Sports Eng (2011) 14:155–164
DOI 10.1007/s12283-011-0081-4

ORIGINAL ARTICLE

Classification of trampoline jumps using inertial sensors

Thomas Helten · Heike Brock · Meinard Müller · Hans-Peter Seidel

Published online: 15 November 2011

Abstract The automatic segmentation and classification of an unknown motion data stream according to given motion categories constitute an important research problem with applications in computer animation, medicine and sports sciences. In this paper, the scenario of trampoline motions is considered, where an athlete performs a routine consisting of sequence of jumps that belong to predefined motion categories such as pike jumps or somersaults. As main contribution, a fully automated approach for capturing, segmenting, and classifying trampoline routines according to these categories is introduced. Since trampoline motions are highly dynamic and spacious, optical motion capturing is problematic. Instead, it is reverted to a small number of inertial sensors attached to the athlete's body. To cope with measurement noise and performance differences, suitable feature and class representations are introduced that are robust to spatial and temporal variations while capturing the characteristics of each motion category. The experiments show that the approach reliably classifies trampoline jumps across different athletes even in the presence of significant style variations.

Keywords Motion classification · Inertial sensors · Segmentation · Motion features · Class representation

T. Helten (✉) · H. Brock · M. Müller · H.-P. Seidel
MPI Informatik and Saarland University, Campus E1 4,
66123 Saarbrücken, Germany
e-mail: thelten@mpi-inf.mpg.de

H. Brock
e-mail: hbrock@mpi-inf.mpg.de

M. Müller
e-mail: meinard@mpi-inf.mpg.de

H.-P. Seidel
e-mail: hpseidel@mpi-inf.mpg.de

1 Introduction

The usage of recorded human motion capture (mocap) data for motion analysis and synthesis has become an essential component in many fields such as computer animation [1], medicine [2, 3] and sports sciences [4, 5]. For example, in [4], mocap techniques have been used to analyze in air motions in elite half-pipe snowboarding. In this paper, mocap techniques are applied with the objective to automatically classify trampoline motion sequences. This constitutes a challenging application scenario because of the high complexity in terms of dynamics and recording volume. In trampolining, an athlete performs a routine that consists of a sequences of trampoline jumps that belong to predefined motion categories such as pike jump or a somersault. The classification problem then consists in automatically segmenting an unknown trampoline routine into its individual jumps and to classify these jumps according to the given motion categories. Here, further challenges arise from the fact that there is a wide spectrum on how a jump from a specific category may be actually performed by an athlete. As one main contribution of this paper, a fully automated classification procedure is described that can handle even significant performance variations.

There are many ways for recording human motion sequences, including optical, inertial and mechanical motion capture (mocap) systems, see [1, 3] for an overview. For example, optical motion capture systems, which are widely used in movie and game productions, provide very rich and easy to interpret data. On the downside, such systems impose strong restrictions concerning the size of the capture volume and lighting conditions making them difficult to use in the described trampolining scenario. Avoiding such restriction, inertial-based sensors have become a low-cost alternative, which is increasingly used in entertainment, monitoring and sports applications [2, 4–6]. The drawback of such systems is that the provided data—accelerations and angular velocities—are difficult to handle and prone to noise. Here, additional sensor information has been used to derive more robust global orientation data [7].

In this paper, a motion classification system is introduced for automatically classifying trampoline routines based on inertial sensor input, see Fig. 1 for an overview. As one contribution, it is discussed how inertial raw data is transformed into meaningful and robust feature representations underlying the classification scheme. As for the predefined motion categories, suitable training data is used to learn class representations that described the characteristics of a specific trampoline jump. Here, as a further contribution, the concept of boolean motion templates [8] is extended to the real-valued case. In particular, the notion of variance templates is introduced that allow for blending out performance variations to be left unconsidered in the classification stage. In this classification system, an athlete, being equipped with a small number of inertial sensors, performs a trampolining routine. The resulting motion stream is first segmented into individual jumps, which are then classified by comparing the segments with the previously learned class representations using a suitable similarity measure. To prove the practicability of the proposed approach, trampoline motions consisting of 750 individual jumps that comprise 13 different classes performed by four different athletes have been recorded. Various experiments are reported on which show that this procedure yields a high classification accuracy even in the presence of significant style variations across the different athletes.

The remainder of this paper is organized as follows. Firstly, some basics on trampolining (Sect. 2) as well as on motion capturing (Sect. 3) are discussed. Then, the segmentation procedure (Sect. 4) is described, various feature representations (Sect. 5) are discussed, and the class representations in form of real-valued motion templates (Sect. 6) are introduced. Subsequently, the actual classification procedure is described and evaluated, demonstrating the practicability of the proposed approach (Sect. 7). Finally, this paper is closed with an outlook on future work (Sect. 8).

2 Trampoline motions

In this section, some characteristics of trampoline motions are described, which can be exploited for segmentation and classification tasks. Trampolining is closely related to gymnastics where athletes perform a sequence of acrobatic moves. During a trampoline performance there are two alternating phases. Firstly, there is a *flight phase* in which the actual moves are performed and, secondly, there is a *contact phase* in which the athlete is in contact with the trampoline bed, see Fig. 2a. Furthermore, a contact phase can be separated into two sub-phases, a *landing phase*, where the athlete slows down, and a *takeoff phase*, where the athlete gains speed for the next jump. In the following, a trampoline *jump* is defined to be the concatenation of one takeoff phase at the beginning, one flight phase in the middle, and one landing phase at the end.

