MEMORY-RESTRICTED MULTISCALE DYNAMIC TIME WARPING

Thomas Prätzlich, Jonathan Driedger and Meinard Müller

International Audio Laboratories Erlangen

ABSTRACT

Dynamic Time Warping (DTW) is an established method for finding a global alignment between two feature sequences. However, having a computational complexity that is quadratic in the input length, memory consumption becomes a major issue when dealing with long feature sequences. Various strategies have been proposed to reduce the memory requirements of DTW. For example, online alignment approaches often have a constant memory consumption by applying forward path estimation strategies. However, this comes at the cost of robustness. Efficient offline DTW based on multiscale strategies constitutes another approach. While methods built on this principle are usually robust, their memory requirements are still dependent on the input length. By combining ideas from online alignment approaches and offline multiscale strategies, we introduce a novel alignment procedure that allows for specifying a constant upper bound on its memory requirements. This is an important aspect when working on devices with limited computational resources. Experiments show that when restricting the memory consumption of our proposed procedure to eight megabytes, it basically yields the same alignments as the standard DTW procedure.

Index Terms— Dynamic Time Warping, alignment, music synchronization, memory requirements

1. INTRODUCTION

The task of finding a global alignment between two feature sequences has received large research interest in the past, in the context of music information retrieval (MIR) and beyond. For example, the goal in music synchronization is to align musically corresponding positions in different representations of a piece of music. The different music representations may include sheet music (images), symbolic score data, or audio recordings. Many methods for *symbolic score-to-audio* alignment [1, 2, 3], *sheet music-to-audio* alignment [4], and *audio-to-audio* alignment [5, 6, 7] have been developed. Furthermore, alignment techniques are important in MIR tasks such as automatic accompaniment [8, 9] or score following [10, 11, 12]. Many different approaches have been used in alignment systems. For example, Hidden Markov Models or particle filters have been employed in score to audio alignment, where the current score position and tempo are modeled in a statistical sense [13, 14]. In this paper, we focus on methods based on Dynamic Time Warping (DTW) [5, 15, 16]. DTW is an effective technique to find an optimal alignment of two feature sequences. However, since the complexity of DTW is proportional to the product of the feature sequences' lengths [16], memory consumption becomes an issue when dealing with long feature sequences. But also aligning shorter sequences may become problematic with the standard DTW procedure when working for example on mobile devices, which usually only provide a limited amount of memory. To reduce the memory requirements of DTW, several strategies were proposed. On a global level, the Sakoe-Chiba band or the Itakura parallelogram impose a *constant global constraint region* on the set of possible alignments [15]. Using these constraints is problematic, as the optimal alignment may lie outside these regions. Another approach is to use *adaptive global constraints* such as in multiscale DTW (MsDTW) [6, 17]. Here, a projection of an alignment on a coarse feature resolution level (see Figure 1a) is used to constrain the computa-

