2017 IEEE International Symposium on Mixed and Augmented Reality Adjunct Proceedings

[POSTER] A Predictive Approach to On-line Time Warping of Motion

Mathew Randall* Ian Williams† Cham Athwal‡

DMT Lab, Birmingham City University

Abstract

The paper presents a novel approach to real-time temporal alignment of motion sequences, called On-line Predictive Warping (OPW) and considers potential uses in interactive applications. The approach develops on the methods of aligning motions based on least cost, used in dynamic time warping (DTW), with the short term predictions of smoothing algorithms, in an iterative step through approach. The approach allows a recorded motion sequence to be warped to align it with a users motion as it is being captured. The paper demonstrates the potential feasibility of the approach to support applications in MR and VR, allowing virtual characters to perform and interact with users and live actors in a variety of rehearsal, training, visualisation and performance scenarios.

Keywords: On-line time warping, character animation, virtual production

Index Terms: I.3.6 [Computer Graphics]: Methodology and Techniques—Interaction Techniques; I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation

1 Introduction

Motion sequences, are motions or actions of people recorded using a motion capture system. When working with motion sequences in mixed reality (MR) and virtual reality (VR) applications, there can be a need to manipulate, edit and combine them in real time, often to allow a virtual character to interact with its environment or the user. To accurately blend or combine motion sequences it is first necessary to align their temporal features. For example, in order to combine elements of two different walk sequences, the motion sequences would need to be temporally aligned, such that the timing of both walk cycles match, with feet making contact with the ground at the same time within each sequence. Once motion sequences are aligned in this way, they can be blended or stitched together to form motion sequences that are larger than the original capture volume [11] or have stylistic features combined to create new motion sequences [4].

A number of off-line techniques have been proposed such as dynamic time warping (DTW) [14] and correlation-optimized time warping (CoTW) [6], based on the analysis of complete motion sequences. MR and VR applications, where there is a need to interact with a user's motions in real time, require an on-line approach that works without knowledge of the users future actions. This paper proposes a novel real time solution to temporal alignment, called On-line Predictive Warping (OPW), that aligns pre-recorded motion sequences with user motions in real time, potentially facilitating new and novel approaches to user interaction, with virtual characters being able time their performance to match a user's actions.

A real time approach has been proposed which utilises machine learning techniques to interactively stylise a user's motion as it is being captured [18], by matching corresponding frames in a dataset. This approach, however, does not provide a robust monotonic temporal alignment of a specific motion sequence with a user's captured motion. A more robust alignment is not only required to support manipulation techniques, but importantly may have the potential to support user interaction in virtual production and other MR and VR applications.

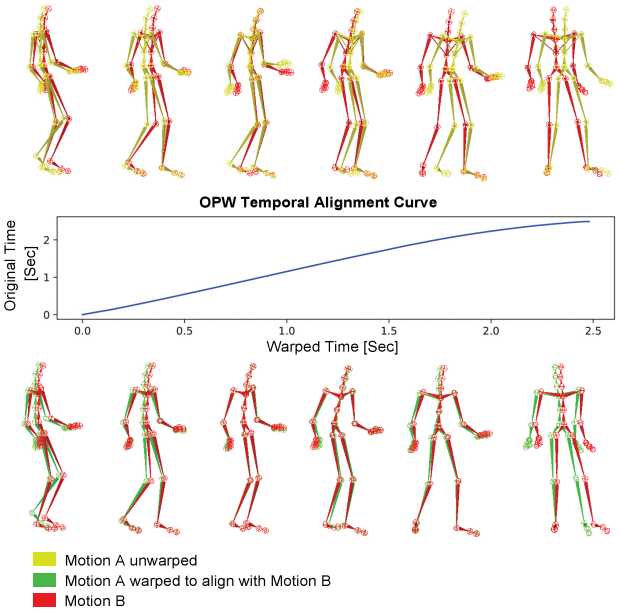


Figure 1: Demonstration of OPW approach being used to temporally align two handshake motions

Virtual production is a term within film and visual effects, which refers to a variety of technologies, some based around VR and MR, that allow CGI elements to be directly manipulated in real time [13]. These can be used in pre-production stages, to rehearse and visualise scenes, as well as during production to visualise how CGI elements will fit within a live action shot. For example, films such as James Cameron's *Avatar* utilised camera tracking and real time rendering to place virtual elements into a camera's view, enabling the actors, crew and director to see how the live action and virtual elements will fit together. Since then, the Dreamspace project has developed a number of virtual production tools including Virtual Production Editing Tools (VPET) which allow CGI elements to be manipulated live as they are being viewed through the camera [16].