During these three phases, the athlete assumes and executes different poses and rotations, see Fig. 3. The first three subfigures (Fig. 3a–c) show different body poses assumed during the contact phase of a jump. Since these poses are determined during the landing phase of a jump, they are referred to as *landing poses*. During the flight phase the athlete assumes certain body poses (Fig. 3d–f) and/or executes rotations (Fig. 3g–i) around the body's lateral and/or longitudinal axis. A given combination of a landing pose in the takeoff phase, poses and rotations during the flight phase and a landing pose in the landing phase of a jump completely characterize a given jump. In the following, all jumps which contain the same sequence of poses and rotations are considered to belong to the same *jump class*. Table 1 shows thirteen jump classes of low and intermediate difficulty along with a short description. For example, the class "tuck jump" (TJP) starts with the pose

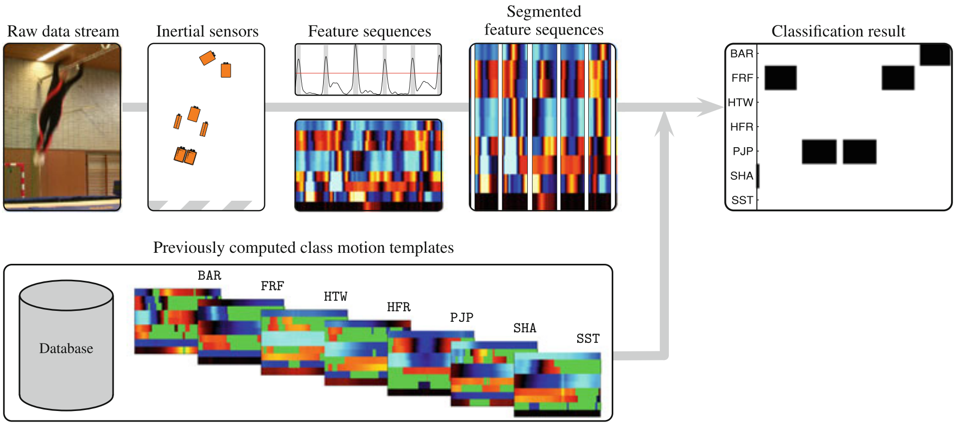

Fig. 1 Classification system used in this paper. *Bottom* class representations are computed for each of the motion categories in a preprocessing step. *Top* an unknown trampoline routine is converted into a feature sequences which is then segmented into single jump. Finally the segmented jumps are compared to the class templates and labeled with the name of the most similar class

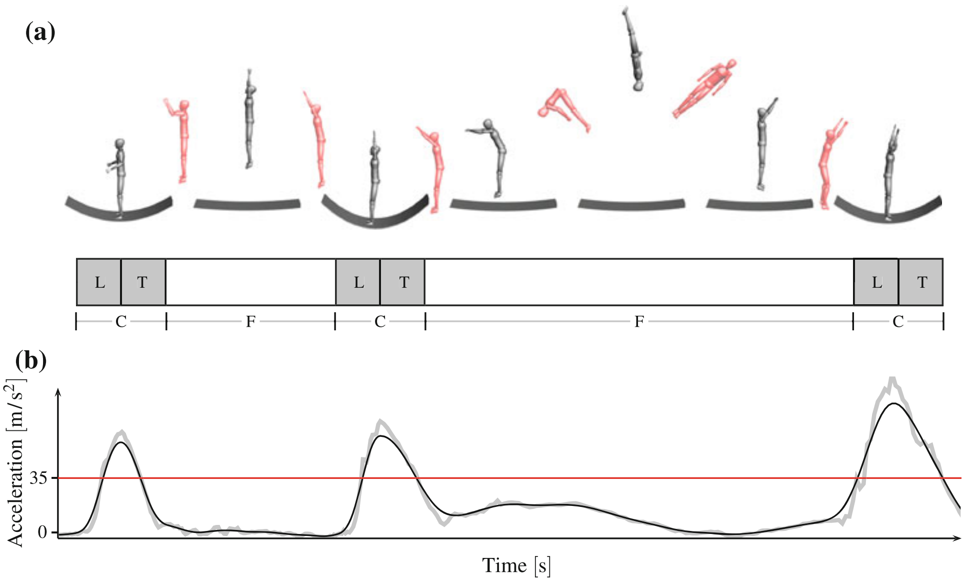

Fig. 2 Illustration of phases during a trampoline jump and the corresponding accelerations. **a** Phases of a trampoline jump comprising a contact phase (*C*), a landing phase (*L*), a takeoff phase (*T*), and flight phase (*F*). **b** Absolute acceleration as measured by sensor s_1 (*light gray*), as well as low pass filtered acceleration (*black*) and threshold ($\tau = 35$ m/s^2) as used for the automatic segmentation described in Sect. 4

“on feet” (Fe) during the takeoff phase, it continues with the pose “tucked” (Tu), and finishes with the landing pose “on feet” (Fe). Another example is the jump class BAR, also known as Barani, consisting of the landing pose “on feet” (Fe) in the beginning, a 360° somersault forwards (F360) combined with a 180° twist (T180) and ending on the feet (Fe). In trampolining, the most basic jump class is the straight jump (STR) which only consists of the pose “on feet” at the beginning and at the end of the jump. During competitions athletes have to perform so called *routines* which are sequences of jumps. Here, a routine starts with a number of straight jumps to gain momentum. After this preparation the athlete has to perform a sequence of ten jumps from a set of predefined jump classes. Then, in this classification scenario, the task is to segment the routine and to determine the classes of the performed jumps.

For the experiments, the trampoline performances of four female, non-professional athletes with intermediate skills were recorded. The athletes were given the opportunity to warm up and train various jumps which occur during the following recording. Then, they were asked to perform eight predefined routines and up to two routines that they could chose by themselves. Each routine was performed two to three times by one athlete. Between the routines the athletes were able to rest. In the experiments the effects of fatigue and training where not accounted for. In total, 109 routines with difficulty scores ranging from 0.4 to 3.1 comprising a total of 750 jumps were recorded. Out of these 109 routines 13 routines were chosen to form a routine database $\mathscr{D}_{\mathrm{R}}$. From the remaining 96 routines, for each of the 13 jump classes 16 instances—four instances for each of the four actors—were manually assembled. The resulting dataset, containing 208 jumps, is denoted as cut database $\mathscr{D}_{\mathrm{C}}$. $\mathscr{D}_{\mathrm{C}}$ is then partitioned into two databases $\mathscr{D}'_{\mathrm{C}}$ and $\mathscr{D}''_{\mathrm{C}}$ each containing two jumps per actor from all 13 jump classes, amounting to 104 jumps. Note that the partitioning is always done in a way that all databases contain routines done by all four actors from every phase of the recording.

