Received: 21 April 2023 | Accepted: 1 May 2023

DOI: 10.1002/cav.2166

SPECIAL ISSUE PAPER

WILEY

Online alignment of human motion using forward plotting-dynamic time warping

Mathew Randall | **Carlo Harvey** | **Ian Williams**

DMTLab, Birmingham City University, Birmingham, UK

Correspondence
Mathew Randall, DMTLab, Birmingham City University, Birmingham B4 7XG, UK.
Email: mathew.randall@bcu.ac.uk

Abstract

A number of approaches to online time warping have been proposed based on plotting an alignment path backwards from a selected end-point. When continually updating a time warp to align a time series with a live data source, such as a stream of motion capture data, the use of backwards plotting makes it hard to maintain a monotonic constraint. To solve this problem, a number of time warping approaches based on forward plotting, referred to as FP-DTW, are proposed and evaluated by applying them to human motions. We demonstrate that forward plotting for temporal alignment is a viable solution when backwards plotting is otherwise not possible.

KEYWORDS

human motion, motion alignment, time warping

1 | INTRODUCTION

There are a variety of potential applications for online time warping of a prerecorded human motion to an incomplete live human motion as it is being captured, within the fields of virtual production, live events and motion training. The ability to accurately and automatically adapt a virtual character's performance to coordinate with that of a live actor on stage or during production, would unlock a variety of creative possibilities, such as tighter interaction between avatars during live stage performances and greater flexibility when working on-set with virtual characters.

Established approaches to time warping such as Dynamic Time Warping (DTW),[1] work in offline scenarios, but do not adapt well to online scenarios in which one or both motions are only partially known. Additionally the computational processing requirements of these approaches grows exponentially with the length of the motions being aligned. This motivated the development an online approach, Open Ended DTW (OE-DTW),[2] however, this approach still relies on plotting an alignment path backwards from a chosen end point, resulting in inconsistencies between the alignment paths plotted as more of a given motion becomes known or captured, as monotonic continuity between these alignment is not enforced. For motion recognition applications this inconsistency is not an issue, however, it is when controlling the playback of recorded motion in live or real-time production scenarios, or as feedback during training.

This paper proposes a number of novel time warping methods, called Forward Plotting DTW (FP-DTW), that plot an alignment path between the two motions in a forwards direction. These methods are used to time warp a dataset of motions, with the resulting aligned motions evaluated using two methods: (i) using a similarity metric optimized for measuring alignment of two motions; and (ii) measuring the deviation between the alignment paths plotted by the proposed forward plotting algorithm and DTW.

2 | BACKGROUND

Time warping is a process of temporally aligning an input sequence of data to a target sequence. Online time warping refers to warping in real-time, in scenarios where one or both time series are incomplete and only partially known. Typically online time warping is used in data capture scenarios, such as speech recognition, in which temporal features of word sounds are being aligned to live audio capture to recognize words as they are spoken. In this study, the data sequences are time series consisting of frames of motion capture data.

There are two key challenges to online time warping. First, the computation requirements of the algorithm, need to be constrained so that they do not increase in relation to the length of the motions being aligned. Secondly, there is lack of explicit boundary conditions, as the length on one or more of the sequences is unknown.[3] The later means that cost functions, typically used in time warping algorithms, cannot be optimized for the entire path and global constraints cannot be applied.

WTW (Windowed Time Warping)[4] reduces computational requirements by breaking the data sequences down into smaller discrete windows. Alignments are plotted within each window using DTW, which are referred to as sub-alignments.

OE-DTW[2] solves the lack of boundaries problem by selecting an optimal end point within the complete data sequence, *A*, from which to start plotting an alignment with the partially known sequence, *B*. The alignment is plotted backwards from the selected end point of sequence *A* and the last know data point of sequence *B*, to the beginning of both sequences.

The selection of the optimal end point in the OE-DTW approach is unconstrained, allowing any point within the complete sequence to be chosen. Therefore, in a live scenario, in which a complete input motion sequence is being aligned to a partially known target sequence, as it is being captured, continuity is not enforced between the alignment of consecutive frames. Figure 1 shows an example in which inconsistent end points have been determined for two sequential frames (15 and 16) of a target motion. In a live scenario, this will cause the playback of the input motion to snap backwards, between the chosen end point points of each alignment, as frames 15 and 16 of the target motion is captured. This results in jittery playback of the input motion and an alignment that is not monotonic.

This paper proposes three approaches to online time warping, which monotonically align a complete prerecorded input motion with a live partially know target motion as it being captured, using forward plotting (FP-DTW). The performance of the FP-DTW algorithms is to be tested using a variety of robust objective techniques, to measure the accuracy and quality of alignment paths created. Specifically the algorithms will need to produce alignments which satisfy two key benchmarks for a majority of the tests performed: (i) produce a more accurate alignment than what can be achieved using Universal Time Warping (UTW); and (ii) produce an alignment so close to that achieved using DTW that the difference would typically not be perceptible.

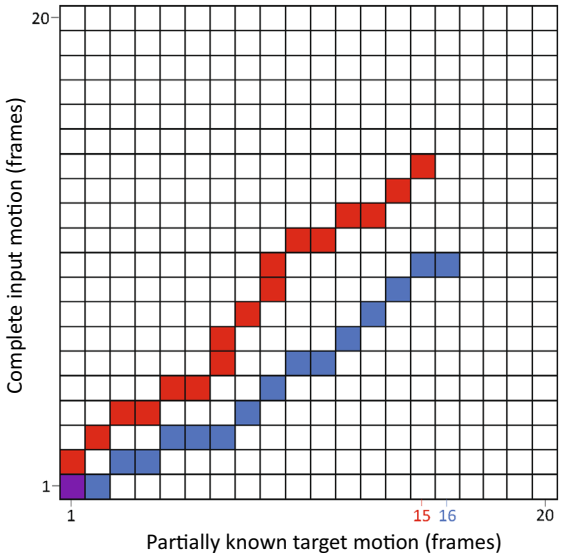


FIGURE 1 An example of the lack of continuity between the alignment paths, plotted for consecutive frames of target motion as it being captured, when the selection of the optimal input frame to map to the last captured target frame to is unconstrained.