There is currently interest in the application of VR within pre-production. Virtual camera systems have been developed to provide a physical manifestation of the virtual camera [1], while the potential applications of VR for scouting locations and technical planning of film productions is also being considered [7].

Bourville, et al [3] tested the use of VR in acting rehearsals, either to allow actors that cannot be in the same place to rehearse together or to allow rehearsals with virtual characters. Their system

*e-mail: mathew.randall@bcu.ac.uk
†e-mail:ian.williams@bcu.ac.uk
‡e-mail:cham.athwal@bcu.ac.uk

978-0-7695-6327-5/17 $31.00 © 2017 IEEE
DOI 10.1109/ISMAR-Adjunct.2017.52

uses an engine called #SEVEN [5] which senses events within an actor's performance which are used to trigger changes in the virtual environment.

While virtual production tools such as VPET allow CGI elements to be manipulated with ease in real time and #SEVEN enables virtual elements to be triggered by an actor's performance, the actor is still for the most part, having to coordinate their performance to correspond with virtual elements. There is a need to reverse or equalise this relationship and allow virtual elements, such as the performance of a virtual character, to automatically align and co-ordinate their performance with that of a live actor. A real time approach to temporal alignment, would solve the temporal aspects of this problem and potentially support real time interaction with virtual characters in other VR and MR applications. The OPW approach proposed in this paper develops on the step-through least cost approach of DTW, with the predictive forecasting of smoothing algorithms, to align a motion sequence with a user's captured motion in real time.

2 Related Works

Real time approaches to temporal alignment have been proposed which utilise machine learning algorithms. In [12], support vector machines (SVM) are used to classify different phases of a hand shake motion. They compared the performance of SVM with K-nearest neighbour (KNN), decision trees and naïve Bayes classifiers, and found that SVM performed the best with the most accurate prediction rates. As the datasets are not very generalisable, large sets of training data would have to be created and labelled for each distinct action, making this approach impractical for use with the unique motions typically found in film production.

Approaches have been proposed which utilise machine learning algorithms for transferring motion styles, using linear time invariant (LTI) models [9] and KNN [18]. Again large datasets are used, although in a more generalised manner, removing the need for specific examples to be captured. As each frame or pose of the input motion is matched to poses in a large dataset of motion sequences with limited constraint, the resulting matching frames cannot be used to support temporal alignment. In [9] temporal and spatial displacement warp curves are created off-line, to produce a translation model which is used to align the two motions in real time. While this is a real time solution it is not an on-line solution. The need to create a translation model for each combination of motions, using an off-line process, means it cannot be applied to a motion as it is being captured.

DTW matches frames or samples of one motion sequence to another, with the objective of minimising the differences between the two sequences, a more detailed explanation of DTW is provided by [14]. The difference minimisation is performed in two steps, first the difference in the values of the two time series is determined producing a cost matrix, then an alignment path is plotted through the lowest values in the cost matrix, minimising the overall alignment cost.

A path is created by stepping through the cost matrix and plotting continuous path of aligned frames or samples, only going forward in time. Once a path has been determined a smooth curve needs to be plotted along the path, typically a B-spline.

Using curves to define temporal and spatial displacements was proposed in [17], this allowed a smooth deformation of keyframed or captured motion using curves, rather than editing each frame. This approach has been widely implemented using piecewise Bézier curves, often referred to as F-Curves within software documentation. In [4], DTW is used to create B-spline to temporally align motion sequences allowing the style of a motion to be transferred from one motion to another, while [11] combines the same technique with spatial alignment, to allow motion sequences to be blended together to synthesise new motion sequences.

The OPW approach requires a method of predicting near future values in a time series, as the alignment of time series Y will be based on the predicted value of time series X. The implementation of different smoothing algorithms: dead reckoning, exponential smoothing and Holt's double exponential smoothing, are discussed in [15], where their performance in predicting the movement of user's hand is compared. They showed that exponential smoothing performed similarly to dead reckoning, but also proposed a new technique called adaptive exponential smoothing, in which double exponential smoothing parameters automatically adapt themselves. This approach was shown to cope better with sudden twitch movements than dead reckoning and exponential smoothing.