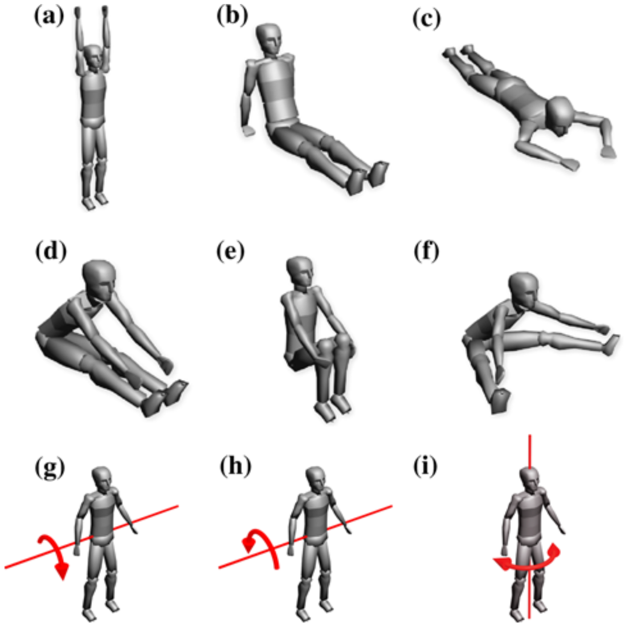

Fig. 3 **a–c** Landing poses during the contact phase: on feet (Fe), seated (Se) and on the front (Fr). **d–f** Different body poses during the flight phase: piked (Pi), tucked (Tu) and straddled (St). **g–i** Rotations around main body axes during flight phase: lateral forwards (F*), lateral backwards (B*) and twists around longitudinal axis (T*)

Table 1 Low and intermediate level jumps used for classification

ID	Description	Poses and rotations during phases		
		Takeoff	Flight	Landing
BAR	Barani	Fe	T180, F360	Fe
FRF	Front to feet	Fr		Fe
HTW	Half twist	Fe	T180	Fe
HFR	Half twist to front	Fe	T180, F90	Fr
PJP	Pike jump	Fe	Pi	Fe
SHA	Seat half twist to feet	Se	T180	Fe
SST	Seat to feet	Se		Fe
BWB	Somersault backwards piked	Fe	Pi, B360	Fe
BWS	Somersault backwards to seat	Fe	Tu, B360	Se
BWC	Somersault backwards tucked	Fe	Tu, B360	Fe
SJP	Straddle jump	Fe	St	Fe
STR	Straight jump	Fe		Fe
TJP	Tuck jump	Fe	Tu	Fe

The table shows how the jumps are composed of the poses and rotations displayed in Fig. 3

3 Sensors

As stated before, there are many ways to record human motion data using, e.g., optical, magnetic, inertial and mechanical mocap systems. A general overview of current optical mocap techniques can be found by Moeslund et al. [1], while Zheng et al. [3] give a summary on mechanical and inertial mocap systems. Now a closer look is taken on optical mocap systems on the one hand and inertial sensor-based systems on the other hand.

The most widely used motion capture systems in computer animation and movie industry are optical marker-based mocap systems as the Vicon MX[1] or the PhaseSpace.[2] Here, a set of calibrated cameras is used to record 2D images of an actor wearing a suit with retro-reflective or active markers, see Fig. 4a. From these 2D images the 3D positions of the markers can be deduced with high accuracy. Besides accuracy, the 3D marker data can be easily converted to other representations based on joint positions or joint angles. However, there are also some drawbacks, as illustrated by Fig. 4a. For example, the lighting during the recording must be dim so that the markers can be distinguished from the background. Furthermore, the setup of the systems is cumbersome as many cameras need to be carefully placed, aligned, and calibrated in order to cover the large capture volume as needed for trampoline motions, see Fig. 4b. In addition, due to the highly dynamic character of trampoline motions, markers can easily detach from the suit requiring an interruption of the recording session. Finally, optical marker-based mocap systems are rather expensive in comparison to other mocap systems.

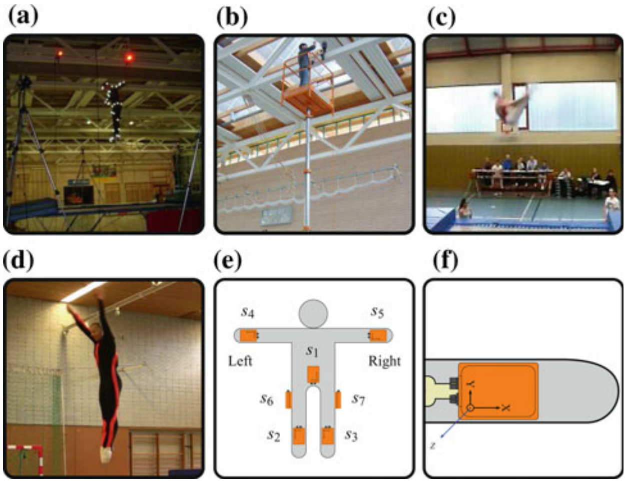

Fig. 4 **a** Recordings using optical systems require controlled lighting conditions. **b** Cumbersome setup of an optical mocap system. **c** Optical recordings suffer from motion blur in case of fast motion. **d** Actor wearing a suit containing inertial sensors. **e** Locations of the seven motion sensors attached to the human body as used in this paper. **f** Inertial sensors are attached in direction of the body's limb and can measure the limb's orientation

For these reasons, in many sports and medical applications, human motion is often recorded using much cheaper devices such as single high-speed cameras or even standard consumer camcorders. Here, the recorded video stream has to be manually annotated using specialized software tools, from which various motion parameters such as joint positions or joint angles are derived. Obviously, the quality of the used cameras highly influences the accuracy of the deduced motion data. For example, if the camera has a low temporal resolution, motion blur as shown in Fig. 4c renders the correct positioning of annotations impossible. Furthermore, as the main drawback of such video based

[1] http://www.vicon.com.

[2] http://www.phasespace.com.

methods, the manual annotation process makes large-scale experiments with a high data throughput infeasible.