Algorithm 1. The Method A time warping algorithm

$w \leftarrow$ Window size
$T_{n+1} \leftarrow$ Incoming target frame
$I \leftarrow$ Input frames $\{I_0..I_m\}$
$M \leftarrow$ Existing maps, $\{M_0..M_n\}$, between target and input frames
$C \leftarrow$ array of three-dimensional Euler rotations

if $M_n + w > m$ **then**
  $w = m - M_n$
end if
if $w = 0$ **then**
  $M_{n+1} = M_n$
else
  for $i \leftarrow 0$ to w **do**
    $C_i = \text{geoDistance}(I_{M_n+i}, T_{n+1})$
  end for
  $M_{n+1} = M_n + \text{argmin}\{C_0, C_1...C_w\}$
end if

3 | TIME WARPING USING FORWARD PLOTTING

This paper proposes the following three time warping methods, use FP-DTW approach:

- Method A – Frame matching: The optimal input frame to align with an incoming target frame, is selected from a window starting at the input frame selected for the previous target frame.
- Method B – Predictive windowing using DTW: The next n incoming target frames are predicted. The optimal input frame is found by plotting an alignment between the predicted target frames and a window of the input motion, using DTW.
- Method C – Predictive windowing using cost matrix: Also based on predicting the next n frames in the target motion, but with the optimal input frame being selected from an accumulated cost matrix.

Before presenting each algorithm in detail, some common terms and concepts need to be defined. Let $I = \{I_0, I_1....I_m\}$ be the frames of prerecorded input motion and $T = \{T_0, T_1....T_n\}$ be frames of a live target motion which have already been received and mapped to an input frame. The incoming target frame being processed is T_{n+1}. Set $M = \{M_0, M_1....M_n\}$ maps each received target frame to an input frame.

3.1 | Method A – frame matching

As each target frame is captured, Method A maps it to an input frame, using Algorithm 1. The frame that most closely matches the incoming target frame T_{n+1}, within window, $\{I_{M_n}..I_{M_n+w}\}$, of the prerecorded input motion, is mapped to the incoming target frame. The window of size w starts at and includes the input frame that mapped to the previously captured target frame, shown in Figure 2.

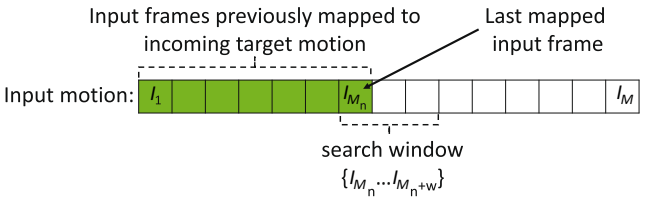


FIGURE 2 Search window used in frame matching algorithm.

Before selecting an input frame a check is performed to ensure that there are still enough unmapped frames remaining in the input motion to facilitate a search window of size w. In cases where there not enough unmapped frames, the window size is reduce accordingly. If windows size is set to 0, there are no more unmapped input frames remaining, therefore incoming target frame is mapped to same input frame as the previous target frame and the process of selecting the best matching frame is skipped.

The *geoDistance()* function measures how well two given frames match, based on the rotational distance between corresponding joints of the two frames. Equation (1) is used to measure the geodesic rotation distance between input frame a and target frame b based on an equally weighted set of joints m represented as quaternions Q.[5] The dot product is multiplied by $\frac{2}{\pi}$ to normalize the rotational distance to between 0, matching, to 1 opposite.

$$d_\theta = \sum_{j=1}^{m} \frac{2}{\pi} \arccos |Q_j^a \cdot Q_j^b|. \tag{1}$$

For this study a small window size of two was selected, mimicking a Type I local continuity constraint, which is commonly used in time warping.[6] A small pilot study was used to validate this approach.

3.2 | Method B – predictive windowing using DTW

Method B applies a smoothing function to sequence T to predict a set of future frames over a small forecast window w. A variation of the standard DTW algorithm is used to plot an alignment path within the window, then an input frame is selected to map to the incoming target frame T_{n+1}, based on the input frames mapped to T_{n+1} in the alignment path.

As each target frame is captured, Algorithm 2 is used to select an input frame to map it to. The algorithm performs the following steps: (i) check if enough target frames have been captured to forecast the motion and that there are still pre-recorded input frames remaining to align, reducing window size if required; (ii) determine predictions for future frames in the target motion F; (iii) calculate a cost matrix, C, between a window a frames within input sequence, starting at the frame mapped to last target frame, I_{M_n}, and the predicted target motion frames, F; (iv) accumulate the costs in the cost matrix to create a accumulated cost matrix D; (v) plot an alignment path P through the cost matrix, and (vi) map the incoming target frame T_{n+1} using the second alignment point on the alignment path P_1.

The *forecastJoints()* function uses dead reckoning, based on the two most recently captured frames, to forecast the Euler rotation of every joint in J over the next w frames. For short-term forecasting dead reckoning has been shown to work as well as other smoothing functions,[7] and a pilot study showed basing the forecast on only the last two frames, to be optimal. Figure 3 shows the performance of different smoothing levels at predicting i frame into future, when applied to motion datasets used in this study.