3 Approach

An on-line approach to temporal alignment aims to temporally align points in a known time series A, to another time series B as it is being captured. Therefore aligning without having knowledge of the remaining values in the time series B.

The OPW approach proposed in this paper combines the stepped and least cost approach of DTW with the near future prediction of smoothing algorithms, creating an on-line approach in which the temporal alignment at each time step, is based on the predicted value of the time series being captured X and the known values of time series being aligned Y. At each time step t, a prediction of the time series X at the next time step x_{t+1}, is compared with values $y_{t+i}, i \in \{0..2\}$ in the time series Y. Equation 1 is used to determine which of these values in time series Y has the smallest difference to predicted value in time series X, and therefore is the closest match.

$$s = \underset{i \in \{0..2\}}{\arg\min} (\hat{x}_{t+1} - y_{t+i})^2 \tag{1}$$

The result of equation 1 determines if the remainder of time series Y needs to be expanded or contracted to align it with time series X as follows:

- $s = 0$: Y needs to be slowed down to align with X by expanding the remaining duration of Y.
- $s = 1$: Y and X are in temporal alignment so the duration of Y does not need to be changed this step.
- $s = 2$: Y needs to be sped-up to align with X by reducing remaining duration of Y.

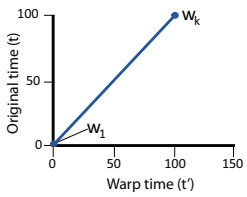
The duration of time series Y actually remains unaltered, instead a time warp curve w is used to define a warped time that is a function of the original time $t' = f(t)$. As shown in table 1 the time warp curve starts as a straight line in which $t' = t$. At each time step t, if time series Y needs to be expanded or contracted, the warp curve is split at time t, adding a new keyframe, and the warped time t' at the end of the warp curve w_k is updated using the following equation 2. This process of constructing the time warp curve is demonstrated in table 1 and its implementation can be seen in algorithm 1.

$$Y_{k'} = Y_{k'} - (Y_{k'} - x_t)\gamma(s-1) \tag{2}$$

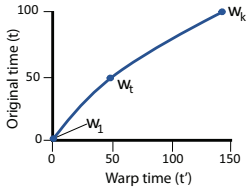
γ is a warping parameter, which satisfies $0 < \gamma < 1$, it determines how much the end of the time warp curve is moved each step, the greater the value of γ the more aggressively the warp is altered at each step.

Piecewise Bézier curves are often used to create key-frame animation curves called F-curves, with each key-frame representing the end of one Bézier curve and the start of another. Unlike other smooth curve plotting algorithms such as b-splines which go near control points, a piecewise Bézier curve will plot through these points, making it ideal for this type of application. The time warp

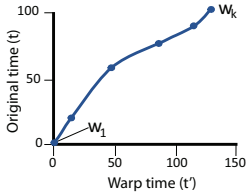
Table 1: Steps in construction of time warp curve



The piecewise Bézier warp curve before any warping is performed, is a single curve forming a straight line. Each point in w maps t' to t. As $t' = t$ the curve causes no temporal displacement.



At step t the duration of the motion needs to be expanded to maintain temporal alignment. The warp curve w is split at into two Bézier curves at time t, by adding point w_t at (t',t). The end of the motion is warped by updating the position w_k in t' using the equation 2.



Once the real time time warping process has been completed, the result is a complete time warp curve composed of multiple Bézier curves.

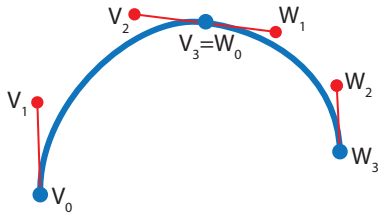


Figure 2: Two Bézier curves, V and W, jointed with G^1 continuity.

is implemented using F-curves within Autodesk MotionBuilder, as the software is able to interpolate these curves to time warp motion sequences in real time.