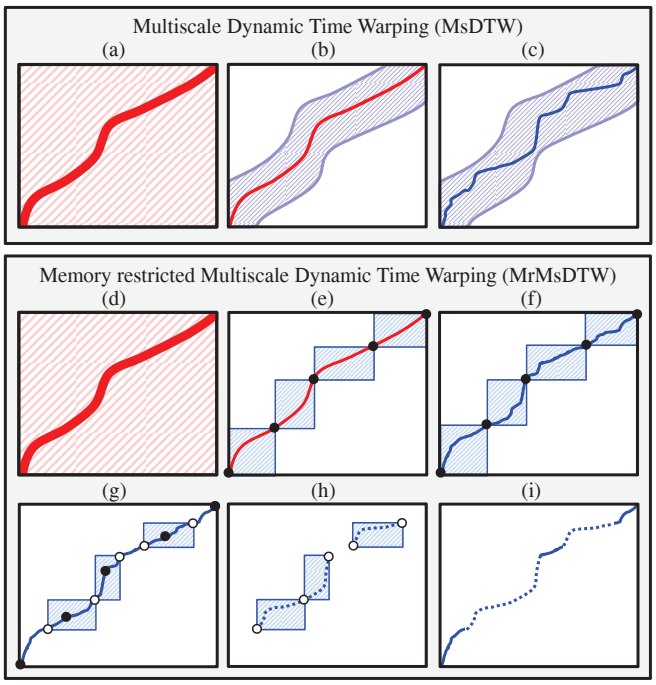

Fig. 1. Illustration of MsDTW (a)-(c) with global tubular constraint regions and the proposed MrMsDTW (d)-(e) with rectangular local constraint regions. **(a)** Alignment on coarse feature resolution. **(b)** Projection of (a) onto finer level with resulting tubular constraint region. **(c)** Refinement of (a) on fine resolution within tubular constraint region. Illustration of the MrMsDTW procedure. **(d)** Warping path P_C^* on coarse level (LEVEL C). **(e)** Projected warping path P_F on fine level (LEVEL F) and derived anchor sequence A. **(f)** First refinement of P_F by computing local paths within rectangular constraint regions defined by the anchor sequence A. **(g)** Derivation of the anchor sequence A' in a neighborhood of the anchor points in A. **(h)** Second refinement constrained by the rectangular constraint regions defined by A'. **(i)** Refined warping path P_F^*.

The International Audio Laboratories Erlangen are a joint institution of the Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU) and Fraunhofer IIS. This work has been supported by the BMBF project *Freischütz Digital* (Funding Code 01UG1239A to C) the German Research Foundation (DFG MU 2686/7-1).

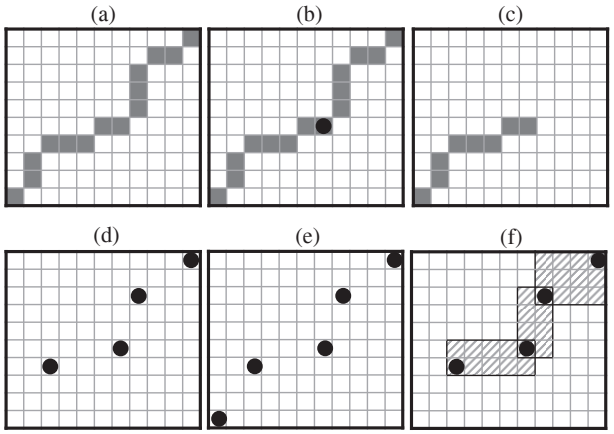

Fig. 2. Illustration of basic concepts. **(a)** Warping path P of length $|P| = 14$. **(b)** Center $h(P) = 8$ of warping path P. **(c)** Subpath $P[1:8]$. **(d)** Anchor sequence $A \subseteq P$. **(e)** Anchor sequence fulfilling boundary conditions $A' \sqsubseteq P$. **(f)** Maximum rectangular extent $\mathcal{R}(A) = 12$ of anchor sequence A.

tion of a refined alignment on a finer feature resolution level. In the refinement, the alignment is restricted to lie within a tubular constraint region that is constructed from the projected alignment (see Figure 1b+c). Note that there are cases where alignments computed by MsDTW deviate from the globally optimal alignment, see [6, 17].

In online scenarios, *local constraints* are imposed on the alignment. Here, the alignment is usually computed using greedy forward path estimation [5] or by using block-by-block processing where a block defines a local rectangular constraint region [18].

In this paper, we propose an MsDTW variant using the idea of building a global path from local alignments. Our memory-restricted MsDTW procedure, in the following called MrMsDTW, uses local rectangular constraint regions in the refinement step. The size, and therefore the required memory to store them, is bounded by a memory restriction parameter τ. The constraint regions are inferred from anchor points on the projected alignment (see Figure 1e). This yields a set of local alignments (see Figure 1f). In a second refinement step, these local alignments are corrected in a neighborhood around the initial set of anchor points. Again, this is done in rectangular constraint regions that are restricted in size by τ. This makes the memory requirement of our method basically constant in dependence of τ, whereas the memory requirement of MsDTW is linearly and DTW is quadratically growing with the lengths of the feature sequences (see Figure 3).