In this paper, an inertial sensor-based mocap system is used consisting of seven Xsens MTx[3] sensor units denoted by s_1, …, s_7. The sensors are placed inside a suit (see Fig. 4d) together with a wireless transmission system which sends the measured data directly to a computer. For this reason, inertial sensors do not pose any restrictions on the lighting requirements and can be used in many locations, even outdoor. Fig. 4e shows the placement of the seven sensors in the setup fixed at the trunk, the forearms, the upper legs and the lower legs of the athlete. Furthermore, as indicated by Fig. 4f, the sensors are carefully aligned in a way that the sensors' local X axes are parallel to the limbs they are attached to while pointing away from the body's center. In general, inertial sensors only provide 3D accelerations $\boldsymbol{a}$ and 3D angular velocities ω which are rather unintuitive quantities prone to noise. For example acceleration sensors not only measure the acceleration due to motion but also the acceleration due to gravity. In other words, an acceleration sensor which is in rest observes a constant acceleration in the upward direction. This superposition of a motion-dependent and a direction-dependent component renders acceleration sensors alone difficult to use for motion analysis. Yet, by combining inertial sensors with other sensor types [7, 9], as done in the Xsens MTx units, it is possible to calculate full three degrees of freedom global orientations denoted by q. In this contribution, the 3D accelerations, 3D rate of turn data and three degrees of freedom orientation data are used as provided by the sensor units. Note that this data is already preprocessed ($\boldsymbol{a}$ and $\boldsymbol{\omega}$), respectively derived (q) by the sensor units from the raw data of the unit's acceleration sensors and rate of turn sensors. Now, some further notations used in the rest of the paper are fixed.

A *sensor data stream* is modeled as a sequence $D = (S_1, S_2, \ldots, S_K)$ of sensor readings $S_k \in \mathcal{S}$ for $k \in [1 : K] := \{1, 2, \ldots, K\}$ (w. r. t. a fixed sampling rate, in this case 100 Hz). Here, $\mathcal{S}$ denotes the space of sensor readings and K denotes the number of frames. Each sensor reading S_k consists of the orientations, accelerations and angular velocities measured by the seven sensor units:

$$S_k := \left(\mathsf{q}^k_{s_1}, \ldots, \mathsf{q}^k_{s_7}, \boldsymbol{a}^k_{s_1}, \ldots, \boldsymbol{a}^k_{s_7}, \omega^k_{s_1}, \ldots, \omega^k_{s_7}\right), \quad k \in [1 : K], \tag{1}$$

where $\mathsf{q}^k_s \in \mathbb{R}^3 \boldsymbol{a}^k_s \in \mathbb{R}^3, \omega^k_s \in \mathbb{R}^4$ for all $s \in \{s_1, \ldots, s_7\}$.

4 Segmentation

The first step of the proposed classification system is the segmentation of an unknown trampoline motion sequence into separate jumps. Here, the two phases are used, the contact phase and the flight phase, which segment jumps in a natural way. While the actual jump is performed during the flight phase, the athlete gains momentum during the contact phase, which is always related to a large absolute acceleration of the whole body. This absolute acceleration can be measured using the L_2-norm of the three-dimensional sensor readings of sensor s_1 which is located at the athlete's trunk. As shown in Fig. 2b, the measurement of $\|\boldsymbol{a}_{s_1}\|_2$ is rather noisy. For this reason, a low pass filter L of width corresponding to 0.1 s is applied to the measured accelerations to obtain $\boldsymbol{a} := L(\|\boldsymbol{a}_{s_1}\|_2)$. Then, those frames k were labeled that satisfy the heuristic $\boldsymbol{a}^k > \tau$ to be a contact phase frame, where τ is a suitably chosen threshold. The threshold $\tau = 35\ \mathrm{m/s}^2$ was defined experimentally using a small test dataset. The exact value was not crucial in the conducted experiments—a variation of $\pm 5\ \mathrm{m/s}^2$ is acceptable—but the threshold might be dependent on the skill level of the athletes or the properties of the used trampoline. Firstly, during the flight phase, the dynamic of the performed rotations affects the value of threshold. Fast rotations, as performed by skilled athletes, require higher thresholds than slower rotations. Secondly, depending on the jump height—or the stiffness of the trampoline bed—the acceleration during the contact phase is affected. Here, higher accelerations allow for a higher threshold, lower accelerations require a lower threshold. An experiment was conducted to get a quantitative impression how well this simple segmentation algorithm works. To this end, the 13 routines from the routine database $\mathcal{D}_{\mathrm{R}}$ were automatically segmented and the results were compared with the manually generated ground-truth segmentations which were obtained by manual inspection of a 25-Hz synchronized video. In this experiment, a jump was considered to be segmented correctly when the computed interval ends only differed from the ground-truth interval ends by a maximum of 0.15 s (15 frames using the 100 Hz Xsens frame rate). The experiment showed that in total 94% of the jumps were segmented correctly. Here, the wrongly segmented jumps were exclusively at the very beginning or at the end of the trampoline routines, where the athletes were still in the preparatory phase and the accelerations were comparatively low. Actually, all of the important jumps during the routine were segmented correctly.

5 Feature representation

As for the classification step, the raw sensor input is much too noisy and inconsistent to yield good motion representations. This is partly due to the noise introduced by the measurements itself. Even more problematic is the fact that different performances of the same jump may reveal

[3] http://www.xsens.com.

significant spatial, dynamical, and temporal differences. In particular, there are many actor-specific performance variations within a jump class. Therefore, instead of working on the raw data itself, suitable feature representations were derived from the inertial data that encode important and intuitive properties of the athlete's body configuration while being invariant under global variations such as the actor's facing direction. In Sect. 6, it is shown how to deal with local performance variations by introducing suitable class representations. Now, three different *feature types* will be introduced. The first feature type ϕ_s measures the angle between the X axis of a sensor s and the horizontal plane. If the sensor is aligned as shown in Fig. 4f, this angle is the same as the angle between the limb and the horizontal plane, see Fig. 5a. In other words, the feature ϕ_s measures the inclination of a limb with respect to the ground plane. The second feature type $\theta_{s,t}/\psi_{s,t}$ measures the enclosed angle between two limbs. Here, the only difference between $\theta_{s,t}$ and $\psi_{s,t}$ is the way the feature is computed. The feature $\psi_{s,t}$ measures the angle between limbs belonging to different extremities (Fig. 5b), while the feature $\theta_{s,t}$ measure the angle between limbs belonging to the same extremity (Fig. 5c). Finally, the third type of feature $\tilde{\omega}_s$ captures the angular velocity of the sensors X axis. In other words, this feature type measures the velocity as a limb rotates around its longitudinal axis. The exact formulas used to compute the introduced feature types are given in the "Appendix".

