Abstract—The advent of commodity 3D sensor technology enabled, amongst other things, the efficient and effective assessment of human movements. Statistical and machine learning approaches map recorded movement instances to expert scores to train models for an automated assessment of new movements. However, there are many variations in selecting the approaches and setting the parameters for achieving good performance, i.e., high scoring accuracy and low response time. The present paper researches the design space and the impact of sequence alignment on accuracy and response time. More specifically, we introduce variants of Dynamic Time Warping (DTW) for aligning the phases of slow and fast movement instances and assess their effect on the scoring accuracy and response time. Results show that an automated stripping of leading and trailing frames not belonging to the movement (using one DTW variant) followed by an alignment of selected frames in the movements (based on another DTW variant) outperforms the original DTW and other suggested variants thereof. Since these results are independent of the selected learning approach and do not rely on movement specifics, the results can help improving the performance of automated human movement assessment, in general.

Keywords: Dynamic Time Warping variants, human movement assessment

1. Introduction

Healthcare is becoming more and more value centered optimizing patient value (best possible health achieved) and system value (effective treatments at efficient costs). Until recently, measuring value, e.g., clinical outcomes, was not a capability of hospital or health IT [1]. With inexpensive sensor technologies and data analytics becoming increasingly available, it is nowadays possible to collect data on the clinical activities of healthcare, the health status of a patient and the change of this status after treatment. We contribute with an automated objective method for the assessment of the physical health of the human musculoskeletal system to help diagnose, predict or prevent related pain, injuries and long-lasting diseases. We use commodity 3D camera technology, such as the Microsoft Kinect. Unlike similar tools for physical therapists not aiming to support the caring or nursing process, our approach localizes potential issues and quantifies their severity.

Any effective and efficient method for the assessment of human movements should be independent of the actual movement. Using statistics and machine learning (ML) based approaches, it is quite efficient to add new movements and to assess them accurately [2]. Experts score some representative instances of a new movement to train a model that is then used for the assessment of unknown instances of this movement.

As in many data driven applications, selecting the right approach is still a tedious, partially manual process characterized by prototyping, trial and error, because there are quite a few variants to select from spanning a huge search space. At the same time, variant selection has a large impact on the performance of the final application. Variants include: feature selection, the selection of preprocessing steps, movement sequence alignment (cutting and matching), statistics and ML model selection, and the selection of training and testing data.

The present paper focuses on selecting variants of sequence cutting and matching. Sequence cutting is the problem of selecting the frames of a recording belonging to the actual movement that should be assessed while stripping leading and trailing frames recording, e.g., entering the scene, preparations, or leaving the scene. Sequence matching is the problem of splitting a movement instance into phases that match with the same phases of other movement instances regardless of the speed of the movement instances. Matching equal phases of different movement instances is an important subproblem of human movement recognition and assessment, necessary to build a statistical model of a movement. Solutions based on Dynamic Time Warping (DTW) [3] and Hidden Markov Models (HMM) [4] are arguably predominant approaches for general sequences and also for movement sequences. The present paper focuses on DTW based approaches.

The paper contributes with (1) introducing novel DTW variants, (2) putting them together to variants of cutting and matching, and (3) experimentally assessing their accuracy and response time. It is organized as follows: Section 2 discusses related work in human movement assessment. Section 3 establishes the foundations of the subsequent
contributions. Section 4 motivates and introduces variants of DTW and of cutting and matching. Section 5 experimentally evaluates the variants with respect to assessment accuracy and response time. Finally, Section 6 concludes the paper and points out directions of ongoing and future work.

2. Related Work

Features necessary for human movement assessment extracted with commodity 3D sensor technology such as the Kinect seem to agree with the same features extracted with high-end motion capturing systems [5]–[7] or with other sensor technology [8], [9]. Only few studies found differences in, e.g., foot placement locations and step lengths [10].

This sets the foundations for quite a few studies that research the recognition of human movements using commodity 3D sensor technology, e.g., [11]–[16]. While these approaches are similar to ours in their feature extraction and preprocessing steps, their goal is the classification of different movements not their quality assessment.