In the remainder of this paper, we first introduce our novel procedure in detail (Section 2) and then discuss the influence of the parameter τ on the robustness of MrMsDTW within a music synchronization application scenario (Section 3).

2. ALGORITHM

In this section, we introduce our memory-restricted alignment procedure that combines concepts from MsDTW [6, 17] with the idea of using rectangular local constraint regions as already proposed for online alignment procedures [18]. We first introduce some basic notions (Section 2.1), review the basics of DTW (Section 2.2) and MsDTW (Section 2.3), and then explain our proposed memory-restricted MsDTW approach (MrMsDTW) in detail (Section 2.4).

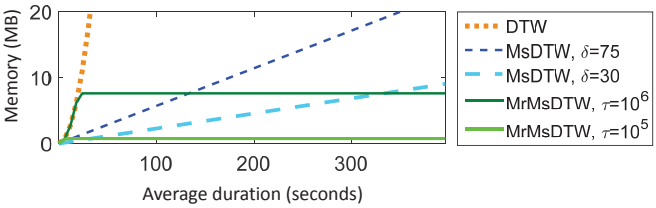

Fig. 3. Memory requirement for storing the accumulated cost matrix $\mathbf{D}$ for DTW, MsDTW, and MrMsDTW in dependency of the average duration of the recordings. The curves for MsDTW are lower bounds assuming a diagonal path for the constraint region computation.

2.1. Basic Notions

Let $\mathcal{F}$ be a suitable feature space. Furthermore, let $X := (x_1, \ldots, x_N)$ and $Y := (y_1, \ldots, y_M)$ be feature sequences with $x_n, y_m \in \mathcal{F}$, where $n \in [1:N] := \{1, \ldots, N\}$ and $m \in [1:M]$. A *cell* is a tuple $p := (n, m) \in [1:N] \times [1:M]$, which encodes the correspondence between two feature vectors x_n and y_m. We define the operators $\pi_1(p) := n$ and $\pi_2(p) := m$ to refer to the elements of a cell. A *path* is a sequence of cells $P = (p_1, \ldots, p_L)$, with $p_\ell \in [1:N] \times [1:M]$ for $\ell \in [1:L]$ satisfying the *step size condition* $(p_{\ell+1} - p_\ell) \in \Sigma := \{(1,0), (0,1), (1,1)\}$. We refer to the *length of a path* by $|P| := L$. A *warping path* is a path that additionally fulfills the *boundary conditions* $p_1 = (1,1)$ and $p_L = (N, M)$ (see Figure 2a). Note that a warping path constitutes a global alignment between two feature sequences. We define the operator $h(P) := \lfloor |P|/2 \rfloor + 1$ to return the center index of a sequence P (see Figure 2b). For retrieving the first, center, and last element of a path, we define the operators $B(P) := p_1$, $H(P) := p_{h(P)}$, $E(P) := p_L$, respectively. Furthermore, we define a *subpath* as $P[i:j] := (p_i, \ldots, p_j)$ with $1 \leq i < j \leq |P|$ (see Figure 2c). An *anchor point* is a specified cell $a = (n, m)$ and an *anchor sequence* is a sequence of anchor points $A = (a_1, \ldots, a_K)$ with $a_k \in [1:N] \times [1:M]$ that fulfills the condition of strict monotonicity $a_{k+1} - a_k \in \mathbb{N} \times \mathbb{N}$. We write $A \subseteq P$ if all cells of A are also cells of P (Figure 2d). Furthermore, we write $A \sqsubseteq P$ if $A \subseteq P$ and A additionally fulfills the boundary conditions $a_1 = p_1$ and $a_K = p_L$ (Figure 2e). Let $d_k := a_{k+1} - a_k + (1,1)$. We define the *maximum rectangular extent* of an anchor sequence A as $\mathcal{R}(A) := \max_{k \in [1:K-1]} \pi_1(d_k) \cdot \pi_2(d_k)$, which is the maximum rectangular area spanned between consecutive anchor points in A (Figure 2f). This concept will later be used to make the memory restriction of the algorithm explicit. Finally, to compare two features in $\mathcal{F}$, we use a local cost measure $c\colon \mathcal{F} \times \mathcal{F} \to [0, 1]$. By comparing each pair of elements in the sequences X and Y, we obtain the *cost matrix* $\mathbf{C}(n, m) := c(x_n, y_m)$. Given two anchor points a_1 and a_2, we define the *local cost matrix*