Based on these three feature types, in total nine *features* are defined as shown in Table 2. Mathematically, a feature is a function $F : \mathcal{S} \rightarrow \mathbb{R}$. By forming a vector of f features for some $f \geq 1$, a combined feature $F : \mathcal{S} \rightarrow \mathbb{R}^f$ is obtained which is referred to as a *feature set*. In this paper, F is equal to one of the following feature sets

Table 2 Description of the used features with feature ID and type

ID	Type	Description
F_1	ϕ_{s1}	Inclination of lower spine
F_2	ϕ_{s2}	Inclination of left lower leg
F_3	ϕ_{s3}	Inclination of right lower leg
F_4	ϕ_{s4}	Inclination of left forearm
F_5	ϕ_{s5}	Inclination of right forearm
F_6	$\theta_{s6,s2}$	Angle between left lower and upper leg
F_7	$\theta_{s7,s3}$	Angle between right lower and upper leg
F_8	$\psi_{s6,s7}$	Angle between left upper and right upper leg
F_9	$\tilde{\omega}_{s_1}$	Absolute angular velocity around the body's longitudinal axis

$$F_{\mathsf{I5A3W}} := (F_1, F_2, F_3, F_4, F_5, F_6, F_7, F_8, F_9)^T, \tag{2}$$

$$F_{\mathsf{A3W}} := (F_6, F_7, F_8, F_9)^T, \tag{3}$$

$$F_{\mathsf{I5W}} := (F_1, F_2, F_3, F_4, F_5, F_9)^T, \text{ or} \tag{4}$$

$$F_{\mathsf{I5A3}} := F_1, F_2, F_3, F_4, F_5, F_6, F_7, F_8)^T, \tag{5}$$

where the index (e. g., `I5A3W`) gives a hint on what features are included in the feature set. The part `I5` stands for the five inclination type features F_1, F_2, F_3, F_4, F_5, `A3` represents the three angular type features F_6, F_7, F_8, and `W` stands for the one angular velocity type feature F_9. This naming convention becomes important in Sect. 7 where the importance of the different feature types for the proposed classification scenario is discussed. Fig. 5e shows how a feature set $F = F_{\mathsf{I5A3W}}$ is applied to a sensor data stream D. The result is represented by a *feature matrix* $F(D) = (F(S_1), \ldots, F(S_K))$ with f rows and K columns, where in this case $f = 9$ and $K = 132$. Each row of such a feature matrix represents one feature, while each column represents the feature values $F(S_k)$ for a frame $k \in [1 : K]$.

6 Class representations

Based on feature matrices, a representation is now described that captures characteristic properties of an entire motion class. To this end, the concept of *motion templates* (MTs) is adapted, which was previously introduced in [8] by Müller and Röder. Here, given a class $C = \{D_1, \ldots, D_N\}$ consisting of N example motions $D_n, n \in [1 : N]$, first all motions are converted into features matrices X_n. Then, the idea is to compute a kind of average matrix. However, note that the N motions generally have a different length. Therefore, dynamic time warping is applied to temporally align the motions and to warp all feature matrices to yield the same length. The average matrix X_C over the warped feature matrices is then referred to as class motion template. Müller and Röder apply this concept to boolean-valued features

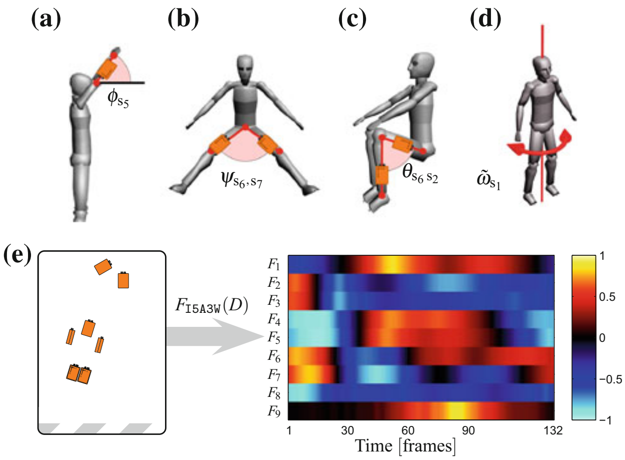

Fig. 5 Illustration of the various feature types and the feature representation

matrices yielding boolean feature matrices. As a consequence, regions in the class MT with the values zero/one indicate periods in time (horizontal axis) where certain features (vertical axis) consistently assume the same values zero/one in all training motions, respectively. By contrast, regions with values between zero and one indicate inconsistencies mainly resulting from variations in the training motions (and partly from inappropriate alignments). This property of MTs can then be used to automatically mask out the variable aspects of a motion class when being compared with an unknown motion data stream. This makes motion classification very robust even in the presence of significant performances differences, see [8] for details.

