Capturing and Analysing Movement using Depth Sensors and Labanotation

Börge Kordts
University of Lübeck
Lübeck, Germany
boerge.kordts@gmx.de

Bashar Altakrouri
Ambient Computing Group
Institute of Telematics
University of Lübeck
Lübeck, Germany
altakrouri@itm.uni-luebeck.de

Andreas Schrader
Ambient Computing Group
Institute of Telematics
University of Lübeck
Lübeck, Germany
schrader@itm.uni-luebeck.de

ABSTRACT
Full body interactions are becoming increasingly important for Human-Computer Interaction (HCI) and very essential in thriving areas such as mobile applications, games and Ambient Assisted Living (AAL) solutions. While this enriches the design space of interactive applications in ubiquitous and pervasive environments, it dramatically increases the complexity of programming and customising such systems for end-users and non-professional interaction developers. This work addresses the growing need for simple ways to define, customise and handle user interactions by manageable means of demonstration and declaration. Our novel approach fosters the use of Labanotation (as one of the most popular movement description visual notations) and off-shelf motion capture technologies for interaction recoding, generation and analysis. This paper presents a novel reference implementation, called Ambient Movement Analysis Engine, to allow for recording movement scores and subscribing to events in Labanotation format from live motion data streams.

Author Keywords
Movement Analysis, Labanotation, Online Gesture Recognition, Depth sensors, 3D Motion, Natural User Interface

ACM Classification Keywords
H.5.2. Information Interfaces and Presentation (e.g. HCI): User Interfaces - Input devices and strategies; H.5.m. Information Interfaces and Presentation (e.g. HCI): Miscellaneous; I.5.0 Pattern Recognition: General

INTRODUCTION
Ambient interactive applications and systems depend highly on context information for interaction acquisition and delivery. Human body movements currently resemble an essential part of this information. Thus, increasing advancements in utilising movements, in the context of HCI, on commercial and research levels have encouraged wide adoption of gestures in commodity devices, intensive use of gestures in interactive applications and emerging calls for flubbed interactions [7].

Research in the area of Natural User Interfaces (NUIs) has clearly demonstrated the use of motion gestures, ranging from simple hand gestures to potentially more complex and richer gestures involving multiple body parts. Recently, affordable depth sensors such as Microsoft Kinect[1] and Asus Xtion PRO LIVE[2] motion sensors have presented a very good playground and opened opportunities for creating novel interaction techniques, where human body movements are used to control games and applications.

Following this success, the development of depth sensors is being carried out strongly in mobile devices as well [6]. This has opened up a huge potential for large and diverse user groups (including physically challenged users) to experience new forms of interactions different from traditional input techniques. Therefore, NUIs have found their way into wide range of AAL scenarios including mobile applications, games, interactive television, smart kitchen appliances, etc.

Full body interactions enhance the opportunities to enrich the design space and customisation of interactive applications in ubiquitous and pervasive environments; nevertheless, the complexity of programming and customising such systems poses real challenges to professional interaction developers. Non-professional interaction developers are even more challenged to create custom interactions (i.e., gestures) for applications.

End-user programming of interactive applications and systems is certainly following a strong trend to allow users to design, develop, configure, and customise their interaction environments according to their own needs and requirements. Hence, easier and more manageable approaches for defining and using motion gestures are desired.

Our work aims at reducing the realisation effort of configurable gesture control in AAL systems. Moreover, it ad-

[1]http://www.microsoft.com/en-us/kinectforwindows/, latest access on 10.04.2015.
[2]http://www.asus.com/Multimedia/Xtion_PRO_LIVE/, latest access on 10.04.2015.

dresses the growing need for simple ways to define, customise and handle user interactions.

In order to define and use motion, developers are often restricted to the state of the art pattern recognition techniques that typically require training a classifier with a set of available training samples [4]. One-shot classifiers reduce the effort required for the system setup as in [12]. Trained classifiers are widely available; however, they often restrict the developers to a required set of features and limited gestures. Our approach fosters the use of off-shelf motion capture technologies for interaction modelling, integration and delivery using Labanotation as one of the most popular visual notations for documenting physical movements. While this notation is commonly used for capturing and analysing movements in choreography, physical therapy, rehabilitation and drama, we utilise the notation in the context of interactions as part of an engine called Ambient Movement Analysis Engine. While defining gestures by demonstration offers an easy and convenient way to define gestures in many cases, declaration is more precise and accurate. It offers better means to edit, adjust, debug, and optimise parts of the gesture individually. We rely on a hybrid approach to utilise the benefits of both approaches. Hence, the engine allows to easily code, analyse and integrate gestures modelled as Labanotation movement sequences in interactive applications.

In this paper, we mainly describe the architecture, implementation and preliminary evaluation of the Ambient Movement Analysis Engine. We highlight and discuss our contribution in two parts, namely:

- **The Movement Analyser**: This module is mainly responsible for detecting movement sequences and creating movement scores. Movements are acquired from camera streams by using computer vision techniques and coded into Labanotation movement representation models. Hence, the engine aims at providing detailed descriptions on performed movements for further documentation, analysis and processing.
- **The Movement Provider**: This module is responsible for subscription and delivery of interaction (i.e., gesture) events based on movement scores. Hence, adequate eventing techniques are used to give software developers the opportunity to define gesture events in a human and machine-readable format to trigger actions. The flexibility of the used notation allows for subscriptions to macro (general) or micro (very detailed) movements as required by interaction scenarios.