$$\mathbf{C}[a_1; a_2] := \mathbf{C}(n, m)_{\substack{\pi_1(a_1) \leq n \leq \pi_1(a_2), \\ \pi_2(a_1) \leq m \leq \pi_2(a_2)}} ,$$

which is a submatrix of $\mathbf{C}$.

2.2. Classical Dynamic Time Warping (DTW)

The goal of DTW is to compute an optimal warping path between the two feature sequences X and Y with respect to the cost measure c. The total alignment cost of a warping path P between two sequences X and Y is defined by $c_P(X, Y) := \sum_{\ell=1}^{L} \mathbf{C}(\pi_1(p_\ell), \pi_2(p_\ell))$. We say that a warping path P^* is optimal, if it has minimal alignment costs $c_{P^*} := \min\{c_P(X, Y) | P \text{ is warping path}\}$. To derive an optimal warping path, we first compute the *accumulated cost matrix* $\mathbf{D}$

Identifier	Composer / Piece	Recording 1		Recording 2		Memory		
		Performer 1	Dur. [s]	Performer 2	Dur. [s]	DTW [GB]	MsDTW [MB]	MrMsDTW [MB]
ID-01	Vivaldi / RV 269 (Spring), 1st movement	Abbado	203.2	Nishizaki	213.2	0.807	4.880	0.763
ID-02	Shostakovich / Jazz Suite No. 2, 6th movement (Waltz)	Chailly	223.7	Yablonsky	193.8	0.808	5.120	0.763
ID-03	Beethoven / Symphony 5, Op. 67, 1st movement	Karajan	444.0	Bernstein	519.2	4.294	11.884	0.763
ID-04	Weber / "Der Freischütz", No. 8	Furtwängler	611.5	Jochum	454.0	5.171	13.996	0.763
ID-05	Wagner, "Meistersinger", Prelude	Armstrong	595.1	Neumann	563.8	6.249	13.621	0.763
ID-06	Ravel / "Bolero"	Abbado	862.7	Ozawa	901.1	14.480	20.625	0.763
ID-07	Beethoven / Symphony 9, Op. 125, 4th movement	Karajan	932.6	Bernstein	928.5	16.129	21.346	0.763
ID-08	Schubert / Symphony 8, D759, 1st movement (Unfinished)	Solti	950.9	Sacchi	817.5	14.479	21.764	0.763
ID-09	Weber / "Der Freischütz", full opera	Carlos Kleiber	7763.3	Davis	8204.1	1186.379	187.784	0.763
ID-10	Wagner / "Das Rheingold", WWV 86A	Karajan	8752.1	Haitink	8930.3	1455.823	204.398	0.763

Table 1. Memory requirements for DTW, MsDTW with $\delta = 30$ (expansion size for tubular constraint region), and MrMsDTW with $\tau = 10^5$ (rectangular constraint region size) for a selection of pieces. ID-09 and ID-10 were not used in our robustness experiments, as the memory requirement for DTW is more than a terabyte. The alignments are therefore infeasible to compute for these pieces.

given by

$$\mathbf{D}(n,m) = \mathbf{C}(n,m) + \min_{(i,j)\in\Sigma} \mathbf{D}(n-i, m-j)$$

with $\mathbf{D}(n,1) = \sum_{k=1}^{n} \mathbf{C}(k,1)$ for $n \in [1:N]$ and $\mathbf{D}(1,m) = \sum_{k=1}^{m} \mathbf{C}(1,k)$ for $m \in [1:M]$. The optimal warping path can be obtained by backtracking the steps through $\mathbf{D}$, see [15] for details. Note that the time and space complexity of the algorithm to compute $\mathbf{D}$ and the optimal warping path is $O(NM)$.