Now, the concept of motion templates is applied to the trampoline classification scenario. Let $\mathscr{C} = \{\mathsf{BAR}, \ldots, \mathsf{TJP}\}$ be the set of all considered jump categories, and let $C \in \mathscr{C}$ one of the motion classes. Using a feature set F, all example motions contained in C are converted into feature matrices. However, opposed to the approach by Müller and Röder [8], the proposed features are real-valued, so that some modifications in the MT computation are needed. First of all, to balance out the importance of the various features contained in F, all features need to be normalized to approximately have the same range. In this case, as described in the "Appendix", the feature were designed to lie roughly in the range $[-1, 1]$. Note that this normalization does not depend on the underlying data (i. e., it does not change with an athletes performance). As an example, Fig. 6a–c shows the resulting feature matrices of three example jumps from the class $C = \texttt{BAR}$. Then, as proposed by Müller and Röder, the normalized feature matrices are temporally warped and an average matrix X_C is computed, see Fig. 6d. Now, starting with real-valued instead of boolean-valued feature matrices, the inconsistencies are not revealed as described by Müller and Röder. Instead, a *variance template* V_C is computed, which encodes the entry-wise variance of the N warped feature matrices, see Fig. 6e. Here, the idea is that inconsistent regions in the real-valued feature matrices induce larger variances than consistent regions. Now the variance template can be used to mask out inconsistencies in X_C. In this setting, those regions of X_C are masked out, where the value in V_C is larger than the 75% quantile of all values of V_C. In other words, the 25% most variant values are ignored, see Fig. 6f. Here, the percentage value of 25% has been determined experimentally, yielding a good trade-off between capturing sufficient motion characteristics while suppressing unwanted motion variations. Mathematically, the masking is modeled as a separate *mask matrix* $M_C \in \mathbb{R}^{f \times K}$, where a value of 0 means that the value is masked out. The entries of M_C can be computed in the following way:

$$M_C(i,j) := \begin{cases} 1 : V_C(i,j) \leq Q_{75\%}(V_C) \\ 0 : \text{else} \end{cases} \tag{6}$$

for $i \in [1:f]$ and $j \in [1:K]$. Here, $Q_{75\%}(V_C)$ is the 75% quantile of V_C. Later in this paper, a scenario is introduced where the influence of certain feature functions needs to be amplified. This can be modeled by allowing other values beside 0 and 1 inside the mask matrix.

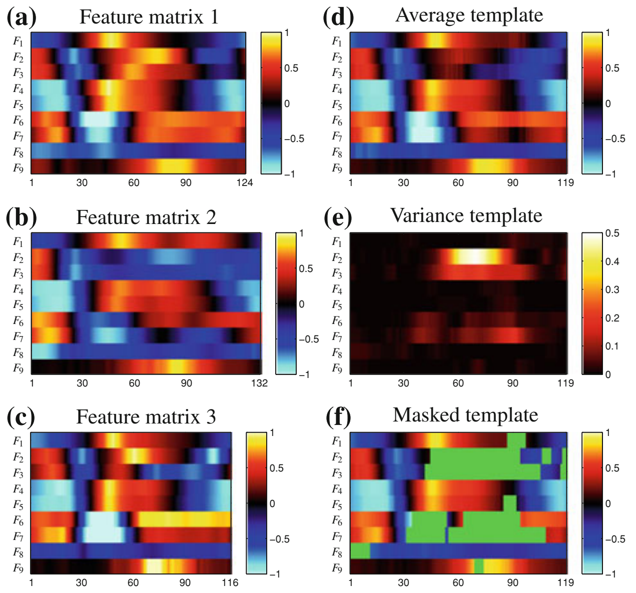

Fig. 6 Template computation: **a–c** feature matrices for three different jumps from the class BAR. **d** Average of aligned feature matrices (average template). **e** Variances of aligned feature matrices (variance template). **f** Template, where regions with 25% highest variances are masked out (masked template)

7 Classification and experiments

For the classification an unknown jump is locally compared with all class MTs X_C for $C \in \mathscr{C}$ and then labeled according to the class MT having the smallest distance to the jump. In the following, let $Y \in \mathbb{R}^{f \times L}$ be the feature matrix of an unknown jump to be classified, where L is the length of the jump and f is the number of features. The used distance measure is a variant of dynamic time warping (DTW) as described by Müller and Röder, but the local cost measure c has to be adjusted in order to be compatible with the proposed masking. Let $m(k) := \sum_{i=1}^{f} M_C(i,k)$, then the masked local cost measure is defined as

$$c(k,\ell) := \left(\frac{1}{m(k)} \sum_{i=1}^{f} M_C(i,k) |X_C(i,k) - Y(i,\ell)|^2 \right)^{\frac{1}{2}}, \tag{7}$$

for $m(k) \neq 0$ and $c(k,l) = 0$ for $m(k) = 0$, where $k \in [1 : K]$ and $\ell \in [1 : L]$. Now, the *distance* Δ_C between a class C with MT X_C and mask M_C and a feature matrix Y is defined as

$$\Delta_C(Y) := \frac{1}{K}\mathrm{DTW}(X_C, Y), \tag{8}$$

where DTW denotes the DTW-distance between the sequences of columns defined by X_C and Y using the local cost measure c. Finally, the classification problem for an unknown jump with feature matrix Y can be solved by identifying the class $C \in \mathscr{C}$ which has the smallest distance $\Delta_C(Y)$.

7.1 Influence of feature types

Firstly, an experiment is reported on that is used for investigating how the quality of the classification depends on the used feature types. To this end, confusion matrices are used, which give a qualitative impression which jump classes are classified correctly, and which jump classes are confused among each other. Such confusion matrices display the ratio of how many motions from a given class (abscissa) were classified as a certain class (ordinate), where dark entries represent a high percentage of motions. If the used feature types discriminate jump classes well, this would result in a dark diagonal leading from the top left of the matrix to the bottom right. In this experiment, jumps from database $\mathscr{D}'_\mathrm{C}$ were used to learn the motion templates while $\mathscr{D}''_\mathrm{C}$ was used for evaluation.

Figure 7 shows the confusion matrices for the four different feature sets defined in Sect. 5, where the feature set F_{I5A3W} includes all feature types, while the feature sets $F_{\mathrm{A3W}}, F_{\mathrm{I5W}}$, and F_{I5A3} lack one of the feature types. In Fig. 7a it can be seen that the feature set F_{A3W}, which lacks the inclination aspect, performs worst. This is expressed by the many high-valued off-diagonal entries which are an indication for massive miss-classifications. This shows that the feature set F_{A3W} is too sparse for distinguishing different jump categories. Figure 7b shows the results for the feature set F_{I5W}. Here, while most of the jumps were classified correctly, the jump classes PJP, SJP, and TJP are mixed up among each other. This is due to the fact that these jump classes only differ in the configuration of the legs during the flight phase. For example, in both jump classes PJP and SJP the legs are straight to the front during flight. The only difference is that in the jump class SJP the legs are additionally straddled. If the feature set contains inclination and angle feature types, as shown in Fig. 7c, the classification works better for the jump classes PJP, SJP, and TJP, but now other jump classes as STR and HTW get mixed up. Here, these two jump classes only differ in a rotation around the bodies longitudinal axis. For this reason, the feature that measures the angular velocity is needed to capture the difference between the two jump classes. Finally, Fig. 7d shows that the proposed feature set F_{I5A3W} almost perfectly separates all jump classes from each other.