Each Bézier curve is specified using four vectors $P_0 ... P_3$ and plotted using equation 3. As shown in figure 2, P_0 and P_3 denote the start and end of the curve, while P_1 and P_2 are control points which influence the curve but are not plotted through.

$$B(t) = (1-t)^3 P_0 + 3t(1-t)^2 P_1 + 3t^2(1-t)P_2 + t^3 P_3, t \in (0,1) \quad (3)$$

When defining F-curves in MotionBuilder it automatically places the control points P_1 and P_2 for each curve, such that the curves are smoothly joined with a G_1 level of continuity as shown in figure 2. G_1 continuity requires not only the positions of V_3 and W_0 to match, but also their derivatives, such that $V'(1) = W'(0)$. Ensuring a smooth time warp is important, as sharp directional changes in a warp would cause jaring and unnatural looking temporal changes when applied to a motion sequence. To achieve G_1 continuity the slope between V_2 and V_3 must equal the slop between W_0 and W_1, as seen in figure 2. Approaches to achieving geometric continuity are discussed in [2].

The time warp curve must be applied to time series Y as it is being constructed. This allows transformations to the time warp curve in one step, to impact on how time series Y is read in the proceeding steps. Values $y_{t'+i\omega}, i \in \{0..2\}$ in time series Y are determined using the time warp curve, with time (ω) between each step or sample.

$$s = \underset{i \in \{0..2\}}{\arg\min} (\hat{x}_{t+\omega} - y_{t'+i\omega})^2 \quad (4)$$

The sample period ω and warp value γ are parameters that can be adjusted to optimise the warping process. This research aims to test the overall feasibility of this approach and considers effective values for ω and γ.

Algorithm 1: The OPW step-through alignment algorithm.

$p \leftarrow \{0,0,0\}$ // set total costs to zero;
$T' \leftarrow \{t'-\omega, t', t'+\omega, t'+\omega 2\}$ // get warped sample times;
$t \in T \leftarrow f(t' \in T')$ // unwarp sample times ;
for $j \in Joints$ **do**
  $x \leftarrow X_j, y \leftarrow Y_j$ // load joints ;
  $b_1 = x_{T_1'}, b_2 = x_{T_2'}$ // get current and previous samples of motion X in warped time;
  $b_3 = b_2 + (b_2 - b_1)$ // forecast motion X in the next sample ;
  $a_1 = y_{T_2}, a_2 = x_{T_3}, a_3 = x_{T_4}$ // get the current and future time samples of motion Y in unwarped time ;
  $c_n = b_3 - a_n, n \in \{1..3\}$ // calculate distance between forecast of motion X and current and future samples of motion Y;
  $p_n = p_n + c_n$ // add joint distances to total costs;
end
if $p_1 > p_2$ ***and*** $p1 > p_3$ **then**
  $w_k - 1 \leftarrow \{t', t\}$ // add new key to warp curve at current time;
  $w_k \leftarrow \{w_k + \gamma, Y_k\}$ // move last key to expand the time warp curve;
else if $p_3 > p_1$ ***and*** $p_3 > p_2$ **then**
  $w_k - 1 \leftarrow \{t', t\}$ // add new key to warp curve at current time;
  $w_k \leftarrow \{w_k - \gamma, Y_k\}$ // move last key to shrink the time warp curve;

4 METHOD

The approach was implemented using the dead reckoning method for predicting values and the Euclidean distance for measuring the alignment cost. This implementation has the least computational cost and is in line with the implementation of DTW in [4] and [9].

In this case the values in time series X and Y represent motion sequences plotted onto a control rig, with 16 skeletal joints in a three dimensional space. The alignment cost was measured using the position of the following 16 joints in three dimensional space: head, center hip, right hip, left hip, middle chest, upper chest, left shoulder, right shoulder, left elbow, right elbow, left wrist, right wrist, left knee, right knee, left ankle, right ankle. The cost is based on the sum of straight line distance between the corresponding joints in each motion sequence. Equation 5 determines the total alignment cost between motion sequence A at time t' and sequence B at time t based on k joints in 3 dimensional space.

$$c(A_{t'}, B_t) = \sum_{n=1}^{k} \sqrt{\sum_{m=1}^{3} (A[t',n]_m - B[t,n]_m)^2} \quad (5)$$

The motion sequences used in the test were downloaded from the Carnegie Mellon University mocap database, also used by