  To our best knowledge, our paper is the first to target a gestures provider based on Labanotation, which excels in providing readable gestures useful for debugging (observing correctly/incorrectly performed parts of the gesture), programming-free authoring of gestures and a manageable event-subscriber user interface.

RELATED WORK

This section covers selected previous work in the area of movement description languages, as well as movement acquisition and capturing.

Labanotation and Movement Description Languages

Despite the relevance of movement description and documentation, both remain unresting problems for many fields such as dance choreography, movement rehabilitation, motion recognition and analysis, and human movement simulation [3]. A very simple but informal approach to describe gestures is sketching of key poses; however, this approach and other informal approaches suffer from different drawbacks including ambiguity, lack of accuracy, missing temporal information, etc [1]. Formal movement description and analysis approaches and systems offer more structured means to deal with movements. Kahol et al.'s [11] calls for a formalised language that facilitates teaching and learning movement styles, permits universally understood scores and provides a universal language to communicate motion. One of the most used human-readable and visual notation systems for describing and analysing movements is called Labanotation [10]. We have adopted Labanotation in our approach because it is used in fields heavily relying on movement descriptions such as choreography and dance; furthermore, it is successfully used in HCI for interaction description and analysis as in [1, 2, 3, 15].

While dedicated Labanotation books, such as [10], offer thorough explanation about the language, herein, we briefly introduce its main constructs. Labanotation is a visual notation that is composed of a large number of abstract symbols, which are used to present and document various movement qualities such as direction, level (high, middle and low), duration, dynamics and quality (often indicated by effort), involved body parts, etc. For clarity, Figure 1 illustrates two gestures (used in our evaluation) modelled in Labanotation as simple interaction sequences. The movements are written on a vertical "body" staff, which is split into vertical columns and a central line that separates the score into left and right sides to present the left or right side body parts involved in the movement. Labanotation is read from bottom to top in a vertical direction. Movements for different body parts are arranged into different columns. By default the inner four columns (on each side) are already assigned to support (i.e., the distribution of body weight on the ground), legs, whole body and arms (presented in Figure 1a by the columns (1) to (4) respectively). All other columns can be individually assigned to other body part (e.g., hands, feet, fingers, toes, digits and toes). Furthermore, empty columns are simply not relevant for the modelled gesture.

The beginning of movement is denoted by two parallel horizontal lines connecting the two vertical outer lines (marked in (5) in Figure 1a). For defining a starting pose, the corresponding symbols should be placed below the start lines. If no particular starting pose is required, this section of the staff should be left empty. Analogously, two parallel horizontal lines at the top of the staff may be used to denote the end of the gesture. Timing is presented by measures and beats (marked by (7) and (8) in Figure 1a). Beats are written as short horizontal lines crossing the centre line of the score. One beat in a score may denote one or more beats in a piece of music. For the Ambient Movement Analysis Engine, beats are defined in milliseconds. A measure on the other hand consists of several beats and is written as a long horizontal line, which is called bar or measure mark, crossing the centre line. When used in describing interactions for NUIs, the measure can be used to separate gestures into different sections that may, for example, indicate the different essential stages or parts of the interaction [3].

The rest of Figure 1a is read as follows: (6) The starting position of the right arm should be placed in the centre (orthogonal to the rest of the body) and in forward diagonal position (arm lies between forward and right side). (7) The measure presents the total time required consisting of 5 beats. (8) Beats in our gesture examples are defined as 200 milliseconds units. (9) The arm should move to forward. Finally, (10) the arm is supposed to return back to the starting position.

Figure 1b illustrates the throw gesture and is read as follows: Columns (1) and (2) present the left upper and lower arms respectively. (3) The starting position of the upper arm should be orthogonal to the left side of the body and (2) the lower arm should point up, together both forming an ”L” shape. (5) The lower arm moves towards front and (6) ends in the front position.

For authoring and modelling gestures, we have utilised our previous work on the *Interaction Editor* [2] that offers an environment for the generation and management of visual Labanotation scores as well as the conversion between visual and a complementary Extensible Markup Language (XML) representation.

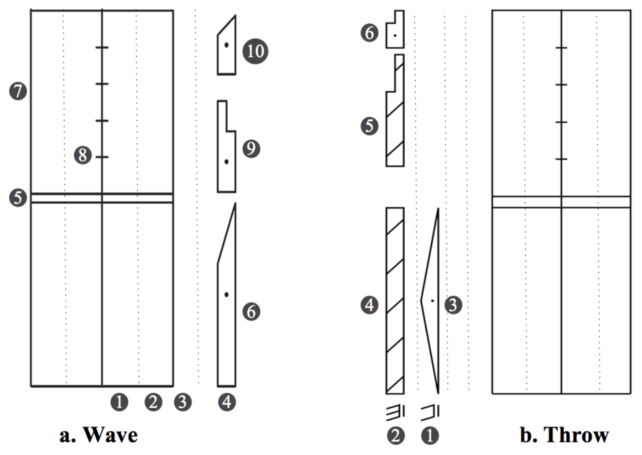
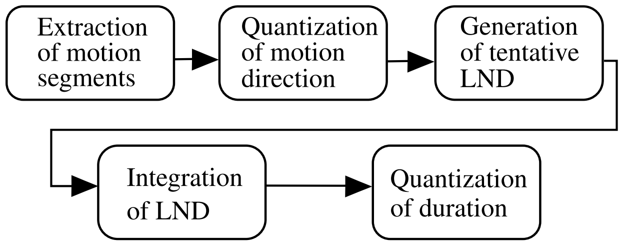


Figure 1: The (a) wave gesture and (b) throw gesture modelled as two simple interaction sequences.