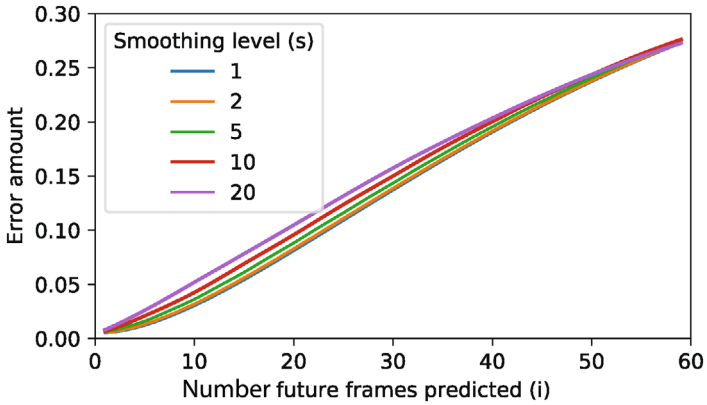


FIGURE 3 The performance of different smoothing levels at predicting human motion. Shows the error resulting from a given smoothing level s, when predicting i frames in to the future.

Algorithm 2. The Method B time warping algorithm

$I \leftarrow$ Input frames $\{I_0..I_m\}$
$T \leftarrow$ Captured target frames $\{T_0..T_n\}$
$T_{n+1} \leftarrow$ Incoming target frame
$M \leftarrow$ Existing maps, $\{M_0..M_n\}$, between target and input frames$w \leftarrow$ Forecasting window size
$s \leftarrow$ Smoothing window size
$J \leftarrow$ Set of joints $\{J_0..J_k\}$
$F \leftarrow$ 2D array of 3D vectors of size k (number of joints) x w (window size)
$C, D \leftarrow$ 2D array of floats size w_i (size of input window) x w (size of target window)
$P \leftarrow \{\}$ integers

if $n<s$ **then**
  $M_{n+1} = n + 1$
else if $M_n >= m$ **then**
  $M_{n+1} = m$
else
  $w_i = w$
  if $M_n + w_i > m$ **then**
    $w_i = m - M_n$
  end if
  $F_0 = T_n$
  $F = F +$ forecastJoints$(\{T_n, T_{n+1}\}, J, w)$
  $C =$ getCostMatrix$(\{I_{M_n}....I_{M_n+w_i}\}, F)$
  $D =$ getTotalCostMatrix(C)
  $D_{0,0} = 10,000$
  $P =$ plotPath(D)
  $M_{n+1} = M_n + P_1$
end if

The cost matrix is based on the rotational distances between corresponding joints, which are determined using quarternions. The costs of C are accumulated from (0,0) to (m, n), in the same manner as DTW. Cell (0,0) of the accumulated cost matrix, D, is set to a high value, to the force the alignment path pass through cells (1,0) or (0,1). Without this adjustment the alignment path would rarely pass through cell (I_{M_m}, F_1) in Figure 4, due to accumulated costs.

Figure 4 shows how an input frame is selected to map to the incoming target frame, based on the lowest number input frame that the plotted alignment path passes through in column T_{n+1}. To facilitate this, the DTW approach of plotting an alignment path, by stepping backwards through the accumulated cost matrix from (m, n) to (0,0), was slightly adapted. When a step in the alignment steps back one frame in the input motion, but remains on the same frame of the target motion, the input frame previously mapped to the target frame is over written with the new step. This adaptation ensures that input frame, I_{M_m}, is selected if the alignment passes through it and allows the selected input frame to be read from second point in the plotted path, P_1.

3.3 | Method C – Predictive windowing using cost matrix

This method, shown in Algorithm 3, is a streamlined version of method B. It removes the need to plot an alignment path through the accumulated cost matrix, by reversing the direction in which the costs are accumulated, accumulating the cost from (m, n) to (0,0). From matrix, D, the input frame with the lowest accumulated cost to the incoming target frame, is selected on the last line.

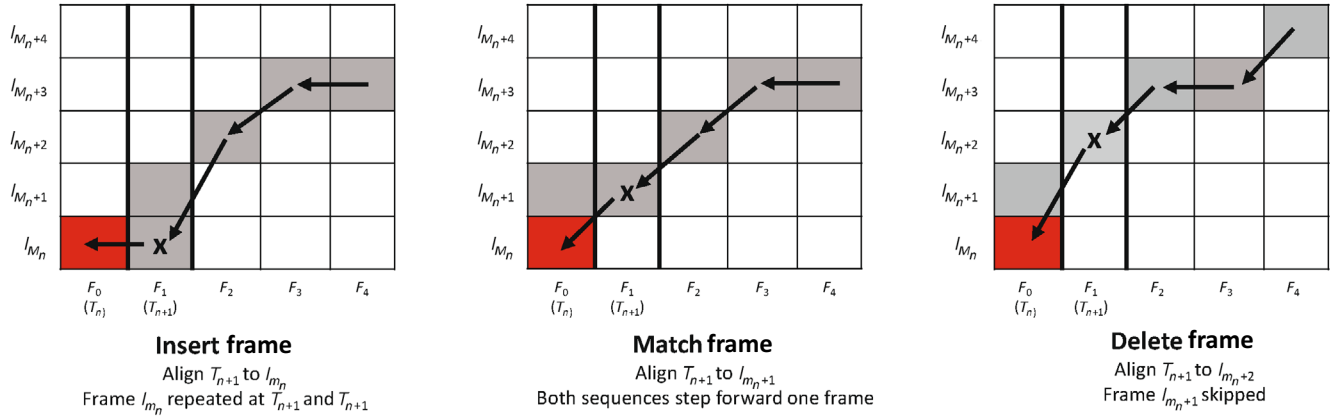


FIGURE 4 Examples of Method B selecting an input frame (X), based on the lowest number input frame that the alignment path passes through in column T_{n+1}.