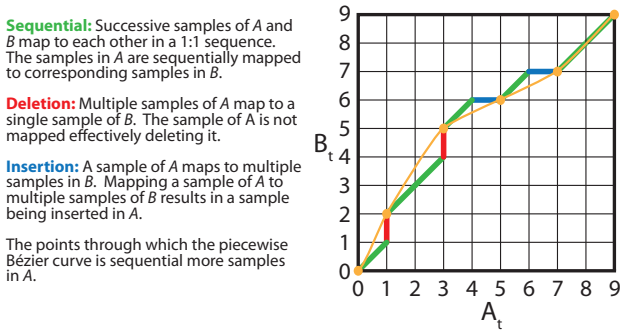


Figure 3: Building a smooth Piecewise Bézier curve through a DTW path aligning A to B.

[18]. Three contrasting motions were used which involved different forms of opposing two character interaction: *Hand Gestures*, a non-contact conversational interaction with hand gestures and eye contact; *Hand Shake* with approach, contact and separation phases of interaction and *Arm Wrestle* with a prolonged period of contact. To focus on how our OPW approach performed with each style of interaction, the motion sequences were cropped at 2.5 seconds to eliminate any impact of variations in temporal length.

The motion sequences were imported into *Autodesk MotionBuilder* and the plotted onto a control rig. As this approach would need to be implemented on a control rig, to support spatial alignment using inverse kinematics (IK), it was felt more appropriate to implement the tests this way rather then directly onto skeleton.

During all tests, the predictive algorithm was run at 25 steps or iterations per second. This is representative of the capture rates of commercial devices such as the *Microsoft Kinect*, which are often used in interactive MR and VR applications.

The accuracy of the OPW approach was assessed by warping a motion sequence (A) using a predefined example time warp curve, resulting in a warped motion (B). OPW will be used to warp the original motion (A) to temporally align it with the warped motion sequence (B). The accuracy of the time warp curve produced by the real time approach will be measured by comparing it against the predefined example time warp curve. This approach allows the temporal difference between the two motions being aligned to be controlled and adjusted for different tests. As temporal errors of more than 150ms are acceptable to viewers less than 50% of the time[8], this will be used a benchmark for evaluating the performance of OPW.

The performance of the OPW approach was compared against an off-line DTW approach, applied to align the same motion sequences. The time warp curve produced by this method will be measured against the same predefined time warp curve, allowing the accuracy of both techniques to be directly compared. The curves were sampled at 50Hz and exported from (MotionBuilder) and analysed using Python.

To support an accurate comparison all time warp curves are constructed using piecewise Bézier curves, therefore, there is a need to construct smooth piecewise Bézier curve based on the least cost path determined by the off-line DTW algorithm. As discussed earlier the piecewise Bézier curve is constructed by specifying a series of (x,y) points, each of which define the start and end of a Bézier curve. Setting an alignment point at each step would result in a jerky and not particularly useful time warp curve. A novel approach was devised to create an accurate and smooth curve, by only aligning time steps after 1 or 2 consecutive steps have resulted in either a sample being inserted or deleted and where the next time step is a sequential substitution as describe in figure 3.

5 Results

Figure 4 shows how the OPW approach performed when applied to the three different motions: hand gestures, non-contact; hand shake, momentary contact and arm wrestle continuous contact). The top row of graphs show the example curve used in the test along with the resulting warp curves produced using DTW and OPW. The example curve is a straight line, from 0 to 2.5 in both axis, with the middle offset by 0.25 seconds, creating a warp curve that starts steep, speeding up time and ends shallow, slowing down time. The middle of the curve represents the maximum warping of time, with 1.5 seconds being warped to 1.25 seconds. The result is a temporal difference of 250ms between the original and warped motions in the middle of the motion sequence, but no temporal difference between them at the start and end of the sequence. The γ warp parameter was set to 5% for all tests in figures 4 and 5.

The middle row of graphs in figure 4 show the deviation of the warp curves produced using the DTW and OPW approaches, from the example warp curve. This shows the potential misalignment of the timing of any interaction based on these techniques in milliseconds. The OPW approach performed consistently when applied to all three different motions, but as expected it did not perform as well as the off-line DTW process. The maximum deviation was 121ms is less than the 150ms suggested as acceptable by Hoyet [8]. The average deviation of the OPW approach was between 79 and 69ms.