Movement Acquisition and Capturing

In computer science, different approaches were reported for motion capture. Hachimura and Nakamura [9] proposed an algorithm that generates Labanotation data (LND) from motion capture data (mocap). Their motion capture approach decomposes motion sequences into motion segments by differences in each joint’s velocity. Next, the direction and level of the movement of each joint are quantised within every frame, by dividing the space from 33 into 27 directions. The two forward directions in Labanotation are combined to one forward direction and the two backward directions are combined to one backward direction. The result of the quantisation is a tentative representation, which means that a part of it may be redundant. These ambiguous parts are two sequences of symbols that have the same meaning; therefore, rewriting rules are applied to remove these redundancies. Using this as a basis, well-formed LND is generated by quantising the duration of the symbols. Figure 2 shows the processing chain of this approach. Yu et al. [18] proposed using user-generated Labanotation for motion capture database search and retrieval.



Extraction of motion segments
Quantization of motion direction
Generation of tentative LND
Integration of LND
Quantization of duration

Figure 2: The five processing steps of the algorithm proposed by Hachimura and Nakamura.

More recently, Zacharatos et al. [19] have used the effort component in Laban Movement Analysis for emotion recognition using a depth camera. [5] has presented an approach for generating Labanotation from motion capture data. Stored Bio-Vision Hierarchical data (BVH) files were used to generate Labanotation scores. The system was used to capture dance gestures and generate the corresponding dance scores for five body parts, namely the arms, legs and the head. In order to generate the score from BVH files, the data stored in Euler angles was firstly converted to world coordinate data. Similar to [9], the movement for each body part was then analysed separately, by identifying the one of the 27 subspaces in which the body part is ranging. This analysis is done per frame but the results were finally given per beat. By combining all parts, all simple motions of the five body parts were generated.

Real-time classification of dance gestures was described by Raptis et al. [17]. Their classification consisted of feature extraction using an angular representation of the skeleton, a cascaded correlation-based classifier and a distance metric based on dynamic time-warping. They used common pattern recognition methods, including training data and a classifier to identify 28 different dance gestures. The gestures were performed by professionals and several dancers of various skills in order to have an oracle and training data. The angular representation used for the classification system was based on the assumption that the torso itself is rigid and each limb, namely arm and leg, has two joints. In this representation, first-degree joints (the ones next to the torso) are robust, since they only depend on the torso itself. Second-degree joints are not robust because the origin of the spherical coordinate system is not part of the rigid body. Euler angles carry the risk of gimbal lock, which can be avoided by quaternions; however, this issue was considered to be unlikely and was not dealt with in their work.

The aforementioned approach was recently followed and extended by Miranda et al. [16]. Their goal was mainly to avoid problems resulting in a complete invariant angular representation. This approach followed a pure joint-angle representation to provide invariance to sensor orientation. The representation used in their work aimed to provide online gesture recognition from key poses, which are recognised by a multi-class support vector machine (SVM). In order to determine performed gestures efficiently, a decision forest containing all gestures and their key poses were used. With the those improvements, the angular skeleton representation by [17] that has been proven robust was enhanced further.

Major parts of our work aim at movement capture and acquisition; nonetheless, the Ambient Movement Analysis Engine can be demarcated from traditional gesture recognition since the recognition is not based on a black box classifier. The recognition is ruled by movement representations, where movement scores are generated and compared. Our approach focuses on the detection and recording of physical movements using Labanotation, in order to simplify the use and utilisation of physical movements interactive applications.

Many common gesture recognition techniques, e.g., [17], rely on using a classifier to label motion sequences and provide confidences. However, they don't provide detailed information on the performed sequences in terms of a motion score and they don't support user-defined motion scores. In contrast, we believe that motion sequences offer our Ambient Movement Analysis Engine more flexibility, where gestures are simply described by motion scores. Those scores are used for recognition independently from training data (not even one-shot training data). The flexibility makes it possible to provide off-shelf motion capture technology; moreover, user-defined motion scores can be used to trigger customised events and to document performed human body movements. Additionally, the Ambient Movement Analysis Engine does not only provide the motion score as in [9] and [5] but also provides gesture recognition of custom gestures modelled in Labanotation as well.

SYSTEM ARCHITECTURE

The conceptual design of the Ambient Movement Analysis Engine is shown in Figure 3. The user can register Labanotation scores in XML format and can send the skeleton stream to the engine. We have used the XML schema for Labanotation scores, which was introduced in [2, 3]. The XML representation can be used to enable a detailed movement analysis for users. Scores can be visually viewed using the *Interaction Editor* [2]. The schema provided in [2, 3] does not cover the entire set of Labanotation symbols; however, it is sufficient to cover the requirements of this work, in addition to its adequate analysis of movements in most use cases. Skeleton data is used as input, in order to avoid specific driver formats and to support various data sources; nonetheless, this feature costs additional preprocessing to convert the motion capture data into a compatible skeleton.