2.3. Multiscale DTW (MsDTW)

In this section, we review the basic concepts of MsDTW originally introduced in [6, 17]. MsDTW aims to reduce the computational requirements of DTW by first computing an alignment on a coarse feature resolution level (LEVEL C). The coarse alignment is then projected onto a finer feature resolution level (LEVEL F) and refined using a tubular constraint region.

More precisely, let P_C^* be an optimal warping path computed on LEVEL C (see Figure 1a). The optimal warping path P_C^* is projected onto LEVEL F, resulting in a (potentially non-optimal) warping path P_F (see Figure 1b). To compute an optimal warping path P_F^* on LEVEL F, a tubular constraint region is constructed by adding $\delta \in \mathbb{N}$ cells to the left, top, right and bottom of P_F. Within this constraint region, P_F^* is computed via DTW (see Figure 1c). The procedure can be recursively applied by introducing further coarse resolution levels. Note that, depending on the choice of δ, the warping path P_F^* might not coincide with the globally optimal warping path computed with DTW. Using the MsDTW strategy reduces the memory requirements compared to classical DTW. However, the required memory is still linearly dependent on the length of the feature sequences [17]. For a detailed description of the procedure, we refer to [6, 17].

2.4. Memory-restricted MsDTW (MrMsDTW)

We now describe our proposed MrMsDTW procedure. Given a memory restriction parameter τ that sets an upper bound on the number of cells that can be used for the alignment computation, our main idea is to use rectangular local constraint regions that have a size of at most τ instead of a single global tubular constraint region. On each rectangular constraint region, a local alignment is computed on a local cost matrix. Each local alignment is computed by applying standard DTW on the local cost matrix. Furthermore, the computations of the local alignments are independent of each other. They can therefore be computed in a sequential way such that at most τ cells are used at any time.

As in Section 2.3, we use two resolution levels to describe the algorithm. The main difference to the classical MsDTW approach is the refinement on LEVEL F, where rectangular local constraint regions are derived from an alignment that is computed on LEVEL C. As before, let P_C^* be an optimal warping path computed on LEVEL C (Figure 1d). At this point, we assume that the computation of P_C^* did not require more than τ cells. Furthermore, let P_F be a suitable projection of P_C^* onto LEVEL F (Figure 1e). In the next step, we use P_F to construct a set of rectangular constraint regions to refine the alignment on LEVEL F. To this end, we derive an anchor sequence $A \sqsubseteq P_F$ (black dots in Figure 1e). Each consecutive pair of anchor points in A defines a rectangular constraint region. We further require A to fulfill the condition $\mathcal{R}(A) \leq \tau$. Such an anchor sequence can be obtained by initially setting $A := (B(P_F), E(P_F))$. If A does not fulfill $\mathcal{R}(A) \leq \tau$, the warping path P_F is recursively divided at its center into subpaths from which the first and last element are used as anchors. This is repeated until $\mathcal{R}(A) \leq \tau$ (see Figure 1e). In the next steps, we use A to compute a refined warping path P_F^* from P_F:

(Step 1) Refinement between anchor points. We compute a set of local paths $Q_1, \ldots, Q_{K-1}$ constrained by the anchor sequence $A = (a_1, \ldots, a_K)$. Each local path Q_k fulfills the boundary conditions $B(Q_k) = a_k$ and $E(Q_k) = a_{k+1}$ and is computed by using standard DTW on the local cost matrix $\mathbf{C}_k := \mathbf{C}[a_k; a_{k+1}]$ (see Figure 1f). If A contains only two anchor points ($K = 2$), we set $P_F^* := Q_1$. Otherwise, we proceed with the following steps.