The bottom row of graphs in figure 4 show the change in the slope of the warp curves between subsequent samples, as a percentage gradient, where a 100% change represents a 90 degree change in angle. This shows the potential smoothness of any motion sequence warped using these curves, as a large change would result in an undesirable jittery playback of the sequence. The OPW approach produces smoother curves as a result of applying a consistent amount of warp at each step.

Figure 5 shows the performance of the DTW and OPW approaches when aligning motions with varying amounts of temporal difference between them. The example warp curve is the same, fast then slow curve, used in the figure 4 tests, but offset in the middle by varying amounts. Each offset value was applied to all three motions and the results combined. The results show that the amount of temporal difference between the two motions has a significant impact on the performance of the OPW approach, which performed worse as the temporal difference increased. The off-line DTW technique was able to perform more consistently.

Figure 6 shows the performance of the OPW approach, with different values for the γ warp parameter. This parameter adjusts the percentage by which the remaining motion being aligned is warped during each step if required. The test were performed using the same example motion used in the tests for figure 4. In these tests the on-line algorithm performed best with $\gamma = 15\%$, this should not be treated as conclusive as there needs to be a greater understanding of this relates to the motion sequences themselves.

Considering that temporal misalignments of over 150ms are undesirable [8], initial test show that OPW was able to cope with a temporal difference of 300ms between two 2500ms motion sequences. This is a conservative analysis of the approaches performance, as figure 6 shows that parameters can be adjusted to further optimise it. If the average deviation was considered rather than the maximum, the approach could be considered to cope with temporal differences of up to 500ms. There needs to be further study of how the ratio between the length of the motion sequences being aligned and amount of temporal difference between them, impacts the performance of OPW. The configuration tested in figure 5 consistently reduced the temporal difference between the motion sequences by approximately 70%. Figure 6 shows that this could be increased to approximately 77% by optimising the γ warp parameter. The γ warp parameter has an optimal setting as setting it too high results in over compensation and suboptimal alignment.

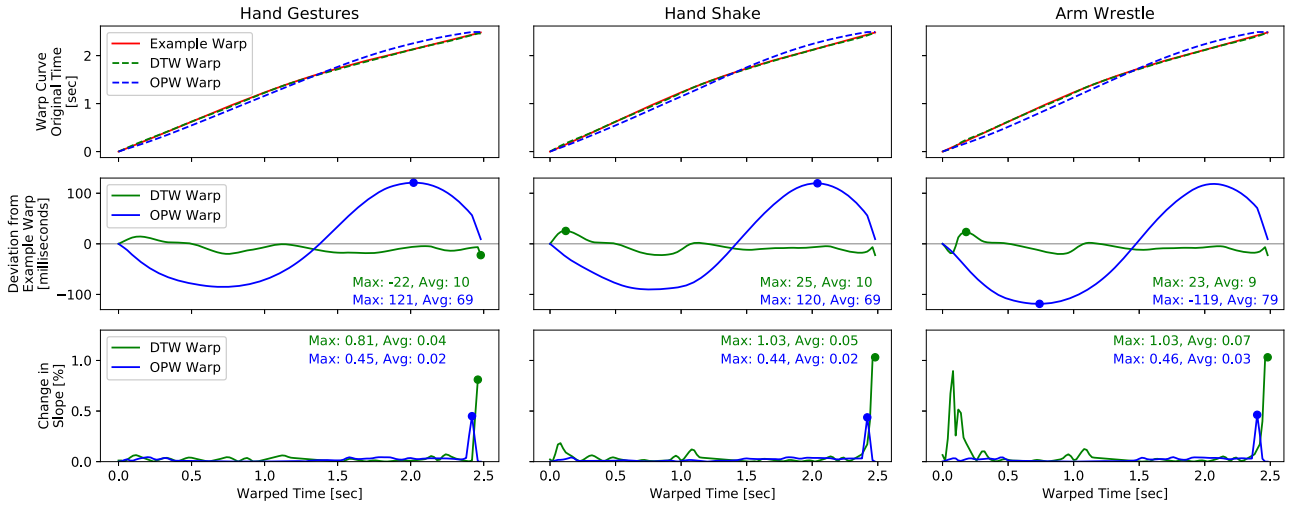


Figure 4: A comparison of predictive and DTW warping methods applied to different motion sequences. Top row shows the resulting time warp curves; middle row shows the deviation of both DTW and OPW warp curves from the example warp curve; bottom shows the percentage of change in the slope of the curves between samples.