The engine aims to recognise ① the registered Labanotation scores from ② generated scores extracted from the skeleton stream. ③ The occurrence of the registered sequences, which

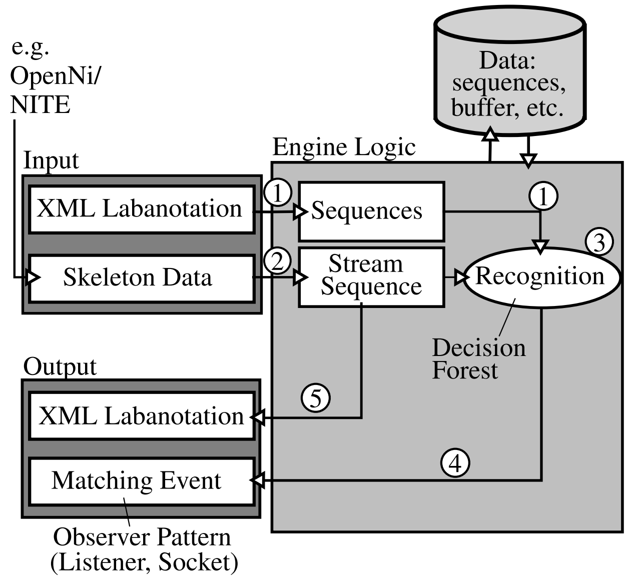


Figure 3: Ambient Movement Analysis Engine conceptual design.

are subsequences in the stream, is detected. ④ Once a gesture is recognised, an event is triggered in order to be processed by the subscribed application. Additionally, ⑤ Labanotation scores, which are based on the input skeleton stream, can be exported in XML format, which enables other services to consume motion sequences in Labanotation format.

The system overview, using the event triggering techniques, is illustrated in Figure 4. The mechanisms are based on listeners and network sockets. In order to guarantee reusability and extensibility, the template method pattern is used. The template method is a pattern of the *Gang of Four* (GoF) [8], which makes it possible to replace parts of the algorithm, for instance the skeleton representation or the sequence representation, to customise representations as well as detection algorithms. Interface realisations are used to implement the desired actions. Different derived classes can be used to realise different actions. Template implementations of Labanotation scores and 3D skeleton data are provided by the engine. Furthermore, other implementations or the replacement of specific parts of the algorithm are possible. The template approach allows developers to use the Ambient Movement Analysis Engine in other fields or for diverse purposes.

ENGINE LOGIC

The gesture recognition algorithm developed for the engine is divided into two main parts. First, a feature extraction part is used to generate Labanotation scores. Due to the way Labanotation scores are used to represent motion, the recognition is based on key frames for each body part. The second part of the algorithm is the recognition of motion sequences using decision forests and the generated scores. Both parts, together, are the basis for this work to provide movement events based on the motion capture data provided by a depth sensor.

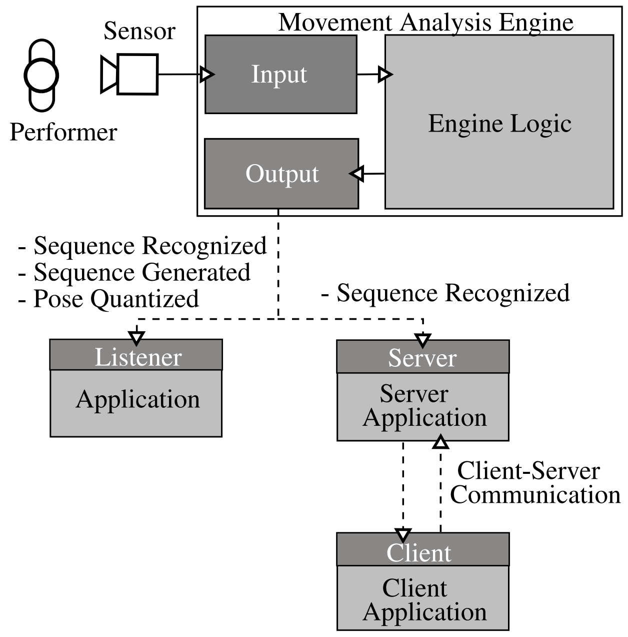
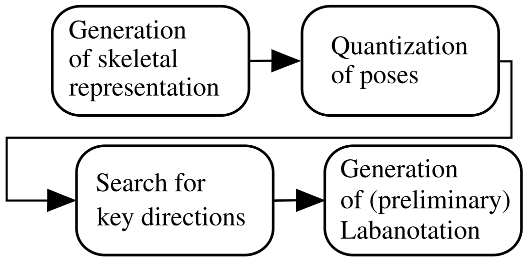


Figure 4: Ambient Movement Analysis Engine eventing and communication model. Interfaces for the listeners and realisations for client and server are provided by the engine and can be used by custom applications. An application can use listeners to access the data directly or a client application can be registered to the server application to receive network events.

The generation of Labanotation scores works similar to the algorithm described by [9]. At first, a custom representation of the motion capture data is generated from a general skeleton. It contains the specific information, torso basis and relative positions, which is required to generate Labanotation scores. Then, poses are quantised using the provided skeleton data. Next, key directions are searched in order to find the start and the end of each of the body part's movements. Finally, the Labanotation scores are generated, based on the information acquired in the previous steps. Figure 5 illustrates the processing chain for the generation of the scores.