(Step 2) Refinement in the neighborhoods of the anchor points. In the first refinement step, the warping path P_F was not refined at the anchor points in A. In this step, we therefore recompute the refinement in neighborhoods of these anchor points. To this end, we derive pairs of anchors A'_k with $k \in [1 : K-2]$, each fulfilling $\mathcal{R}(A'_k) \leq \tau$. We initialize each $A'_k := (H(Q_k), H(Q_{k+1}))$ and iteratively decrease the size of the respective rectangular constraint regions until they obey the memory restriction constraint (see white dots in Figure 1g). Now, we compute the set of local paths $Q'_1, \ldots, Q'_{K-2}$ where each local path Q'_k is constrained by the anchor pair A'_k (see Figure 1g+h).

(Step 3) Concatenation. The local paths from the previous steps are combined to a global warping path P_F^* by concatenation (see Figure 1i). When computing the local alignments sequentially, the maximum memory requirement of the algorithm in the two refinement steps can be directly inferred from the used anchor sequences A and A' by computing $\mathcal{R}(A)$ and $\max_k \mathcal{R}(A'_k)$ respectively.

As a final remark, note that the procedure described above can be applied in a recursive fashion by introducing further levels of decreasing feature resolution, similar as in the classic MsDTW procedure. In practice, we choose the coarsest feature resolution level such that we need at most τ cells for the computation of the full DTW procedure on this level. Note that this leads to a constant memory requirement for MrMsDTW.

3. EXPERIMENTS

In this section, we first describe our experimental setting. Then, we investigate the robustness of our proposed MrMsDTW procedure in dependency of the memory restriction parameter τ in the context of an audio-to-audio music synchronization scenario.

In the experiment, we use four fixed feature resolution levels (50 Hz, 10 Hz, 2 Hz, 1 Hz), referred to as Level 1–4 where Level 1 corresponds to the finest feature resolution and Level 4 to the coarsest. In the case that the feature sequences are so long that the initial alignment would violate our memory constraint, we add a fifth level choosing a feature resolution that still fulfills the memory requirement. We use the same feature set as described in [19].

Our dataset contains 53 pairs of classical music recordings, each pair consisting of two different performances of the same piece of music. Table 1 shows a representative selection of the pieces used in our experiments. For each entry, a lower bound for the memory requirement of DTW, MsDTW, and MrMsDTW is given. The lower bound is derived from the number of memory cells required to store the accumulated cost matrices $\mathbf{D}$ that are used in DTW, MsDTW, and MrMsDTW. The memory usage in megabytes (MB) is obtained by multiplying the required number of memory cells to store $\mathbf{D}$ by $\frac{8}{1024^2}$ (assuming a 64 bit machine using double precision floating point numbers). Note that these values may need to by multiplied by a factor of 2 or 3, depending on the implementation (for additionally storing the cost matrices and a matrix that saves the step sizes and indices leading to a cost-minimizing path in $\mathbf{D}$). However, this factor is the same for all three DTW variants. For MsDTW, we computed the lower bound by assuming a diagonal path between the two feature sequences, leading to a requirement of $2\delta \cdot N$ memory cells, where N denotes the length of the longer feature sequence. In our dataset, the shortest piece is roughly one minute long, and the longest about 15 minutes. Note that the memory requirement of DTW for the piece `ID-07` in Table 1 is already higher than 16 gigabytes. As the memory usage of DTW grows quadratically with the length of the input feature sequences, it is not feasible to use it for longer pieces on a normal desktop computer (assuming a maximum of 16 gigabytes of available memory). For example, full DTW would require more than one terabyte of memory on the opera recordings `ID-09` and `ID-10` in Table 1. However, our proposed MrMsDTW approach has basically constant memory requirements, and is therefore capable to compute alignments for longer pieces. The memory requirements for DTW (quadratic), MsDTW (linear), and MrMsDTW (constant) are illustrated in Figure 3.