Algorithm 3. The Method C time warping algorithm

$I \leftarrow$ Input frames $\{I_0..I_m\}$
$T \leftarrow$ Captured target frames $\{T_0..T_n\}$
$T_{n+1} \leftarrow$ Incoming target frame
$M \leftarrow$ Existing maps, $\{M_0..M_n\}$, between target and input frames$w \leftarrow$ Forecasting window size
$s \leftarrow$ Smoothing window size
$J \leftarrow$ Set of joints $\{J_0..J_k\}$
$F \leftarrow$ 2D array of 3D vectors of size k x w
$C, D \leftarrow$ 2D array of floats size w_i x w

if $n<s$ **then**
  $M_{n+1} = n + 1$
else if $M_n >= m$ **then**
  $M_{n+1} = m$
else
  $w_i = w$
  if $M_n + w_i > m$ **then**
    $w_i = m - M_n$
  end if
  $F = \text{forecastJoints}(\{T_n, T_{n+1}\}, J, w)$
  $C = \text{getCostMatrix}(\{I_{M_n}....I_{M_n+w_i}\}, F)$
  $D = \text{getReverseTotalCostMatrix}(C)$
  $M_{n+1} = M_n + \text{argmin}\{D_{0,0}....D_{m,0}\}$
end if

4 | METHODOLOGY

A total of 333 recorded motions of 44 different movements, were used in this study. Fifty seven from Reference 8 and 267 from widely used HDM05 dataset.[9] In both sources, human motion was sampled at 120 Hz. Recorded motions with obvious errors or unfeasible joint rotations where not included in the data set. A range of different movements were selected including: upper, lower and full body motions; ballistic and non-ballistic motions; and cyclic and noncyclic motions. Every permutation of motions with same movement were time warped in both directions, creating a data set of 3248 pairs of input and target motions to be time warped.

The window sizes tested with Methods A and B were selected based on the period of a walking gait, as this was the most prominent periodic movement cycle in the dataset. Given that the motions were sampled at 120 Hz and the period of a typical walking gait is between 0.82 and 0.9 Hz, forecast windows of much more than half a walk cycle, or 67–73 frames is likely to cause errors. Therefore Methods B and C were each implemented with forecast window sizes of 10, 20, 40, and 80 frames.

The FP-DTW methods were implemented using subset of joints which have the most impact on the overall pose of a character. These joints are: hip, knees, shoulder and elbows. The hips and shoulders had three degrees of freedom, while the knee and elbow joints had one and two degrees of freedom, respectively.

Two performance tests were used to evaluate the performance of each FP-DTW algorithm. For the first test, a similarity metric based on correlation,[10] was used to measure alignment between the resulting time warped input motion A and the target motion T. The same subset of joints used when time warping the motions, were parameterised as joint trajectories (i.e., the $\Re 3$ translation of a one unit length vector along the joints orientation) and of each corresponding joint axis correlated using Kendall Tau rank correlation in Equation (2), where j is a joint in subset of joints $J = \{J_0, J_1 ... J_k\}$ and a is an axis in $\{x, y, z\}$. The online FP-DTW algorithms proposed in this paper are not expected to perform better than DTW, however, the alignment scores achieved by a simpler offline time warping algorithm, UTW,[11] will be used as a performance benchmark.

$$r^t(A, T) = \frac{\sum_{j=1}^{k} \sum_{a=1}^{3} r^t(A_{j,a}, T_{j,a})}{3k}. \tag{2}$$

The second performance test measures the average deviation between the alignment paths plotted by DTW and the FP-DTW algorithm in frames, using Equation (3), where P^a and P^b represent the paths plotted by DTW and the FP-DTW algorithm, respectively. As character interactions which are mistimed by less than 150 ms[12] are typically imperceptible, a benchmark of the same period, 18 frames, will be used as a benchmark.

$$D(P^a, P^b) = \frac{\sum_{q=1}^{n} |P_q^a - P_q^b|}{n}. \tag{3}$$

5 | RESULTS

Figures 5 and 6 show the results of the similarity metric and deviation performance tests, respectively. Time warping algorithms achieving a higher result in the similarity metric or lower result in the deviation test are considered better. Algorithms exceeding the benchmarks set for either test, are of particular interest and should be considered potentially usable in real-world applications. To exceed the benchmark the median result for the algorithm needs to be above the similarity results achieved by UTW, shown as a red line in Figure 5, or below the perceptible deviation from the

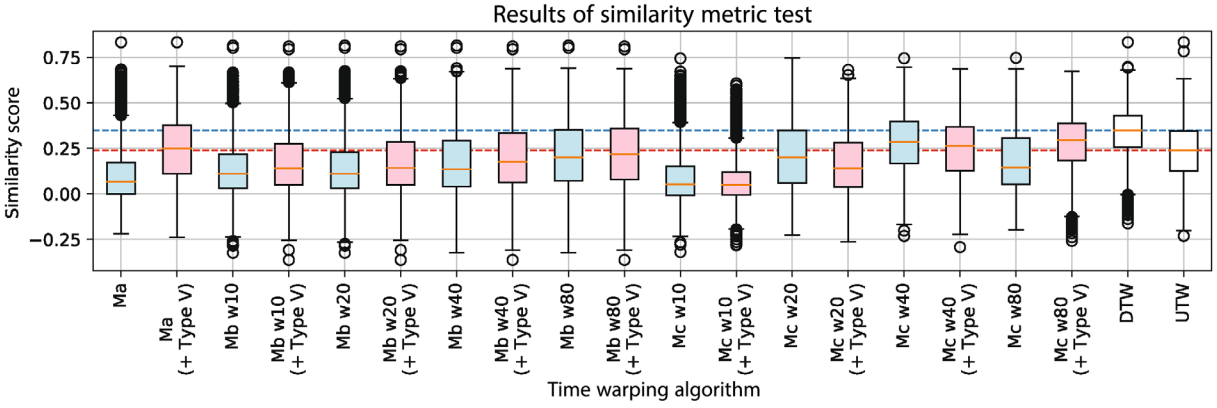


FIGURE 5 The alignment scores achieved be each time warping algorithm.