Generation of skeletal representation
Quantization of poses
Search for key directions
Generation of (preliminary) Labanotation

Figure 5: The processing chain for the generation of Labanotation scores from the skeleton data.

Since all information is provided for each frame and the quantised pose information is stored into a buffer, the buffer size must be defined. If sequences that should be recognised are already provided, the buffer size is defined to be greater than the maximum length of the sequences. When exporting a Labanotation score, the length of the sequence must be defined. Since this cannot be determined by the engine, the buffer size can be set externally by calling the provided the movement controller method.

After the generation of Labanotation scores, recognition of registered sequences within the performed sequence can carried out. Each time a sequence is recognised, an event is triggered. Therefore, the observer pattern is applied and listeners (i.e., client applications) can be registered with the engine. Moreover, sockets are provided which cover network communication as well as inter-process communication (IPC).

Skeleton Representation

The torso basis used for the detection of poses (i.e., quantisation of direction and level) and key poses (i.e., quantisation of duration) is adapted from the skeleton representation of [16], which evolved from [17]. The torso basis is used for the orientation of the body in space in order to define and quantise the 27 different key directions used by Labanotation. Thus, the basis serves as a reference for the key directions and relative joint positions, which are used to get the direction and level for each frame.

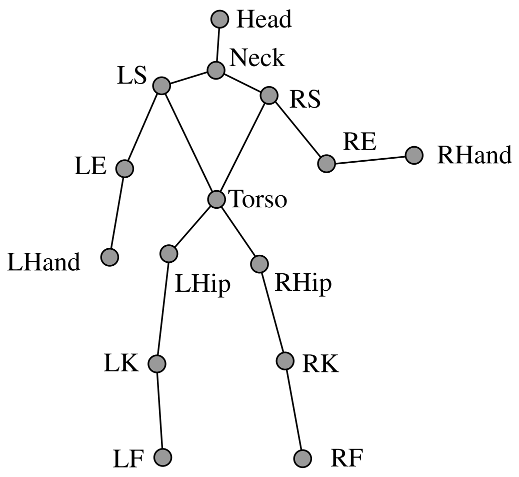


Figure 6: The 15 skeleton joints provided by the *NiTE* middleware. L is for left, R for right, S for shoulder, E for elbow, K for knee, and F for foot.

At first, the torso basis is constructed from the provided torso joints, i.e., torso given by the *OpenNI/NiTE* framework. Here, we follow the assumption of [17] that the torso is a rigid body since in most motions it is scarcely deformed. In order to extract the basis from the given joints, a Principal Components Analysis (PCA) is applied to the $n \times 3$ torso matrix (for n torso joints, in case of the NiTE skeleton $n = 6$) and two orthogonal main directions are obtained. A PCA is used to extract the principal components of a set of observations by sorting the eigenvectors of the matrix formed by these observations. This leads to calculating the main directions of a set of points. Although, PCA is not entirely accurate when not all torso joints lie on the same 3D plane, Raptis et al. [17] argued that the "basis is an exceptionally robust and reliable foundation for a coordinate system based upon the orientation of the human body". In terms of the torso basis, the PCA is applied in order to receive the two main directions of the torso, which are typically top-down and left-right. Therefore, the first direction denotes the vertical direction (vertical to the user), and the first eigenvector is used as the first component u of the basis. The second direction is orthogonal to the first one and denotes the horizontal direction (horizontal to the user). Therefore, the second eigenvector is used for the second component of the torso basis r. To have a fixed direction, top-down for vertical and right-left for horizontal, u and r are aligned in these directions. This is done by referring to the distance of the left shoulder joint to the right shoulder joint added to the r, distance decreases if the vector points to the wrong direction, and neck joint to torso joint added to u, distance increases if vector points to the correct direction. The third, and the final, component of the basis (t) is the cross product of the first two components:

$$t = u \times r. \tag{1}$$

Therefore, t always points from the torso to the front of the performer.

After a normalisation of the three components, the orthonormal torso basis is denoted by $\{u, r, t\}$ (see Figure 7).

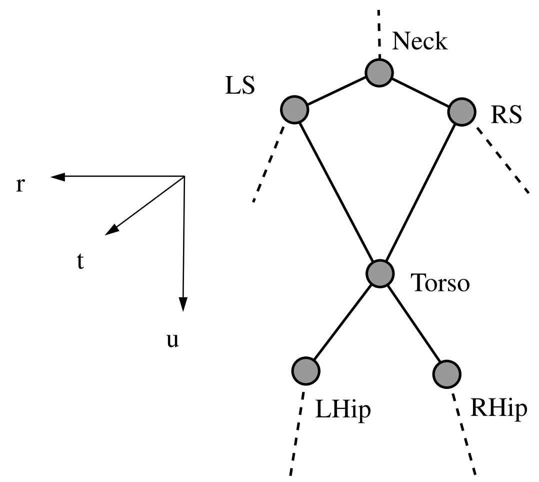


Figure 7: The orthonormal torso basis $\{u, r, t\}$ is generated through the torso joins, which are six for the NiTE skeleton. L is for left, R for right and S for shoulder.

The direction of weight plays an important role for Labanotation. When the user is in a standing position, it is equal to the top-down direction of the torso, denoted by u, and therefore this direction is used as default. When another direction is required for the body weight, the variable for the weight of the skeleton can be used. The weight is used instead of the top-down vector and the basis is modified in order to stay orthonormal when the weight is specified.