The same technology has also been used in movement quality assessment, e.g., for identifying anterior cruciate ligament (ACL) injury risks [17], human abduction-adduction motion analysis [18], for ergonomic assessment in real workplace conditions [19], for the measurement of lower extremity jump landing and squatting kinematics, for sit-to-stand analysis [20], and for static and dynamic balance assessment [21]. Similar studies include [22]–[27]. However, these studies aim at qualitative assessments—checking, e.g., whether movements are correctly or incorrectly, well or poorly performed—or at identifying different abnormal movement patterns rather than at quantitatively scoring the quality of a movement and localizing impairments. Some of the studies [24], [26] use preprocessing transformations such as dimensionality reduction using manifold learning [28], which enables a fast online assessment but even prohibits impairment localization.

Dressler et al. [29], [2] validate an automated, quantitative assessment of human movements based on the Kinect camera against expert scores. Their automated assessment even localizes the quality issue. It provides the baseline of our performance comparisons.

Statistics and ML methods used in the studies include linear regression [29], [2], support vector machines [30], [31], [27], Adaboost [27], [30], random forest [32], multilayer neural network [27], and distance functions [27], [31], [33] for estimating exercise quality classes and scores. A recent study [34] used deep neural networks such as CNN, RNN, and hierarchical neural networks for assessing the quality of 10 different exercises with high accuracy, including, deep squat, hurdle step, and sit to stand. Our study uses linear regression, random forest, support vector regression, and neuronal networks to assess the impact of the sequence cutting and matching variants on accuracy.

Solutions of the problem of matching equal phases of different movement instances are based on Dynamic Time Warping (DTW) [3] and Hidden Markov Models (HMM) [4]. DTW approaches are used, e.g., in [12], [22], [23], [15], [29], [2], HMM approaches in [13], [26]. None of the two approaches—neither DTW nor HMM—can ignore irrelevant leading, intermediate or trailing subsequences. Cutting out these subsequences, if done at all, is handled in (manual or hard-coded) preprocessing steps. DTW based techniques have been improved in robustness [35] and performance [23], [36]. Our variants of DTW are able to cut out relevant subsequences either as a step before sequence matching or as part of finding an optimal matching.

3. Basic Definitions

The Kinect identifies 24 “joints”. Their $x$, $y$, $z$ positions are given in a 3D coordinate system with the camera in its origin and a horizontal $x,z$ plane with $z$ the depth coordinate. Because of low reliability of the other joints, we only use the following: head, neck, spine shoulder, spine mid, spine base, left/right shoulder, left/right elbow, left/right wrist, left/right hip, left/right knee, and left/right ankle.

A sequence is a list of frames from a recording of a human movement, e.g., an overhead deep squat. It is called a master sequence if the movement is assessed and scored by a human expert and used for building a statistical model of the movement. A sequence to be scored automatically is a user sequence. A frame is a record of features. It is a part of a sequence and describes the body posture at a specific point in time during the movement. A feature is a part of a frame and describes an aspect of the body posture at specific point in time.

A feature is called direct if it is directly measured by the 3D camera or indirect if it is computed from the direct or other indirect features. The direct features include the $x$, $y$, $z$ coordinates of skeleton joints. Indirect features include the angles between limbs and angles between limbs and the axes of the 3D coordinate system. An aggregated sequence is a list of aggregated frames aggregating one or more sequences into one. An aggregated frame aggregates two or more frames into one. It is a vector of sample distributions of the feature values of each feature. For scoring, we compare a user sequence with an aggregated master sequence.

All variants of our movement assessment algorithm perform five steps, described in the following paragraphs and precisely defined in [29].

Step 1) Building an aggregated master sequence builds an initial aggregated master sequence and aggregates it with the other master sequences. (1.1) Select the best master sequence $S$. It should have a constant movement speed without any delays and stops. (1.2) Cut off leading and trailing frames of $S$ containing postures that do not belong to the movement. (1.3) Prepare $S$, i.e., apply Step 2. (1.4) Aggregate subsequent frames of $S$. We use 15 groups each containing the same number of frames ($\pm 1$). Each group
contributes to a separated aggregated frame. (1.5) Normalize $S$, i.e., apply all but the first sub-steps of Step 4). (1.6) Compute a numerical sample distribution of the feature values, separately for each group and for each feature.

For any of the remaining master sequences $S'$, (1.7) Prepare $S'$, i.e., perform Step 2). (1.8) Match $S'$ with current aggregated master sequence, i.e., perform Step 3. This matching maps each frame of $S'$ to an aggregated frame of the aggregated master sequence. (1.9) Normalize the master sequence, i.e., perform Step 4). (1.10) Aggregate subsequent frames of $S'$, i.e., for each frame and each feature of $S'$, add the feature value to the sample distribution of the respective feature of the mapped aggregated frame.