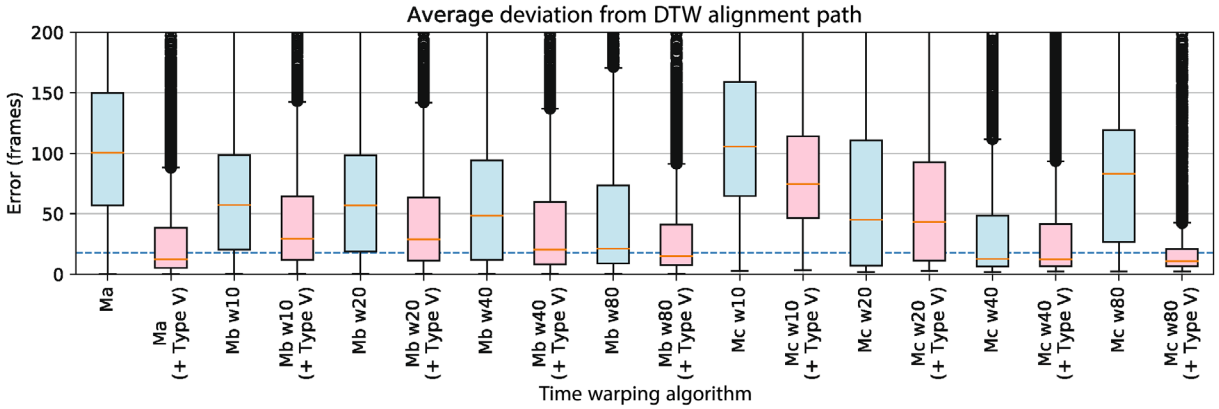


FIGURE 6 The average deviation of each alignment path plotted by a time warping algorithm from the dynamic time warping alignment.

TABLE 1 Median alignment scores for time warps performed using Method C, on motions sampled at 60 and 120 Hz.

	Window size				
Sample rate (Hz)	**10**	**20**	**40**	**80**	**Universal time warping**
60	0.133	0.281	0.178	0.068	0.242
120	0.053	0.200	0.285	0.146	0.240

DTW alignment path (average of 18 frames), shown as a blue line in Figure 6. Ma, Mb, and Mc, denote FP-DTW algorithms based on Methods A, B, and C respectively. The window is denoted using "w." The results shown in pink, and denoted with "+ Type V," are for FP-DTW algorithms with a Type V constraint applied, which is discussed in more detail in Section 6.

Method C with a window size of 40, *Mc w40*, is the only online FP-DTW algorithm to exceed either of the benchmarks, without a constraint. As expect Method A, *Ma* was on of the worse performing algorithms in both tests, as it was using a more naive approach.

There is an interesting contrast in the impact of windows size, between Methods B and C. While the performance of Method B continued as the windows size increased, Method C hit an optimal point at window size of 40, with the performance dropping at window size 80. This difference is likely to stem from the less constrained approach Method C uses to select input frames in contrast method B. While this allows Method C to achieve a better alignment, it also seems to make it more sensitive to the cyclic frequency of the movement being aligned. To confirm this, the motions in the data-set were down sampled to 60 Hz, then time warped using Method C. The results in Table 1 show the optimal window size for Method C shifting down to 20 frames, when applied to motion sampled at 60 Hz, supporting this theory.

Figure 7 visualizes the alignment path plotted by different time warping algorithms. Each heat-map shows an aggregate of all the paths plotted by given time warping algorithm, for every pair of input and output motions in the dataset. An optimal alignment algorithm will cluster alignment paths around the diagonal as shown with the DTW algorithm. Hot areas under the diagonal indicate algorithms incorrectly sticking on frames during time warping, while hot areas above the diagonal indicate FP-DTW algorithms incorrectly skipping frames.

The plots in Figure 7 for Method B algorithms, *Mb w10* and *Mb w80*, show that it has a tendency to incorrectly stick input frames when time warping, with different window sizes having a limited impact. The aggregate plots of Method C, show window size having much more impact on characteristics of the alignments produced using this Method. Similar to Method B, a small window size of 10 frames, *Mc w10*, causes Method C to incorrectly stick on input frames, while a large window size of 80 frames, *Mc w80*, causes Method C to incorrect skip input frames. Plot *Mc w40* shows that Method C with a window size of 40 is a more optimal balance between skipping and sticking during alignment.

Table 2 shows the average time each time warping algorithm took to process a single frame in microseconds (μs). The algorithms were run on a single processing core running between 3.58 and 3.95 GHz. Given a sample rate of 120 Hz, a per frame processing time significantly less than $(1 \times 10^6)/120$ or 8333 μs is required. Without further optimization Table 2 shows that a window size of 80 would not be feasible for use with a sample rate of 120 Hz. The exponential relationship between windows size and processing time, that is characteristic of DTW, can be clearly seen in these results. The lower than expected processing time of Method C with a window size of 80 (*Mc w80*), is an anomaly, due to the algorithm running out of input frames to match to, early in the alignment process, therefore skipping any attempt align to the remaining target frames.

A visualization of the alignment paths plotted by each time warping algorithm, when aligning two walk motions, can be seen in Figure 8, along with the motion curves for the right hip and character poses from a side elevation, to visualize