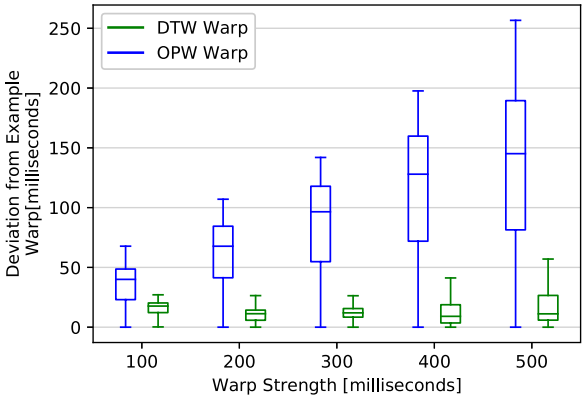


Figure 5: Performance of DTW and OPW warping algorithms when aligning motions with different amounts of temporal deviation

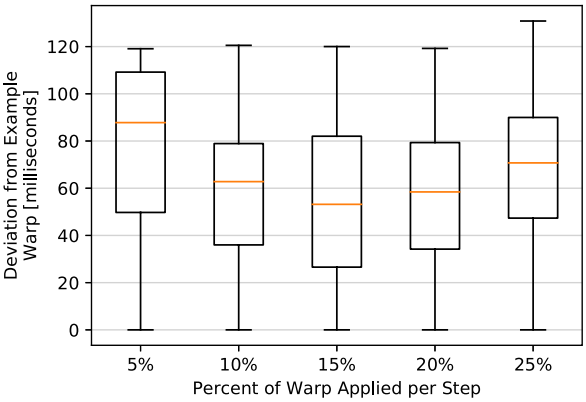


Figure 6: Performance of OPW warping algorithm, when applying different warp γ percentages.

6 Conclusion

This paper presents a method for on-line predictive time warping, demonstrating potential to align deviations between two motions. The OPW approach performs consistently which applied to different motions, giving an average deviation between 69 and 79ms, below the 150ms perceptual suggested by [8].

A potential weakness with OPW, is its dependence on consistent movement in the motion sequences to accurately evaluate their alignment. It may not work consistently if there are periods of standing still. There may also be a need to combine this approach with motion classification, to recognise when a motion has started or which motion has started.

When implemented in MR and VR, the OPW approach will have to cope with additional factors not simulated within these tests. Examples include: spatial as well as temporal differences between motion sequences, different temporal and spatial differences between each pair of corresponding joints in the two motion sequences, aligning motion sequences applied to different size rigs or skeletons. How much each of these factors effects the performance of the OPW approach needs to be evaluated. Basing the alignment cost on the difference between the derivatives of joint positions, rather than the actual joint positions [10] may cope better with these issues.

On-line temporal alignment of a recorded motion sequence with a user's motion, has the potential to allow virtual characters to closely interact with users or performing alongside them. The approach could also be used to evaluate a user's motion against a pre-recorded one, for use in training scenarios. When a recorded motion sequence is temporally aligned with a user, that alignment information can be applied to any other motion sequence that was recorded at the same time. For example, if the hi-five motion of two people is captured at the same time, when one of the motions is aligned with that of a user, the alignment information can be used to align the opposing motion. There are a number of potential applications of this approach in MR and VR including: visualizing and aligning the performance of a virtual character during the production of film visual effects, interaction between real and virtual performers in live performances, real time evaluation of the timing of user's performance in training or rehearsals and creating closer more constant interaction between users and virtual characters in gaming or training scenarios. While a motion capture studio may

be used to record motion sequences, it is expected that markerless motion capture solutions, such as *Microsoft Kinect* would be used to implement on-line user interaction.

The tests in this paper demonstrate that the approach is feasible and able to align motions to within 150ms, considered acceptable in [8]. There are questions regarding the amount of difference between motion sequenced, both temporal and spatial, that this approach is able to cope with, particularly in relation to the spatial and temporal differences it would typically have to deal with when implemented in the example applications discussed.

There is a need for further understanding of the nature and size of the temporal and spatial differences that are likely to occur within the different applications discussed above.

