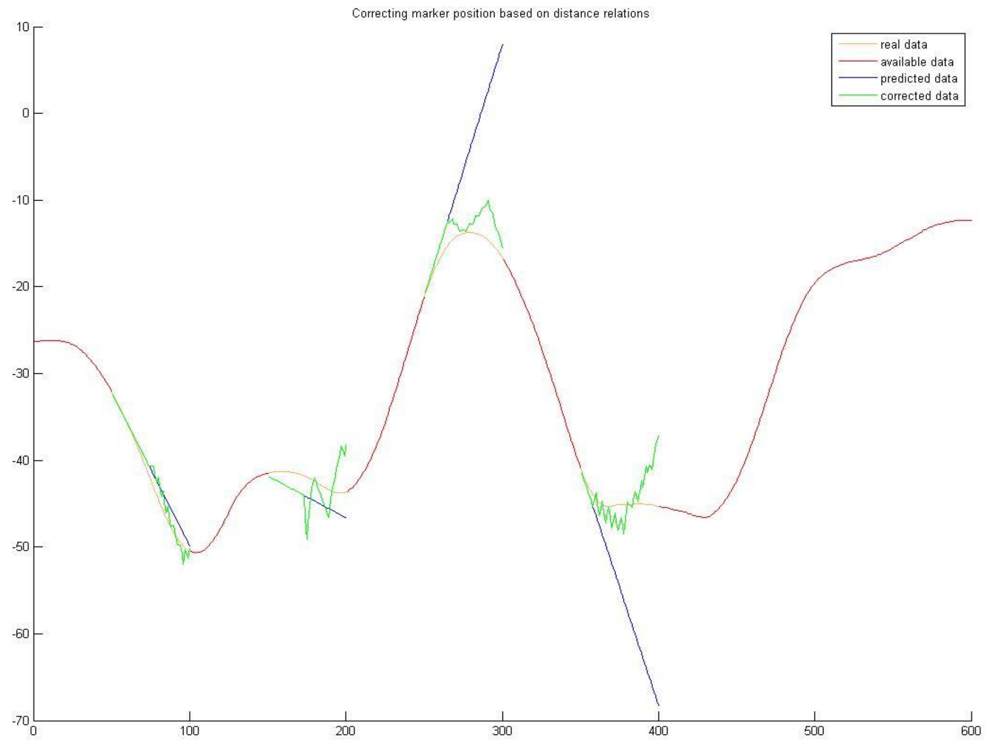


Figure 19. Comparison of extrapolation using average velocity prediction and prediction combined with distance correction for the x-coordinate of the upper arm.