Step 2) Preparing the sequences: (2.1) Align the floor clip plane: for each frame, the joint positions are rotated such that the floor clip plane is parallel to the x,z plane. (2.2) Smoothening: for all direct features, a sliding average of feature values is computed. (2.3) Interpolate: if a joint was not visible for $< k$ (resp. $\geq k$) consecutive frames its position is interpolated (resp. it is considered not tracked and an error is reported).

Step 3) Cutting and Matching constitute the main variation points of the assessed solution variants, cf. Section 4. (3.1) Cutting removes leading and trailing frames of postures not belonging to the movement. (3.2) Matching aligns a sequence with the aggregated master sequence.

Step 4) Normalizing: (4.1) Compute a Scaling transformation (procrustes analysis) that moves each joint of the first frame of a sequence to the corresponding mean joint position of the first aggregated frame of the aggregated frame sequence. Then apply this scaling to all other frames of the sequence. (4.2) Compute a Hip rotation transformation for the first frame of a sequence that lets skeletons “look” towards the camera. Then apply this rotation to all other frames of the sequence. (4.3) Compute the indirect features from the direct features for each frame.

Step 5) Scoring: (5.1) Calculate frame weights for the aggregated master sequence frames based on the extremeness of the poses in these frames, i.e., the deviation of the features from a normal null pose. This is done offline, once for the aggregated master sequence. (5.2) Calculate the feature scores of user sequences feature values $v$ as the $z$-score of $v$ in the corresponding sample distributions calculated in sub-step 1.10 and multiply them with the respective weight calculated in sub-step 5.1. (5.3) Aggregate the feature scores to an overall score depending on the statistical or ML approach used. Various regression based machine learning algorithms are applied to estimate the overall score. A linear regression method is the baseline. Alternatives are random forest regression (RF), support vector regression (SVR), and feedforward deep neural network (NN). The machine learning pipeline, cf. Figure 1, uses recursive feature elimination (RFE) for RF and SVR to select the subset of features that achieves the best estimation results [37]. For NN, RFE was not applied, because the importance of features is automatically regarded. Parameter tuning was applied in all models. The estimated scores are passed to the model evaluation step where they are compared with human experts scores.

4. Cutting and Matching Sequences

Let $N$ be the number of aggregated master sequence frames and $M$ be the number of user sequence frames. A matching $\mathcal{M}$ is a relation $\subseteq [1 \ldots N] \times [1 \ldots M]$. For a matching to be correct, all aggregated master sequence frames are matched (i) $\forall n \in [1 \ldots N] : (n,\_ ) \in \mathcal{M}$ and the matching must obey the order of frames in the sequences (ii) $(n,m) \in \mathcal{M} \Rightarrow \neg \exists (n',m') \in \mathcal{M} : n' < n \land m' > m$. For a matching to additionally be complete, all user sequence frames are matched (iii) $\forall m \in [1 \ldots M] : (\_, m) \in \mathcal{M}$.

The costs of a matching are defined as the deviations of the aggregated master sequence frames and the matched user sequence frames. More specifically, the deviation of a value $v$ of feature $f$ of a user sequence frame from a distribution $D_{f,n}$ of the corresponding feature values in an aggregated master sequence frame $n$ with mean $\mu_{f,n}$ and standard deviation $\sigma_{f,n}$ is the $z$-score of $v$ in $D_{f,n}$, i.e., $d_{f,n}(v) = \frac{v - \mu_{f,n}}{\sigma_{f,n}}$. The deviation of a frame to an aggregated master sequence frame is the average of the deviations of all contained features. The deviation of an aggregated master sequence and a user sequence is the sum (resp. average) deviation of the matched aggregated and user frames.

The goal of cutting and matching is to cut off leading and trailing frames not belonging to the movement and to find a correct, minimum cost matching in order to compare the phases of a user movement with the right aggregated frame. Known algorithms for sequence alignment such as DTW- or HMM-based algorithms are not capable of cutting leading and trailing frames or perform this in a suboptimal way. We therefore developed own DTW variants that aim at finding minimum cost cuts and matches automatically. Section 4.1 introduces these variants of DTW while Section 4.2 puts the basic algorithms together to create variants of cutting and matching.