Quantisation of Direction and Level

The quantisation of direction and level is carried out by calculating the distance of the real value to the set value. Distances are given by the angular deviation in degrees. This is simple for vectors as the 27 set values are vectors provided by the torso basis, since Labanotation's direction and level are relative to the body, and these always point in the same direction when represented by the torso basis. The angle, between each bone and the set value vector, then denotes the distance between the real (v) and the set (s) value:

$$\beta = \text{angle}(v, s). \tag{2}$$

Note that β ranges from $0°$ to $180°$, given the opposite direction is furthest from the set value.

After the calculation of distances, the direction with the least distance is used as the quantisation of direction and level. While this works for all other directions and levels, it may be incorrect for the direction *place middle*. This is the case when other directions have smaller distances, but *place middle* was performed. Therefore, *place middle* is favoured when the distance is less than $22.5°$.

Additionally, a hysteresis may be applied which prevents the previously quantised direction and level from changes if the angle is not greater than the given hysteresis threshold.

Quantisation of Duration

In order to quantise movements, i.e., finding the start and the end of each movement, the quantised poses are processed. First of all, the key poses for each body part are estimated. Key poses define a change in direction and level and describe the first local minimum of the distance to the direction and level. To improve the accuracy, an interval in which the global minimum of the distance can be searched, is applied.

Starting of a motion is determined for every found key pose, for each of the body parts. Therefore, the first local maximum of the distance to the direction and level of the key pose is searched for. The distance to the present direction is also handled by using the same interval as for the key poses. If the beginning of a motion exceeds the previous key pose, the same is assumed to be the beginning of the motion.

Generation of Sequences

The generation of the Labanotation sequences is based on the classes and interfaces provided for this purpose. These are oriented towards the Labanotation XML schema, and provide a simple way to generate the desired sequences. First, a lookup for key poses, the beginning of the motions, quantisation of the starting time and duration is required. Next, the division into the measure and the beat is performed. This results in a Labanotation sequence, which is exported as an XML Labanotation score.

Recognition of Sequences

The last step of the algorithm, used by the Ambient Movement Analysis Engine, is the recognition of the sequences. This means that generated sequences from the skeleton stream are compared to previously registered sequences. Hence, registered sequences performed by the user can be detected. Similar to the recognition performed by the system described by [16], a decision forest is used. In their system, a decision forest is generated from the registered sequences using key poses. However, key poses are not present in Labanotation. The Ambient Movement Analysis Engine uses key movements per body part for the decision forest, which have a certain length and a position in the staff. As Labanotation movements are defined per body part, the decision forest is composed of subforests for each of these parts. Each subforest contains trees that carry information about the registered sequences for the body part it presents. The sequence of trees is in reversed order, which makes it easy to match these with the subsequent streaming sequence. Thus, the root movement of a tree is the last movement of a body part. Figure 8 illustrates column sequences, the same as movements of a specific body part, which are added to a decision forest for the column. A decision maker is used to decide whether two movement symbols match. This process takes in account the position and length of the sequence and responds to approximately equal items. A deviation up to a certain limit, which can be customised, is accepted. The default limit is 0.5 beats. The decision maker is used for insertion as well as recognition. Due to this, the same symbol, but of different length, may be the root element of different trees.

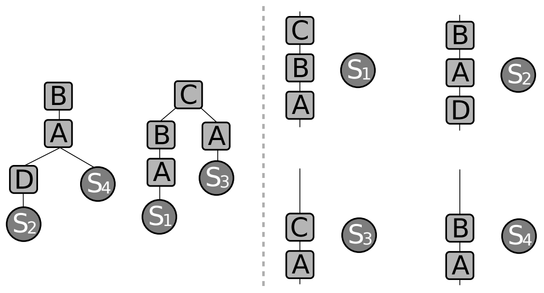


Figure 8: A small decision forest for column movements. Note that the different sequences, displayed on the right side, are written from bottom to top like Labanotation scores. The trees, displayed on the left side, contain these sequences in reverse order.

When a new sequence is to be added to the forest, some steps must be followed. The first step is the translation into a default format. This means that the same timing information and column definitions are used for all sequences in order to have a correct recognition at the end. This implies that the columns must be matched, and the positions and the durations of the movements must be adjusted.

The second step is applying rewriting rules due to the ambiguity of Labanotation. The sequence itself must be added to the forest, in addition to any sequence that is semantically equal. Therefore, all sequences, which may replace a sequence, are determined. This process is based on configured rewriting rules, which contain a sequence of elements and its possible replacement. Next, the lookup of occurrences of subsequences must be added to the forest. When a match is found the entire sequence is copied and the subsequence is replaced. The new sequence must then be checked for matches of rewriting rules as well. This finally results in all sequences that are semantically equal to the provided rewriting rules.

When it comes to the recognition of registered sequences in the entire streamed sequence, the entire sequence will be processed movement by movement. This processing starts at the end of the sequence, this means starting with the last performed movement. Matching decision trees for each body part are then searched. If a match is found, all matches of subsequences are stored and finally compared against the matches of other body parts (see Figure 9). All sequences, which have a match on every body part, are gestures that are performed by the user and thus an event will be provided informing on the recognition.