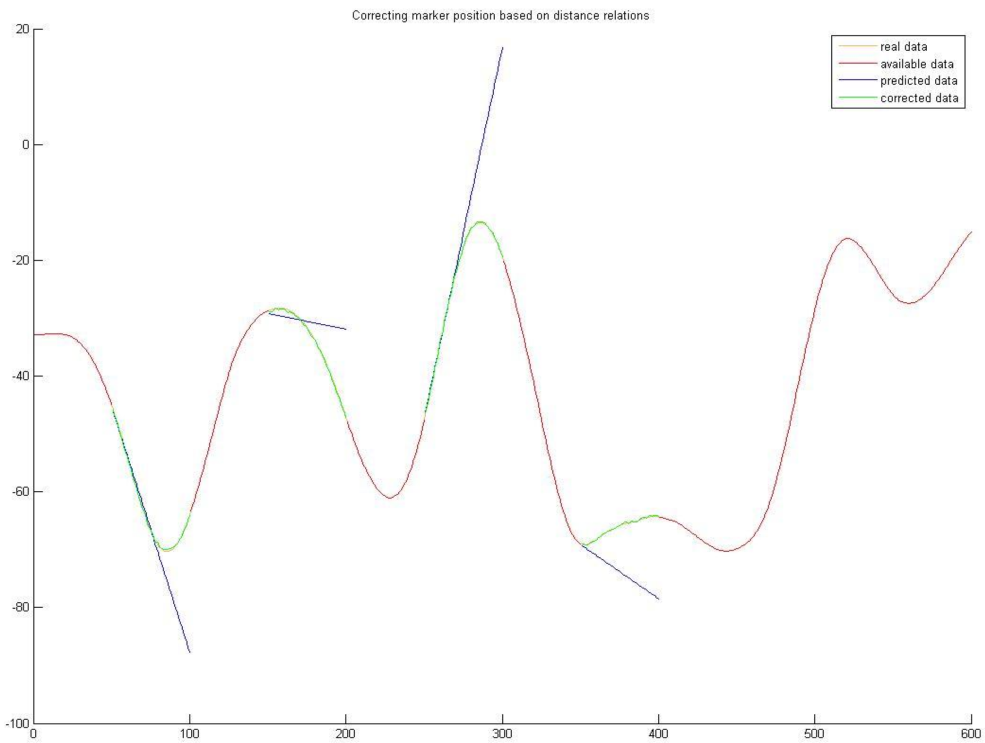


Figure 20. Comparison of extrapolation using average velocity prediction and prediction combined with distance correction for the x-coordinate of the top wrist.

Table 8 compares the average deviation (from the actual data) of prediction without distance correction and prediction with distance correction for all markers, for various gap sizes. The results have been normalized by dividing the average deviations by the minimum of the two prediction methods for each specific marker and gap size. As one can see, there are some differences between the results for different markers. Predictions for the marker placed on the elbow and the markers placed on the wrist are drastically improved by adding distance correction. Prediction for the marker on the upper arm becomes more accurate after distance correction only after a certain amount of frames, while prediction for the shoulder marker is more accurate without correction. However, for this marker, there is a rapid decrease between differences in prediction accuracy as gap size increases, suggesting prediction with correction may outperform prediction without correction at some point as gap size increases above 30 frames. Also, it should be noted that accuracy of prediction with correction may well be improved by smoothing the corrected data, since it shows a lot of high frequency noise.

The differences between the results for different markers can be explained by looking at the amount of distance relations and the error margins that are used for correction. The shoulder marker only has one neighbor (the upper arm) and the variation in distance between these markers (and the error margin based on this deviation) is relatively large compared to that of other distance relations, as can be seen in Table 8. The wrist markers, on the other hand, each have three neighbors (the other two wrist markers and the marker on the elbow) and the deviations between these markers are relatively small, resulting in tight error margins. This allows for a more accurate prediction based on distance relations. It is possible that prediction accuracy of markers such as the shoulder marker could be improved by using a smaller error margin during correction. (During marker identification, however, use of the larger error margin is advisable, since such large deviations may in fact occur.)

In general, experimental results suggest distance correction provides useful improvements to extrapolation accuracy, depending on the amount of distance relations, error margins and gap size.

gap size	5	10	15	20	25	30
shoulder (uncorrected)	1	1	1	1	1	1
shoulder (corrected)	1.654089	1.490199	1.358643	1.201002	1.086559	1.009728
upper arm (uncorrected)	1	1	1	1.231517	1.240659	1.687076
upper arm (corrected)	1.835317	1.170413	1.039515	1	1	1
elbow (uncorrected)	1.622572	2.657257	3.812826	4.839934	6.820776	10.02819
elbow (corrected)	1	1	1	1	1	1
inner wrist (uncorrected)	3.431981	8.133409	16.70691	25.19662	36.86682	54.33745
inner wrist (corrected)	1	1	1	1	1	1
top wrist (uncorrected)	4.661658	9.900018	17.78536	29.45367	43.9443	61.93673
top wrist (corrected)	1	1	1	1	1	1
outer wrist (uncorrected)	2.585695	5.127177	9.731586	18.10275	25.99722	32.61628
outer wrist (corrected)	1	1	1	1	1	1

Table 8. Comparision of gap filling for gaps of different amounts of frames using average velocity prediction (over the last two smoothed frames) and gap filling using the same prediction followed by distance correction.

7.2 Marker identification using distance constraints

Fixed distances between marker pairs may also be used during the process of marker identification. When a new marker appears, its distance to other markers can be compared to the fixed distances between a missing markers and its neighbors. If the difference between the position of a new marker and a neighboring marker of the missing marker is too large (allowing for a certain error threshold of course), it is easy to conclude that the new marker cannot be the missing marker.
Alternatively, distance relations could also be used to determine which constraints are met most accurately by the new marker, in cases where there are multiple possible candidates.