In the following, we investigate the connection between the memory restriction parameter τ and the robustness of our proposed MrMsDTW procedure, see Table 2. To this end, we compare the alignments computed by MrMsDTW with the optimal alignments computed by the full DTW procedure as described in Section 2.2. When aligning two music recordings, the warping path P_{MrMsDTW} may deviate from the optimal global warping path P^*_{DTW}. The smaller the constraint regions, the more likely it is that such deviations occur. Since small deviations from the optimal alignment might be acceptable, we introduce a tolerance parameter ε for the evaluation. We say that a cell in P_{MrMsDTW} was correct in case it lies within a region of ε-cells to the left, top, right, and bottom around a cell from P^*_{DTW}. This means that for ε=0, a cell in the P_{MrMsDTW} is only considered to be correct if it coincides perfectly with a corresponding cell in the optimal warping path P^*_{DTW}. In contrast, for ε=16, a deviation of up to 0.32 seconds from the optimal path is allowed (assuming a feature resolution of 50 Hz).

τ \ ε	0	1	2	4	8	16	τ [MB]
10^7	99.90	99.90	99.90	99.91	99.93	99.94	76.29
10^6	99.81	99.83	99.85	99.87	99.88	99.92	7.629
10^5	95.82	96.57	96.90	97.49	98.31	99.13	0.763
10^4	82.60	87.94	90.18	93.36	96.37	98.44	0.076
10^3	37.58	59.53	69.81	82.00	91.98	97.29	0.008

Table 2. Overall percentage of warping path cells in P_{MrMsDTW} that have a deviation $\leq \varepsilon$ cells from the corresponding cells of the optimal global warping path P_{DTW}. The value has been computed for each warping path and was then averaged over the whole dataset (to avoid that longer pieces have a stronger influence). The error tolerance ε defines a tolerance region around the P_{DTW} of $\pm\varepsilon$ frames in horizontal and vertical direction. One frame corresponds to 20 ms.

Table 2 shows the overall percentage of correct warping path cells in dependency of the memory restriction τ and the tolerance ε for our dataset. Furthermore, the memory requirement for a given τ is stated. For τ=10^7, having a memory requirement of ≈76 MB, nearly all alignment cells coincide with the full DTW approach. As a comparison, the full DTW needs already more memory for pieces that are longer than 63 seconds. For τ=10^6, reducing the memory requirement to ≈7.6 MB, even with $\varepsilon = 0$, more than 99% of the alignment cells are still correct. The memory requirement of MsDTW with $\delta = 30$ is only smaller for pieces shorter than 334 seconds (see Figure 3). With τ=10^5, requiring less than 1 MB of memory, the deviations from the optimal path start to increase. In this case, MsDTW with $\delta = 30$ is only more memory efficient for pieces shorter than 34 seconds (see Figure 3). For $\tau \leq 10^4$ (when providing less than 100 kilobytes of memory), the procedure becomes unstable, leading to stronger deviations from the optimal warping path. When reducing the memory requirement down to 8 kilobytes ($\tau = 10^3$), only 37.58% of the warping cells correspond with the full DTW results with a tolerance $\varepsilon = 0$. However, within a tolerance of $\varepsilon = 16$ (0.32 seconds), more than 97% of the cells are still considered to be correct. Depending on the application, this accuracy might still be acceptable.