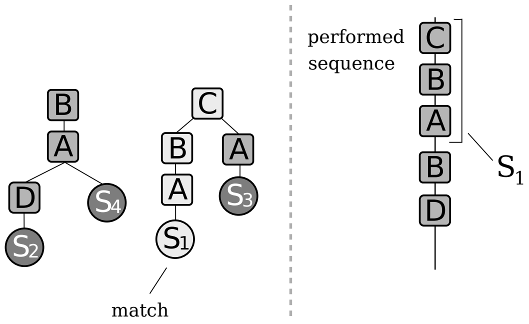


Figure 9: The recognition of a sequence performed by a user. The entire performed sequence is displayed on the right side and the recognised sequence in fact its subsequence. A decision forest, displayed on the left side, is used to check the matching sequence.

In order to find sequences that actually match the registered sequence, both the movement symbols in a column and the absolute position of the movements in the staff, must be checked. This means that the sequence cannot be checked per column only, resulting in a recognition when all columns match. Instead, the connection between the column movements must be considered, which also includes starting poses. Starting poses can be checked by taking the starting pose into the trees as well. When checking reaches the leaf of the tree, the distance to the next element of the performed sequence is checked, (the distance to the last pose is not checked at this stage. Thus, the check looks up whether there is no other movement than the defined in the column. In this case, the column of the streamed sequence must be empty as long as other body parts are still performing movements. The correctness of the positions of the elements can be checked by considering distances between a symbol and its predecessor, and by checking only the last movement of a sequence (which is the root element of the trees), for having the correct position within the staff. Therefore, the last element of each column sequence must be checked to have the correct distance to the last movement of the entire score.

RESULTS AND EVALUATION

We have performed a preliminary study to evaluate the feasibility of our novel approach based on the current implementation. We have hired twelve volunteers in a short recruitment period. The participants were between 23 and 59 years old from different educational and professional backgrounds (i.e., students, secretary, project administration, and research staff) and different levels of experience with gesture control (from none to experienced).

The participants were introduced to a set of 10 gestures and they were asked to perform each gesture 10 times in a realistic setup (a room with normal lighting and objects around). First, the users were introduced to the experiment's general purpose. Next, a step-by-step explanation and demonstration on how to perform each gesture were conducted by the instructor. Subsequently, the participants where given time for practicing the gestures and to be familiar with the experiment setup. Finally, the participants were asked to perform the gestures.

Our apparatus for the evaluation consisted of: the Asus Xtion PRO LIVE sensor for capturing the movement, a display for instructing participants, a videocamera for recording the experiment, and a PC for controlling the experiment procedure and recording the results.

In total, we have collected 1160 gesture recordings (about 120 recordings per gesture, some gesture were dropped for one participant due to shoulder problems).

Our gesture set contained 10 distinctive gestures that are easily understood and performed by the participants (illustrated in Figure 10):

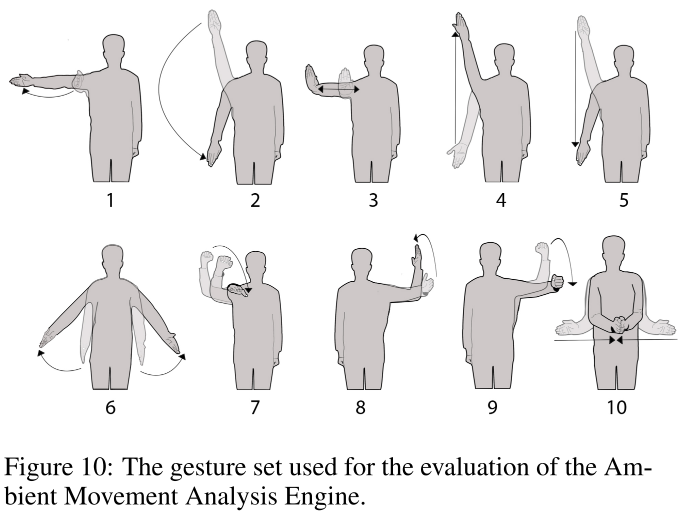


Figure 10: The gesture set used for the evaluation of the Ambient Movement Analysis Engine.

1. Swipe Moving the right arm horizontally fully stretched from front to the right side.

2. Rotate Perform a 180 degree clockwise movement with the right arm. The gesture starts with the arm fully stretched in front high position and ends with the arm in front low position.

3. Wave Horizontal waving with the right arm stretched in front.

4. Scroll Up (SU) Moving the right arm up. The gesture starts with the arm fully stretched in low position and ends with the arm in front high position. The hand inner surface should be kept facing up during the gesture.

5. Scroll Down (SD) Moving the right arm down. The gesture starts with the arm fully stretched in high position and ends with the arm in front low position. The hand inner surface should be kept facing down during the gesture.

6. Pinch To Zoom Moving both arms fully stretched away from each other starting in the lower front direction.

7. Throw Performing a throw with the right arm. The hand starts with in a grip states and ends fully stretched with the inner surface facing down.

8. Never Mind (NVM) Starting with the left upper arm to the left side and the forearm to the front. Next, moving the forearm up. The inner surface of the hand should be kept facing the body.