Distance constraints may also be used to identify marker switching, as discussed in Section 6.2.1. If the observed position of a marker violates its distance constraints, a check may be performed see if its position meets the constraints of another marker, and vice versa.

8. Discussion

In this report, a setup was provided for motion capture data processing pipelines, evaluating both real-time and post-processing methods. Different methods for data smoothing, gap filling and marker identification have been discussed. Recommendations about the use of particular methods for both a real-time and a post-processing pipeline will be discussed next.

Real-time pipeline
Although smoothing previous data points was expected to increase prediction accuracy due to a more accurate estimation of the last known position and velocity, experimental results contradicted this hypotheses, indicating a decrease in accuracy after smoothing. Comparison using a larger dataset belonging to a larger set of markers may determine whether smoothing previous data segments before extrapolation is in fact useful. The use of a Kalman smoother filter (third order, with experimental noise of 10e7) is recommended for this purpose over the use of the Butterworth lowpass filter, since the last becomes inaccurate at the end of a data segment, as was discussed in section four.

For gap filling, extrapolation can be applied using the average velocity prediction method discussed in Section 5.1. Experimental results indicated average velocity prediction based on 12 previous non-smoothed frames was most accurate in general, but these results are based on a small data set consisting of a limited amount of markers.
Experiments with correction of predictions based on neighboring marker constraints indicated that such corrections may in fact increase prediction accuracy in general. It should be noted that the benefits of such correction depend the amount of distance relations, error margins used during correction and the gap size over which prediction takes place.
Furthermore, more sophisticated extrapolation methods have been developed in earlier research, such as [2], [3], [4] and [5], discussed in Section 2. These may also prove to be more accurate than the methods examined in this report.

For marker identification, the approach discussed in Sections 3 and 6 seems appropriate, but improvements could be made in a number of ways: extensions could be made to reverse the effects of incorrect marker identifications at an earlier point in time, experiments with different error thresholds could be performed, information about the total amount of markers could be incorporated and the use of distance relations between markers could be added as an identification criteria.

Post-processing pipeline

In a post-processing pipeline, smoothing can be applied as a final step, after gap filling and marker identification. (As was discussed above, the usefulness of smoothing as a way to improve prediction accuracy remains unclear.) The use of a third order Kalman smoother filter with experimental noise of 10e7 produced good results for the data used during experimentation, but a different value of experimental noise may be more appropriate depending on the amount of noise in the data.

In a post-processing pipeline, gap filling will still need to be done using extrapolation for the purpose of marker identification. However, once a marker has reappeared, spline interpolation may also be used to fill the gap in the data. Although, as discussed in Section 5.3, use of the pchip function in Matlab did not result in improved accuracy. Beside Hermite spline interpolation, future research with other interpolation methods could also be performed.

For marker identification, the same approach as the one used in the real-time pipeline is recommended, and similar improvements could be made. Future research might look into the possibilities of using post-processing specific information to assist in marker identification.

In general, gap filling is considered to be the weakest (and most important) link in the pipelines and it is recommended that this area be the focus of initial future improvements.

References

[1] The MathWorks, Inc.: Butterworth filter
http://www.mathworks.com/help/toolbox/signal/butter.html

[2] Aristidou, A., Cameron, J. and Lasenby, J.: Real-time Estimation of Missing Markers in Human Motion Capture (2008)

[3] Van Rhijn, A. and Mulder, J.D.: Optical Tracking and Automatic Model Estimation of Composite Interaction Devices (2006)

[4] Sul, C.W., Jung, S.K. and Wohn, K.: Synthesis of Human Motion Using Kalman Filter (1998)

[5] Piazza, T., Lundström, J. Kunz, A. and Fjeld, M.: Predicting Missing Markers in Real-time Optical Motion Capture (2009)

[6] Murphy, K.: Kalman filter toolbox for Matlab (1998)
http://www.cs.ubc.ca/~murphyk/Software/Kalman/kalman.html

[7] The MathWorks, Inc.: Polynomial curve fitting
http://www.mathworks.com/help/techdoc/ref/polyfit.html

[8] The MathWorks, Inc.: Interpolation
http://www.mathworks.com/help/techdoc/ref/f16-5872.html#f16-6361

[9] The MathWorks, Inc.: Piecewise Cubic Hermite Interpolating Polynomial
http://www.mathworks.com/help/techdoc/ref/pchip.html