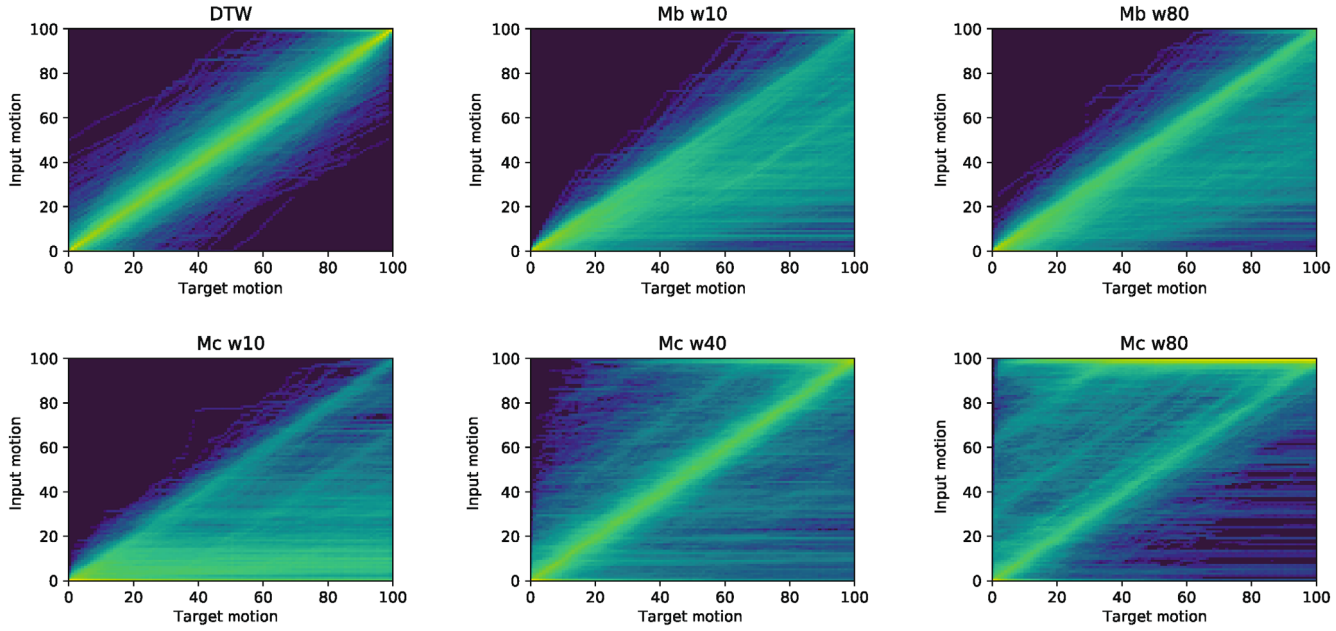


FIGURE 7 Heat maps visualizing the alignment paths plotted by each algorithm.

TABLE 2 The mean time each algorithm takes to process a single frame of motion data.

	Window size (w)			
	10	**20**	**40**	**80**
Method B	856μs	1654μs	4806μs	16945μs
Method C	782μs	1583μs	4764μs	10106μs

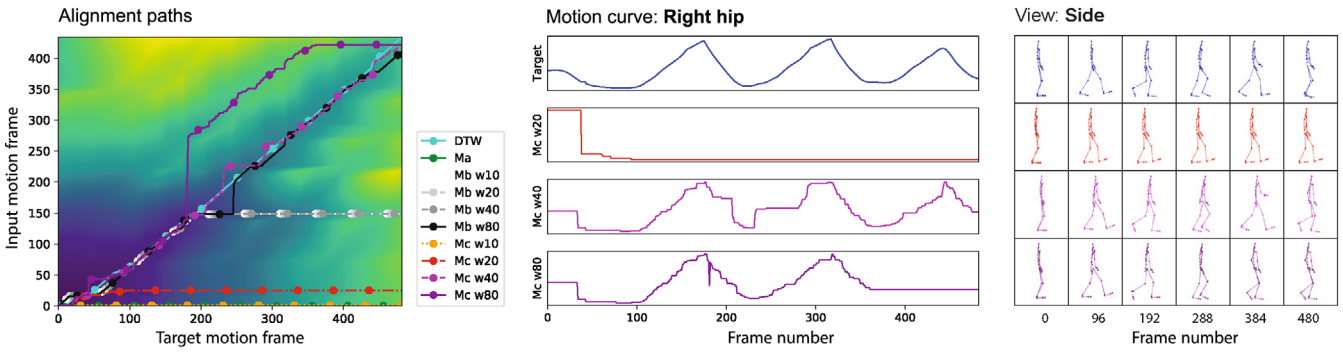


FIGURE 8 The alignments plotted by time warping algorithms, without constraints or penalties applied, when aligning a walk motion. Videos resulting motions for this motion pair, sample number 3239 can be seen online.[13]

the resulting aligned motions. The alignment paths are overlaid over a heat map of the accumulated cost matrix. Better quality alignments will follow the darker minimal cost areas of the heat map, closely following the path plotted by DTW. The target motion being aligned to has been included as the top set of motion curves and character poses, to compare against.

The alignments plotted by algorithms *Mb w80*, *Mc w20*, *Mc w40* and *Mc w80*, deviated from path plotted DTW by an average of 8, 177, 62, and 6 frames, respectively. With only algorithms *Mb w80* and *Mc w40* with errors (or deviations) below the 18 frame benchmark deviation. While *Mc w40* plots a good alignment, the motion curve is jittery. *Mc w80* skips a walk cycle, while the *Mc w20* complete fails, sticking on an input frame early in the alignment. Rendered examples of motions aligned using the FP-DTW algorithms in this study are available online.[13]

6 | CONTINUITY CONSTRAINTS

A common approach to optimizing the performance on a time warping algorithm is to impose a local continuity constraint, which limits the alignment paths that can be plotted over a set number of steps.[6] Previous research has shown the Type V constraint to be an optimal choice for real-time applications.[4]

To facilitate online forward plotting, the Type V constraint was adapted as shown in Figure 9. As the constraint is designed to be used in a DTW algorithm in which alignment paths are plotted backwards, the constraint needs to be reversed as shown in Figure 9ii. The vertical points that can reached, $\{P_3, P_4, P_5\}$, can be constrained by limiting the window of input frames the algorithm can select from, when choosing a frame to map. However, constraining the horizontal points which can be reached to $\{P_1, P_2,\}$, requires the constraint to be maintained between processing different target frames. To facilitate this, states are used to represent which input frames can be selected for alignment, on a given frame, as shown in Figure 9iii. Not only does the state determine which frames are allowed to be selected, but also which state the algorithm will be in when processing the next target frame. These states can effectively represented as a table as shown in 9iv, to simplify implementing the constraint logic. This same approach can be used to implement other local continuity constraints.

The impact of implementing the Type V constraint can be seen in Figures 5 and 6, which shows the results of the similarity and alignment path deviation performance tests, for each FP-DTW algorithm, both with and without the constraint applied. The impact of the constraint is not consistent across all algorithms. Algorithms which perform well without the constraint or used a small window size, did not appear to benefit greatly from the constraint.