4.1 Variants of DTW

Let $\text{dev}[n,m]$ be the deviations of the $m$-th frame in a user sequence matched to the $n$-th aggregated frame in a
master sequence, $1 \leq n \leq N, 1 \leq m \leq M$ with $M$ and $N$ the length of the user and the aggregated master sequences, resp.

**Algorithm 1:** Min cost sequence matching (DTW)

Algorithm 1 is a recursive implementation of DTW computing the minimum cost of a matching. The recursive function $Seq\_Match$ computes the minimum costs for an aggregated master sequence starting at frame $n$ and a user sequence starting at frame $m$. It is called initially (line 1) with the first frame numbers of both sequences. If applicable, the algorithm then tries to match the aggregated frame $n$ with the user frame $m$ (line 8), or the aggregated frame $n - 1$ with the user frame $m$ (line 11), or the aggregated frame $n$ with the user frame $m - 1$ (line 14), and selects whatever leads to minimum costs. For this and the following algorithms, the actual matching can trivially be derived as a side effect of this selection. The recursion base case (lines 3–5) makes sure that the algorithm terminates and is never called with $n > N + 1$ nor $m > M + 1$. The algorithm computes a correct and even complete matching. Especially, completeness implies that also leading and trailing frames of a recorded user movement get matched. The matching minimizes the sum of deviations of the matched frames. Therefore, the algorithm is biased towards matchings with few matched frames.

Algorithm 2 is a variant of DTW computing a correct, minimum cost but incomplete matching, i.e., not all frames of the user sequence are guaranteed to be matched. Additionally, the same user sequence frame $m$ will never be matched with two consecutive aggregated frames $n$ and $n + 1$. It is also called initially (line 1) with the first frame numbers of both sequences and then, if applicable, tries to match the aggregated frame $n$ with the user frame $m$ or to ignore the user frame $m$ (line 8), whatever leads to minimum costs. The recursion base cases (lines 3–6) make sure that the algorithm terminates and is never called with $n > N + 1$ nor $m > M + 1$. Moreover, if not all $N$ aggregated master sequence frames are matched but there are no user sequence frames left for matching, i.e., $m == M + 1$, the algorithm returns infinite costs (line 6) and, hence, avoids matchings violating the correctness condition (i).

Algorithm 3 is another DTW variant computing a correct, minimum cost but incomplete matching. However, it only ignores leading and trailing frames of user sequences. The recursive function $Seq\_Cut\_Match$ computes the minimum costs for an aggregated master sequence starting at frame $n$, a user sequence starting at frame $m$, and a matching of cardinality $k$. This means that the matching contains $k$ pairs of aggregated master and user frames. If applicable, the algorithm tries to match the aggregated frame $n$ with the user frame $m$ (line 12), or the aggregated frame $n - 1$ with the user frame $m$ (line 15), or the aggregated frame $n$ with the user frame $m - 1$ (line 18), or to skip a leading user frame $m$ (line 21), and selects whatever leads to minimum costs. The recursion base case (lines 7–9) makes sure that the recursion stops for correct matches. For incorrect matches, if all $N$ aggregated master sequence frames are matched but there are too many or too few user sequence frames matched, i.e., $k \neq 0$, the algorithm eventually returns infinite costs (line 10) as none of the if-statements becomes applicable. Therefore, matchings obey the correctness condition (i) and return a matching of exactly size $k$. The algorithm calls the recursive function $Seq\_Cut\_Match$ for $k \in [1 \ldots M]$ (lines 2–4) and returns (the matching with) the lowest average costs (line 5). Therefore, it is not biased to matchings with low cardinality, as the previous DTW variants that return (the matchings with) the lowest sum of costs.

All recursive algorithms of this section are exponential.
Data: $N, M : \mathbb{N}; \text{dev} : [1 \ldots N, 1 \ldots M]$ of $\mathbb{R}$
Result: $\text{costs} : \mathbb{R}$
1. $\text{costs} = \text{MAX\_VALUE}$
2. for $k \in 1 \ldots M$ do
3. \hspace{1em} $\text{costs} = \min(\text{costs}, \text{Seq\_Cut\_Match}(1, 1, k)/k)$
4. end
5. return $\text{costs}$