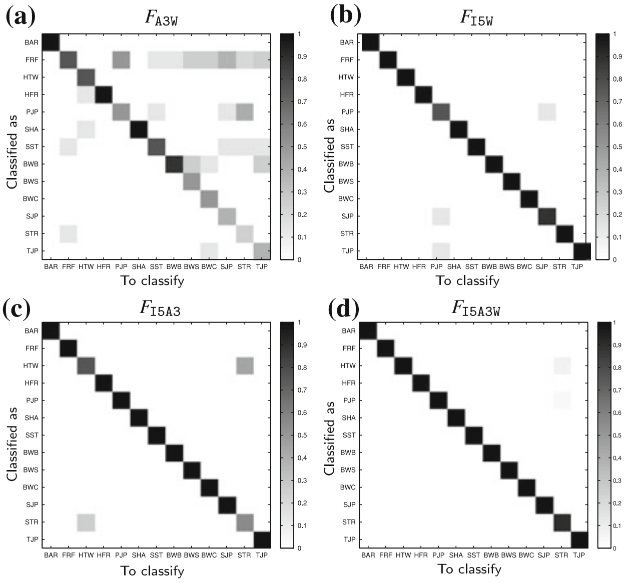

Fig. 7 Confusion matrices showing the influence of the different feature types. The learning database is $\mathscr{D}'_\mathrm{C}$ while the evaluation database is $\mathscr{D}''_\mathrm{C}$. In all four cases the quantile mask introduced in Sect. 6 is used

7.2 Routine classification

As main experiment, the automatic segmentation from Sect. 4 is combined with the classification introduced above. Here, the overall system performance is evaluated in a realistic trampolining scenario. Furthermore, it is discussed how the used masking affects the retrieval results. For this evaluation, the thirteen routines from the database $\mathscr{D}_\mathrm{R}$ are used for evaluation, while the motion templates are again learned from the databases $\mathscr{D}'_C$. Furthermore, quantile masks are used as defined in Eq. 6. Figure 8a displays a classified routine where the black regions represent the automatic classification result and the red rectangles indicate the manual ground-truth annotations. It can be seen that for this example 14 out of 18 jumps were classified correctly. Here, for example, the misclassification of the jump SHA (frames 2,200–2,350) with the class SST is due to the fact that the feature F_9 is the only feature which is actually able to capture the difference between this two classes. Similarly, the confusion between STR (frames 3,050–3,200) and HTW can be explained. In such cases, the influence of the feature F_9 on the local cost measure c is not large enough (its only one ninth compared to the features $F_1, \ldots, F_8$). In order to better separate the confused jump classes from each other, the influence of the feature F_9

can be increased by replacing all ones in the quantile mask matrices of the class representations belonging to feature F_9 with some value larger than one (five in this experiments). The effect of such so called *weighted mask matrices* can be seen in Fig. 8c, where the previously misclassified jumps `SHA` and `STR` are now classified correctly. The misclassifications between the jump `TJP` (frames 1,950–2,075) with `STR` and the jump `PJP` (frames 2,380–2,500) may be explained as follows. Firstly, it has to be noted that the performance variations between jumps that belong to the same class are often significant—even within the jumps of the same athlete. Such variations are actually masked out by the local cost measure. Now, the differences between two jump classes such as `TJP` and `STR` or `PJP` and `SJP` are often subtle and only refer to a single motion aspect. It may happen that such aspects are actually masked out by the proposed masking concept, which then leads to unwanted confusion. These examples indicate the trade-off between robustness on the one hand and discrimination capability on the other hand.

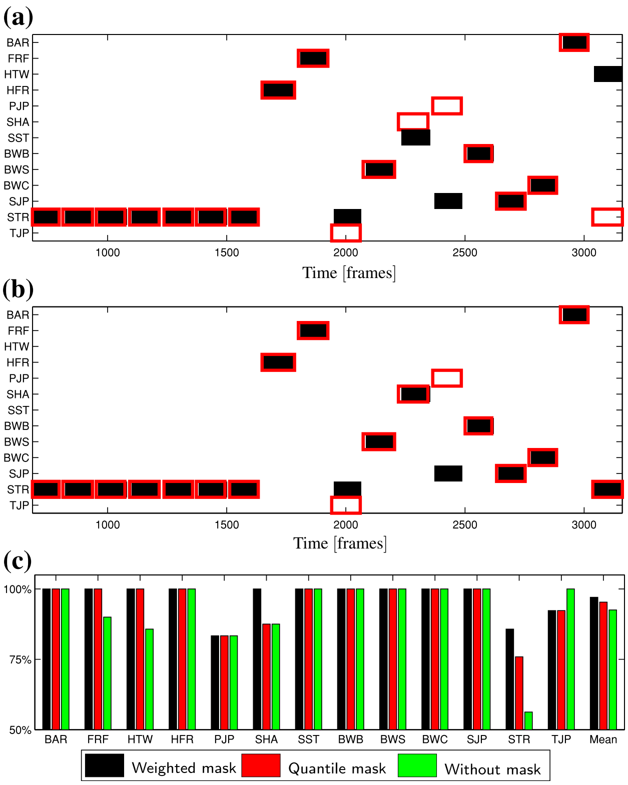

Fig. 8 Classification results for routine scenario (*filled rectangles* automatic classification, *outlined rectangles* manual annotation). The class representations where learned using database $\mathcal{D}'_{\mathrm{C}}$, while the classified routines are taken from database $\mathcal{D}_{\mathrm{R}}$. **a** Classification result for an example routine when using quantile masks. **b** Classification result for the same routine when using weighted masks. **c** Classification accuracy for the 13 learned jump classes using different masking techniques