4. CONCLUSIONS

In this contribution, we proposed MrMsDTW, a new MsDTW variant using rectangular shaped local constraint regions. It combines the block-by-block processing from online alignment techniques with a multiscale strategy. The introduced procedure has a constant memory requirement, being explicitly controlled by a memory restriction parameter. In addition, MrMsDTW basically yields the same alignments as a full DTW approach, even when restricting it to require only eight megabytes of memory. As a final remark, opposed to classical MsDTW, our presented procedure has the potential of being implemented in a parallel fashion as the individual local paths can be computed independently of each other. This would trade the explicit memory restriction for a better runtime performance.

5. REFERENCES

[1] Roger B. Dannenberg and Ning Hu, "Polyphonic audio matching for score following and intelligent audio editors," in *Proceedings of the International Computer Music Conference (ICMC)*, San Francisco, USA, 2003, pp. 27–34.

[2] Cyril Joder, Slim Essid, and Gaël Richard, "A conditional random field framework for robust and scalable audio-to-score matching," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 8, pp. 2385–2397, 2011.

[3] Christopher Raphael, "A hybrid graphical model for aligning polyphonic audio with musical scores," in *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, Barcelona, Spain, 2004, pp. 387–394.

[4] Frank Kurth, Meinard Müller, Christian Fremerey, Yoon ha Chang, and Michael Clausen, "Automated synchronization of scanned sheet music with audio recordings," in *Proceedings of the 8th International Conference on Music Information Retrieval (ISMIR)*, Vienna, Austria, 2007, pp. 261–266.

[5] Simon Dixon and Gerhard Widmer, "MATCH: A music alignment tool chest," in *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, London, GB, 2005.

[6] Meinard Müller, Henning Mattes, and Frank Kurth, "An efficient multiscale approach to audio synchronization," in *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, Victoria, Canada, 2006, pp. 192–197.

[7] Robert J. Turetsky and Daniel P. W. Ellis, "Ground-truth transcriptions of real music from force-aligned MIDI syntheses," in *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, Baltimore, USA, 2003, pp. 135–141.

[8] Roger B. Dannenberg and Christopher Raphael, "Music score alignment and computer accompaniment," *Communications of the ACM, Special Issue: Music information retrieval*, vol. 49, no. 8, pp. 38–43, 2006.

[9] Christopher Raphael, "A probabilistic expert system for automatic musical accompaniment," *Journal of Computational and Graphical Statistics*, vol. 10, no. 3, pp. 487–512, 2001.

[10] Arshia Cont, "A coupled duration-focused architecture for real-time music-to-score alignment," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 6, pp. 974–987, 2010.

[11] Nicola Orio, Serge Lemouton, and Diemo Schwarz, "Score following: State of the art and new developments," in *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)*, Montreal, Canada, 2003, pp. 36–41.

[12] Andreas Arzt, Sebastian Böck, and Gerhard Widmer, "Fast identification of piece and score position via symbolic fingerprinting," in *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, 2012, pp. 433–438.

[13] Zhiyao Duan and Bryan Pardo, "A state space model for online polyphonic audio-score alignment," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Prague, Czech Republic, 2011, pp. 197–200.

[14] Nicola Montecchio and Arshia Cont, "A unified approach to real time audio-to-score and audio-to-audio alignment using sequential Montecarlo inference techniques," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Prague, Czech Republic, 2011, pp. 193–196.

[15] Lawrence Rabiner and Bing-Hwang Juang, *Fundamentals of Speech Recognition*, Prentice Hall Signal Processing Series, 1993.

[16] Meinard Müller, *Fundamentals of Music Processing*, Springer Verlag, 2015.

[17] S. Salvador and P. Chan, "FastDTW: Toward accurate dynamic time warping in linear time and space," in *Proceedings of the KDD Workshop on Mining Temporal and Sequential Data*, 2004.

[18] Robert Macrae and Simon Dixon, "Accurate real-time windowed time warping," in *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, Utrecht, Netherlands, 2010, pp. 423–428.

[19] Sebastian Ewert, Meinard Müller, and Peter Grosche, "High resolution audio synchronization using chroma onset features," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Taipei, Taiwan, 2009, pp. 1869–1872.