function $\text{Seq\_Cut\_Match}(n, m, k)$:
1. if $n == N + 1 \times k == 0$ then
2. \hspace{1em} return 0
3. end
4. $\text{costs} = \text{MAX\_VALUE}$
5. if $n <= N \times m <= M$ then
6. \hspace{1em} $\text{costs} = \min(\text{costs}, \text{dev}[n, m] +
7. \hspace{2em} \text{Seq\_Cut\_Match}(n + 1, m + 1, k - 1))$
8. end
9. if $n > 1 \times m <= M$ then
10. \hspace{1em} $\text{costs} = \min(\text{costs}, \text{dev}[n - 1, m] +
11. \hspace{2em} \text{Seq\_Cut\_Match}(n + 1, m + 1, k - 1))$
12. end
13. if $n <= N \times m > 1$ then
14. \hspace{1em} $\text{costs} = \min(\text{costs}, \text{dev}[n, m - 1] +
15. \hspace{2em} \text{Seq\_Cut\_Match}(n + 1, m, k - 1))$
16. end
17. if $n == 1 \times m <= M$ then
18. \hspace{1em} $\text{costs} = \min(\text{costs}, \text{Seq\_Cut\_Match}(n, m + 1, k))$
19. end
20. return $\text{costs}$

Algorithm 3: Min cost sequence cutting and matching

However, dynamic programming avoids multiple recursive invocations of the functions $\text{Seq\_Match}$, $\text{Frame\_Match}$, and $\text{Seq\_Cut\_Match}$, resp., with the same parameters. While still trying all correct matches, it keeps book of the minimum costs of matchings resulting from the recursive calls. This leads to $O(N \times M)$ implementations of the Algorithms 1 and 2, and to an $O(N \times M^2)$ implementation of Algorithm 3 that are efficient in practice, cf. Section 5.

4.2 Variants of Cutting and Matching

Now we are ready to use the different DTW variants introduced in Section 4.1 for solving the actual problem of cutting and matching. This solution also comes in variants as described below.

Variant 1) Cutting and Matching in One Step: The first variant cuts off leading and trailing frames and computes a matching in just one step. We simply apply Algorithm 3 that is tailor-made for these tasks. It is, however, the variant with the highest time complexity $O(N \times M^2)$.

Variant 2) Cutting then Sequence Matching: The second variant first cuts a user sequence and then computes a complete matching of the aggregated master and the truncated user sequence.

The cutting step uses Algorithm 2. It requires a special master sequence that only consists of start and end pose frames of one or more master sequences. If start and end poses of the movement are too similar to each other or to any of the poses in the sequence, the special master sequence is extended with some intermediate frames. For instance, in a deep squat exercise, the start and end poses are ideally identical: the person stands strait with the arms up. Here the special master sequence needs to contain a frame of an intermediate pose, e.g., the deep-down pose, that is clearly distinguished from start and end. Then the order preservation correctness criteria (ii) avoids a matching of two (almost) consecutive frames at the start (or at the end) of the user sequence to the start (end) frame of the special master sequence. The start, end, and intermediate frames of the special master sequence are selected manually only once. Applying Algorithm 2 to this special master sequence and an arbitrary user sequence leads to a correct matching that yields the minimum costs such that the start (end) frame of the special master sequence is mapped to the start (end) frame of the user sequence. We can ignore the matching of intermediate frames and cut the user sequence before the start (and after the end) frame.

For matching, Algorithm 1 computes a correct, complete, minimum cost match of the actual aggregated master sequence and the truncated user sequence. The time complexity of each of the two steps and, hence, the whole variant is $O(N \times M)$.

Variant 3) Cutting then Frame Matching: The third variant again separates cutting and matching. However, it uses Algorithm 2 for both cutting and matching. Cutting is done exactly as in the variant 2) discussed before.

Matching computes then an incomplete match of the actual aggregated master and the truncated user sequence. As a result, each aggregated master sequence frame is matched to exactly one user sequence frame that has the minimum average feature value deviation from the means of the feature distributions of the matched aggregated master frame.

Also for this third variant, the time complexity of each of the two steps and the whole variant is $O(N \times M)$.