The two algorithms which were most improved by the constraint are *Ma* and *Mc w80*, improving their performance sufficiently enough to satisfy both the UTW alignment and frame deviation benchmarks. This is particularly useful, as this now provides a choice of FP-DTW algorithms which could potentially be used for real-world applications. While the performance of all algorithms based on Methods A and B, are improved by the implementation of a constraint, *Mc w80* was the only Method C algorithm that improved.

The aggregated alignments plotted in Figure 10 visualize the impact of constraints. The alignment paths of *Ma* algorithm are improved, with a stronger highlight along the 1:1 diagonal slope and no alignment paths sticking on the first few frames of the input motion. The performance of the *Mc w10* algorithm was not improved by the implementation of a constraint, the diagonal highlight is in the wrong place, shallower than the 1:1 diagonal slope. Although the constraint has stopped *Mc w10* from sticking on a single input frame, the algorithm is still often plotting an alignment path which

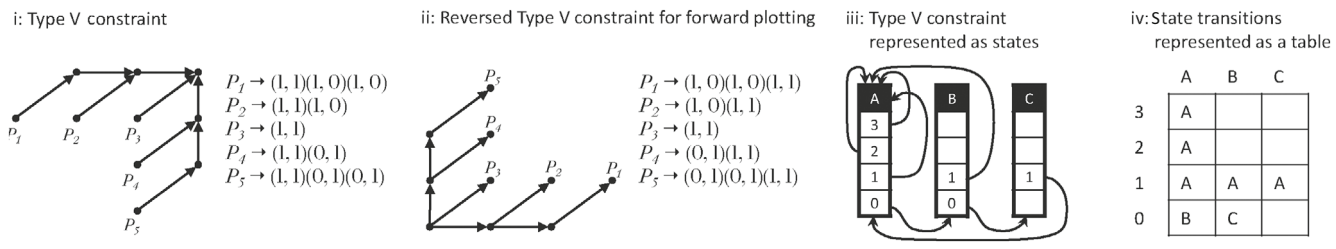


FIGURE 9 Adaptation of Type V constraint to facilitate with forward plotting of alignment paths.

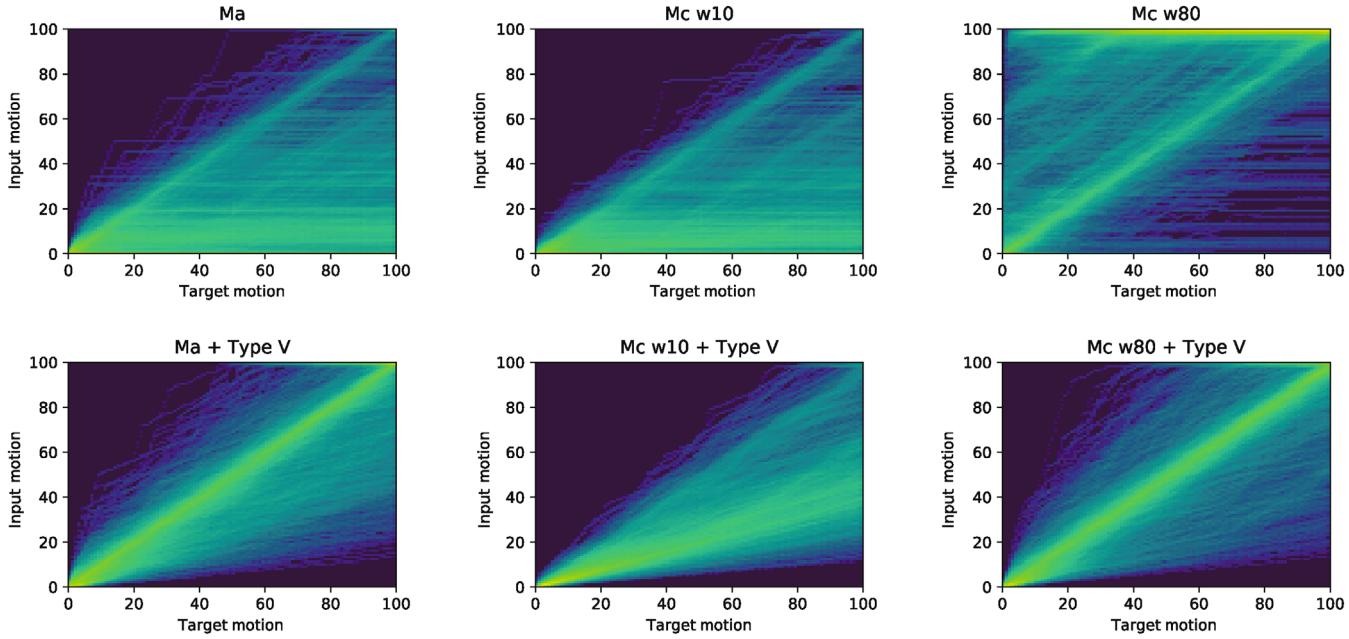


FIGURE 10 A comparison between the aggregate plots of time warping algorithms with and without the Type V constraint applied.

follows the shallowest diagonal that the constraint will allow. The *Mc w80* algorithm was also improved considerably by the implementation of a constraint, as well as showing a stronger highlight along the 1:1 diagonal slope, fewer time warps are incorrectly skipping input frame during alignment. The *Mc w80 + Type V* algorithm is exhibiting that same balance between sticking and skipping input frames during alignment, as shown by *Mc w40*.

7 | CONCLUSIONS

This study has shown that forward plotting of a temporal alignment, is a potentially viable solution for scenarios in which backwards plotting is not suitable or produces undesirable results. For example, timing a virtual character's performance to coordinate with that of a live actor.

Three of the FP-DTW algorithms proposed in the paper satisfied both the UTW alignment and deviation from DTW benchmarks, *Ma + Type V*, *Mc w40*, and *Mc w80 + Type V*. These algorithms can be considered to perform well enough to potentially support real-world applications, although further research evaluating their performance in real-world applications is required.