In addition to this qualitative analysis, a quantitative analysis was done to measure the classification accuracy for each jump class. An automatically segmented jump is considered as classified correctly if its segment boundaries lie in the neighborhood of an annotated jump (using a tolerance of 0.15 s) and if the computed class label coincides with the annotated label. In this evaluation, three different masking strategies are considered: binary masking (quantile mask), weighted masking, and no masking at all. As Fig. 8c shows, the classification results are very good for most classes independent of the used masking strategy. This again shows that the proposed features are capable to capture relevant motion characteristics. When using the weighted mask matrix the classification results are generally better than when using the binary mask. Especially, the jump classes `SHA` and `STR`, as in the previous paragraph, benefit from the use of weighted masking. A good example how masking in general improves the classification results are the jump classes `FRF`, `FTW`, and `STR`. Here, the variances, within the jump classes are very high among actors and result in misclassified jumps, when no masking is used. On the contrary, the jump class `TJP` does not benefit from masking out variant regions, since, in this case, these regions also contain the only information that is able to discriminate this jump class from other jump classes.

8 Conclusion and future work

In this paper, a system for the automatic segmentation and classification of trampoline routines based on inertial sensor input has been introduced. Here, the motivation for using inertial sensors was that such sensors better deal with dynamic motions and do not impose constraints as far as the recording volume or lighting conditions are concerned. As the main contribution, suitable feature representations have been discussed that are invariant to spatial variations and robust to measurement noise. Based on this feature representations, real-valued motion templates were introduced that grasp the characteristics of an entire jump class. To handle significant performance variations, a masking scheme based on variance templates was introduced. Furthermore, a weighting strategy to enhance the influence of certain features was presented. For future work it is planned to apply these techniques in an online scenario, where the performance of an athlete is assessed and feedback for performance improvement is given directly. A possible means of such feedback might be the sonification of certain motion parameters with respect to a learned reference performances.

Appendix: feature computation

In the following, it is shown how the features ϕ_{s}, $\theta_{\mathrm{s,t}}$, $\psi_{\mathrm{s,t}}$ and $\tilde{\omega}_{\mathrm{s}}$ can be computed. To this end, it is assumed that the sensor data stream is defined as shown in Sect. 3 The rotations inside the sensor data stream must be given in a suitable rotation representation as for example unit quaternions (see [10]). Furthermore, if q is a rotation in a given representation, then let $\mathsf{q}[v]$ be the 3D vector $\boldsymbol{v}$ rotated by q. The features ϕ_{s}, $\theta_{\mathrm{s,t}}$, $\psi_{\mathrm{s,t}}$ and $\tilde{\omega}_{\mathrm{s}}$ are now defined as

$$\phi_{\mathrm{s}} = 1 - \frac{2}{\pi}\arccos\left\langle (0,0,1)^T, \mathsf{q}_{\mathrm{s}}\left[(1,0,0)^T\right]\right\rangle, \tag{9}$$

$$\theta_{s,t} = 1 - \frac{2}{\pi}\arccos\left\langle \mathsf{q}_{\mathrm{s}}\left[(1,0,0)^T\right], \mathsf{q}_{\mathrm{t}}\left[(1,0,0)^T\right]\right\rangle, \tag{10}$$

$$\psi_{s,t} = 1 - \frac{2}{\pi}\arccos\left\langle \mathsf{q}_{\mathrm{s}}\left[(-1,0,0)^T\right], \mathsf{q}_{\mathrm{t}}\left[(1,0,0)^T\right]\right\rangle, \text{ and} \tag{11}$$

$$\tilde{\omega}_{\mathrm{s}} = \frac{2}{3\pi}\left|(\boldsymbol{\omega}_{\mathrm{s}})_x\right|. \tag{12}$$

Here, $\langle\cdot,\cdot\rangle$ denotes the scalar product of two vectors, while $(\cdot)_x$ is the x-component of a vector. Please note, that the features as defined above have roughly a range of [−1, 1]. Note that this normalization is data-independent. This is required, as mentioned in Sect. 6.

References

1. Moeslund T, Hilton A, Krüger V (2006) A survey of advances in vision-based human motion capture and analysis. Comp Vis Image Underst 104(2):90–126
2. Boissy P, Choquette S, Hamel M, Noury N (2007) User-based motion sensing and fuzzy logic for automated fall detection in older adults. Telemed J E Health 13(6):683–694
3. Zheng H, Black N, Harris N (2005) Position-sensing technologies for movement analysis in stroke rehabilitation. Med Biol Eng Comput 43:413–420
4. Harding J, Mackintosh CG, Hahn AG, James DA (2008) Classification of aerial acrobatics in elite half-pipe snowboarding using body mounted inertial sensors. Eng Sport 7:447–456
5. Ohgi Y, Ichikawa H, Miyaji C (2002) Microcomputer-based acceleration sensor device for swimming stroke monitoring. JSME Intern J Ser C Mech Sys Mach Elem Manuf 45(4):960–966
6. Sabatini A, Martelloni C, Scapellato S, Cavallo F (2005) Assessment of walking features from foot inertial sensing. IEEE Trans Biomed Eng 52(3):486–494
7. Kemp B, Janssen AJMW, van der Kamp B (1998) Body position can be monitored in 3d using miniature accelerometers and earth-magnetic field sensors. Electroencephalogr Clin Neurophysiol Electromyogr Mot Control 109(6):484–488
8. Müller M, Röder T (2006) Motion templates for automatic classification and retrieval of motion capture data. In: Proceedings of the 2006 ACM SIGGRAPH/Eurographics Symposium on Computer Animation. ACM Press, New York, pp 137–146
9. Luinge HJ, Veltink PH (2005) Measuring orientation of human body segments using miniature gyroscopes and accelerometers. Med Biol Eng Comput 43(2):273–282
10. Shoemake K (1985) Animating rotation with quaternion curves. ACM SIGGRAPH Comp Gr 19(3):245–254