5. Assessment of the Variants

Method: We use an overhead deep squat as the evaluation movement, an exercise standardized by the National Academy of Sports Medicine (NASM)\(^2\) that comes with movement execution and scoring specifications. We evaluate the accuracy of the different solution variants as follows. The baseline approach is the digitalization of the NASM overhead deep squat assessment and scoring, i.e., it implements the NASM scoring specification. Manual human-expert-based NASM overhead deep squat assessment and its digitalization show a high agreement [29] so that the

Fig. 2: Accuracy of variants (Pearson correlation $r$ with the ground truth) using different statistics and ML approaches
digitalization serves as our ground truth. We assess the agreement of the scores for the three solution variants introduced in Section 4.2 with the scores of baseline. A high agreement is considered a high accuracy. We therefore calculate the Pearson coefficient $r$ of the correlation of scores coming from the different variants with baseline.

**Data:** The assessment is based on a sample of 2094 user sequences recorded in 2018, the bulk during so-called company health days in Germany where AIMO\(^3\) recorded deep squats of health interested employees. The sample data contains scans of female and male persons, 25–65 years old, working in office jobs. No further effort was made to get a representative and fair sample of the full population.

**Implementation:** For the variants 1) to 3) including preprocessing, cutting and matching, and scoring with linear regression, the evaluation uses a Matlab implementation running on a MacBook Pro with a 2.3 GHz Intel Core i7 processor. The machine learning based scoring uses the Sklearn (RF, SVR) and the Keras libraries (NN) for Python. The baseline approach uses AIMO’s production code. For response time evaluation, only preprocessing, cutting, matching, and feature scoring are assessed as the variants do not impact the response time of feature score aggregation. Note that we are interested in the relative response time between the variants not in minimizing the absolute response time. For linear regression and ML, the accuracy results depend on the splitting of the sample in training and test data sets. We therefore repeat the experiments and report the average $r$ of 100 runs with fair random splittings in training and test data sets each containing 50\% of the sample data for a run.

**Results:** Figure 2 shows the Pearson correlation between the ground truth and the overall score using the different cutting and matching variants. The Random Forrest (RF) approach outperforms the others. It is marginally better for variant 2) that performs cutting with Algorithm 2 then matching with the original DTW Algorithm 1 ($r = 0.831$) compared to variant 3) cutting and matching with Algorithm 2 ($r = 0.828$). The first variant 1) based on Algorithm 3 still strongly correlates with the ground truth but is clearly behind the two other variants in accuracy ($r = 0.801$).

Figure 3 shows the response times of the three variants in $msec$. We build a statistical model of master sequences and then perform feature scoring of user sequences using that model to compute an overall score. Model building and feature scoring need preprocessing displayed with a separate bar in the diagram. Variant 1) requires $\approx 0.6sec$ per sequence. It is the fastest in preprocessing as it does not involve any cutting. However, the $O(N \times M^2)$ complexity makes building a statistical model and using it in feature scoring perform worst: it responds after $\approx 30sec$ per user sequence. The other two variants perform up to $\approx 50\%$ worse in preprocessing as it includes cutting with Algorithm 2, but clearly better in building and using the statistical model. Feature scoring takes $\approx 1.8sec$ for variant 2) and only $\approx 0.06sec$ for variant 3).

**Discussion:** Variant 3) is on par with variant 2) in accuracy and clearly the fastest in responding, which makes it the champion variant. Variant 1) shows a lower accuracy regardless of the statistical or ML method applied. It responses fast enough on a server\(^4\) but is probably too slow for running it on a (mobile) client. Especially, when the user sequence is long and contains many leading and trailing frames not belonging to the actual movement, the cubic complexity hampers the user experience.

6. Conclusions and Future Work

In the paper, we introduced two novel DTW variants with applications in human movement assessment and maybe beyond: one for frame matching instead of full sequence matching and another for cutting leading and trailing frames during sequence matching (1). We then used the original DTW and its variants to create three different solutions for

\(^3\)Our industry partner AIMO AB: https://www.aimo-health.com

\(^4\)AIMO’s server site implementation takes $< 10sec$ for preprocessing and scoring using this algorithm.
the cutting and matching problem (2). Finally, we experimentally assessed the accuracy and the response time of the solution variants (3). These experiments show that frame matching leads to the fastest and (almost) most accurate cutting and matching solution variant while cutting during sequence matching is the slowest and least accurate variant.

Currently, we develop a mobile movement assessment with the champion variant in the scoring engine. Therefore, we develop online algorithms allowing for pipelined preprocessing, cutting, matching, and scoring. We also assess further variants of sequence matching that are not biased towards short matches. Future work aims at 2D camera technology in the full movement assessment framework.

References