Method C is clearly sensitive to the relationship between the frequency of any movement cycles within the motion being performed and the sample rate at which the motion is being recorded. This should be taken into account when using Method C, by restricting the size of the forecast window to less than half the period of the movement cycle being captured.

Within this study, a subset of joints considered most pertinent to everyday motions was used, with each of these joints equally weighted within the similarity metrics and cost functions used in this study. Approaches have been proposed for preprocess motion data to determine the optimal weighting of joints for a given motion. In a live performance scenario, in which a pre-recorded motion is being align to the performance of a live actor, the prerecorded motion could be pre-processed to determine the optimal joint weightings before time warping.

There is potential to improve the computational performance of these algorithms by implementing a multi-threaded approach to calculating the cost matrix in Methods B and C. Other approaches to explore are down sampling cost matrix or imposing a global constraint. A global constraint would restrict the area in which a path can be plotted within the cost matrix, reducing the number of cells which need to be calculated.

Further research needs to be performed, implementing and evaluating the performance of proposed FP-DTW algorithms in a real-world scenario. In particular, consideration needs to be given as to how these algorithms can be modified to cope with dropped frames, if required.

There are a number of potential applications for FP-DTW in real-time or live production scenarios. Recording the performances of both virtual character(s) and live actor(s) together, during preproduction, would allow FP-DTW to time warp the pre-recorded performances by aligning them to that of a live actor. There are a number of markerless motion capture solutions, which can be used to discreetly capture the motion of an actor to facilitate this.

ORCID

Mathew Randall https://orcid.org/0009-0000-2869-1008
Carlo Harvey https://orcid.org/0000-0002-4809-1592
Ian Williams https://orcid.org/0000-0002-0651-0963

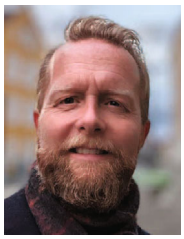
REFERENCES

1. Sakoe H, Chiba S. Dynamic programming algorithm optimization for spoken word recognition. IEEE Trans Acoust Speech Signal Process. 1978;26(1):43–9.
2. Tormene P, Giorgino T, Quaglini S, Stefanelli M. Matching incomplete time series with dynamic time warping: an algorithm and an application to post-stroke rehabilitation. Artif Intell Med. 2009;45(1):11–34.
3. Dixon S. Live tracking of musical performances using on-line time warping. In: Proceedings of the 8th International Conference on Digital Audio Effects (DAFx'05). Madrid, Spain. 2005 Sep 20-22. p. 97.
4. Macrae R, Dixon S. Accurate real-time windowed time warping. In: Proceedings of the 11th International Society for Music Information Retrieval Conference, ISMIR 2010. Utrecht, Netherlands. 2010 Aug 9-13. p. 423–428.
5. Guerra-Filho G, Bhatia H. A comparison and evaluation of motion indexing techniques. In: Allbeck JM, Faloutsos P, editors. Motion in Games. MIG 2011. Lecture notes in computer science. Volume 7060. Berlin: Springer; 2011. p. 436–47.
6. Rabiner L, Juang BH. Fundamentals of speech recognition. Hoboken, NJ: Prentice-Hall, Inc.; 1993.
7. Stakem F, AlRegib G. An adaptive approach to exponential smoothing for CVE state prediction. In: IMMERSCOM '09: Proceedings of the 2nd International Conference on Immersive Telecommunications. Berkely, CA, USA. 2009. p. 1–6.
8. Randall M. Motion capture dataset of single person actions. 2022. Available from. https://github.com/matRandall/Mocap_SinglePersonActions/. Accessed 7 Feb 2023.
9. Müller M, Röder T, Clausen M, Eberhardt B, Krüger B, Weber A. Mocap database hdm05. Inst Inform II Univ Bonn. 2007;2(7):1–32.
10. Etemad SA, Arya A. Correlation-optimized time warping for motion. Visual Comput. 2015;31(12):1569–86.
11. Fu AWC, Keogh E, Lau LYH, Ratanamahatana CA, Wong RCW. Scaling and time warping in time series querying. VLDB J. 2008;17:899–921.
12. Hoyet L, McDonnell R, O'Sullivan C. Push it real: perceiving causality in virtual interactions. ACM Trans on Graph. 2012;31(4):1–9.
13. Randall M. Online time warped motions. 2022. Available from. https://github.com/matRandall/exampleOnlineTimewarpedMotions/. Accessed 7 Feb 2023.

AUTHOR BIOGRAPHIES

Mathew Randall is a Senior Lecturer at Birmingham City University in the United Kingdom. He is a member of the Graphics and Vision Research group, within DMTLabs, where he is currently completing his PhD and specializes in motion capture, virtual production and the analysis and manipulation of motion capture data. He is a senior fellow of the Higher Education Academy, lecturing in visual effects and computer graphics.

Carlo Harvey is an Associate Professor in Computer Games Technology at Birmingham City University, in the School of Computing and Digital Technology. He teaches C++ for Games, Computer Graphics and Artificial Intelligence / Machine Learning. He is the Director of Future Games and Graphics overseeing the research arm for Games and Graphics within the DMTLab at BCU.



Ian Williams is Professor of visual computing and Head of the Digital Media Technology Lab (DMT Lab) at Birmingham City University. He gained his PhD from Manchester Metropolitan University in 2008 in low-level feature analysis and Ai for multiple-scale biomedical image analysis. His current research interests center on the visual and interactive computing domains, with a key emphasis on creating novel methods, theories, and paradigms for interacting with, and improving on, the Quality of Experience for users of Augmented Reality (AR) and Virtual Reality (VR).

How to cite this article: Randall M, Harvey C, Williams I. Online alignment of human motion using forward plotting-dynamic time warping. Comput Anim Virtual Worlds. 2023;34(3):e2166. https://doi.org/10.1002/cav.2166