Within this paper the OPW approach has been implemented in a straightforward manner, there is potential to utilise more suitable techniques for measuring alignment costs and forecasting. The approach could be further optimised by automatically adjusting the γ warp parameter and ω sample periods, perhaps based on a feedback loop that measures the accuracy of the alignment as the temporal warp is being constructed and applied in real time.

REFERENCES

[1] G. Balakrishnan and P. Diefenbach. Virtual Cinematography: Beyond Big Studio Production. In *ACM SIGGRAPH 2013 Studio Talks*, SIGGRAPH '13, pages 13:1–13:1, New York, NY, USA, 2013. ACM.

[2] B. A. Barsky and T. D. DeRose. Geometric continuity of parametric curves: constructions of geometrically continuous splines. *IEEE Computer Graphics and Applications*, 10(1):60–68, Jan. 1990.

[3] R. Bouville, V. Gouranton, and B. Arnaldi. Virtual Reality Rehearsals for Acting with Visual Effects. pages 1–8, 2016.

[4] A. Bruderlin and L. Williams. Motion Signal Processing. In *Proceedings of the 22Nd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '95, pages 97–104, New York, NY, USA, 1995. ACM.

[5] G. Claude, V. Gouranton, R. Bouville Berthelot, and B. Arnaldi. Short Paper: #SEVEN, a Sensor Effector Based Scenarios Model for Driving Collaborative Virtual Environment. In T. Nojima, D. Reiners, and O. Staadt, editors, *ICAT-EGVE, International Conference on Artificial Reality and Telexistence, Eurographics Symposium on Virtual Environments*, pages 1–4, Bremen, Germany, Dec. 2014.

[6] S. A. Etemad and A. Arya. Correlation-optimized Time Warping for Motion. *Vis. Comput.*, 31(12):1569–1586, Dec. 2015.

[7] J. Foss. Lessons from learning in virtual environments. *British Journal of Educational Technology*, 40(3):556–560, 2009.

[8] L. Hoyet, R. McDonnell, and C. O'Sullivan. Push It Real: Perceiving Causality in Virtual Interactions. *ACM Trans. Graph.*, 31(4):90:1–90:9, July 2012.

[9] E. Hsu, K. Pulli, and J. Popovi. Style Translation for Human Motion. In *ACM SIGGRAPH 2005 Papers*, SIGGRAPH '05, pages 1082–1089, New York, NY, USA, 2005. ACM.

[10] E. J. Keogh and M. J. Pazzani. Derivative dynamic time warping. In *Proceedings of the 2001 SIAM International Conference on Data Mining*, pages 1–11. SIAM, 2001.

[11] L. Kovar and M. Gleicher. Flexible Automatic Motion Blending with Registration Curves. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '03, pages 214–224, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association.

[12] J. Oh, Y. Lee, Y. Kim, T. Jin, S. Lee, and S.-H. Lee. Hand Contact Between Remote Users Through Virtual Avatars. In *Proceedings of the 29th International Conference on Computer Animation and Social Agents*, CASA '16, pages 97–100, New York, NY, USA, 2016. ACM.

[13] J. Okun and S. Zwerman, editors. *The VES Handbook of Visual Effects*. Focal Press, Abingdon, UK, 2 edition, 2015.

[14] P. Senin. Dynamic time warping algorithm review. *CSDL Technical report*, 2008.

[15] F. Stakem and G. AlRegib. An Adaptive Approach to Exponential Smoothing for CVE State Prediction. In *Proceedings of the 2Nd International Conference on Immersive Telecommunications*, IMMERSCOM '09, pages 13:1–13:6, ICST, Brussels, Belgium, Belgium, 2009. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

[16] J. Starck, A. Cherbetji, B. Michoud, O. Grau, F. Einabadi, V. Helzle, P. Slusallek, R. Membarth, S. Rogmans, and V. Jacobs. Final prototype dreamspace toolsets. `http://www.dreamspaceproject.eu/Docs`, October 2016.

[17] A. Witkin and Z. Popovic. Motion Warping. In *Proceedings of the 22Nd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '95, pages 105–108, New York, NY, USA, 1995. ACM.

[18] S. Xia, C. Wang, J. Chai, and J. Hodgins. Realtime Style Transfer for Unlabeled Heterogeneous Human Motion. *ACM Trans. Graph.*, 34(4):119:1–119:10, July 2015.