9. Hammer Opposite of NVM, but forming a fist with the left hand.

10. Clap Clap the both hands at once.

The results of our evaluation show that the engine is working with a reasonable recognition rate of 78% with strict gesture definition and 87% with a slight deviation. Table 1a illustrates the recognition rates achieved with one strict Labanotation score to describe the gesture (with different tolerance values for deviance in position and duration). Table 1b illustrates the recognition rates using two closely similar Labanotation scores to describe each gesture for tolerance in movement paths. False positive activations are registered with very low rate (3.4% and 2.2% for one and two definitions respectively). Those false activations are due to movements mostly caused by the performers moving after the end of gesture.

Table 1 illustrates the different tolerance levels and corresponding recognition rates. Tolerance levels denote how much deviation is accepted for duration and position of symbols in the sequence. The gestures in our set are defined for a 1 second in length (i.e., 5 beats). Our results show that for low tolerance levels, recognition rates are low. This is due to the actual time difference between the performed gesture by the participants to the defined gesture. Generally, gesture execution deviations from the defined Labanotation score should be allowed, because it is very difficult for participants to match the define time precisely. Of course, the timing gap between the defined and the performed gesture may certainly improve with repetition and increasing familiarity with the gesture. As seen in Table 1a, the low recognition rate of the wave gesture is due to the different sizes of our participants. The used depth sensor was easily confused between the level of the arm (facing straight or slightly down). The recognition rate is much improved by allowing some a deviation of the arm position as in Table 1b.

Table 1: The recognition rates of the engine. One beat is set to 200 ms and the tolerance denotes how much deviation is accepted for duration and position of symbols in the sequence.

(a) No deviations allowed (one strict Labanotation score)

Score	Tolerance in Beats						
	0.5	1	1.5	2	3	4	5
Clap	0.2	0.43	0.58	0.68	0.7	0.7	0.7
NVM	0.23	0.59	0.74	0.77	0.8	0.82	0.82
Hammer	0.19	0.53	0.76	0.8	0.8	0.8	0.8
Pinch	0.1	0.38	0.64	0.76	0.93	0.97	0.97
Rotate	0.05	0.17	0.28	0.39	0.5	0.59	0.6
SD	0.34	0.55	0.73	0.83	0.93	0.95	0.97
SU	0.06	0.25	0.67	0.83	0.95	0.95	0.96
Swipe	0.36	0.61	0.76	0.85	0.92	0.93	0.93
Throw	0.01	0.17	0.34	0.46	0.55	0.59	0.59
Wave	0	0.05	0.1	0.18	0.3	0.42	0.44
Total	0.15	0.37	0.56	0.65	0.74	0.77	0.78

(b) Deviation allowed (two closely similar scores)

Score	Tolerance in Beats						
	0.5	1	1.5	2	3	4	5
Clap	0.23	0.47	0.62	0.72	0.74	0.74	0.74
NVM	0.25	0.63	0.78	0.82	0.85	0.86	0.86
Hammer	0.19	0.53	0.76	0.81	0.82	0.82	0.82
Pinch	0.12	0.55	0.79	0.89	0.96	0.97	0.97
Rotate	0.07	0.23	0.35	0.49	0.62	0.72	0.73
SD	0.34	0.55	0.73	0.83	0.93	0.95	0.97
SU	0.11	0.41	0.83	0.95	0.96	0.96	0.96
Swipe	0.53	0.81	0.96	0.98	1	1	1
Throw	0.01	0.21	0.57	0.62	0.74	0.75	0.75
Wave	0	0.05	0.11	0.25	0.57	0.79	0.86
Total	0.18	0.44	0.65	0.73	0.82	0.86	0.87

Compared to previous studies in this domain, such as Hachimura et al. [9] and Chen et al. [5], our work is the first to report evaluation data and the first to use live sensor data streams for recording and analysing Labanotation movements scores. When comparing our approach to other gesture recognition systems, the Ambient Movement Analysis Engine clearly demonstrates promising results. Miranda et al. [16] achieved an overall recognition rate of 83.5% and up to 97.3% for one of three specific set of gestures tested in their evaluation using pose kernel learning and decision forests. A direct comparison with previous studies cannot be justified as our approaches are not similar and the tested gestures are not identical.

The efficiency of the system was evaluated by checking the required time to process the pose buffer for different buffer sizes. The evaluation was done on an i7 CPU M 640 @ 2.80GHz machine with 6GB RAM running Ubuntu 14.04 LTS. The results indicate that the time requirement (computational cost of at most 33 ms) is fulfilled on an average machine for a buffer size of up to 3600 frames (which corresponds to a two minute buffer).

We believe that the engine can be improved to achieve better results and provide more elaborated movement information. A hybrid approach using both, a high performance gesture recognition system (for recognition) and our algorithm for Labanotation motion score generation would provide high recognition accuracy as well as detailed motion descriptions. Another approach is to change our recognition approach from decision forests to a trained classifier using the Labanotation scores' elements for feature vectors and compare the performance with the current approach. Additionally, recognition rate can be improved with a more extensive Labanotation model covering more relevant information (our current implementation does not cover all structural movements such as turns).

Our current Ambient Movement Analysis Engine implementation is an extensible library that can be used for gesture control and provides the chance to extend or modify its behaviour. The engine provides simple access to the *OpenNI* driver but it can be extended for the operation with any other driver by referring to the skeleton as input. The generic engine can be configured to work with different skeletons (hierarchy defined) and the flexible Labanotation. A wider range of the Labanotation specification can be implemented by simply extending the controllers entrusted with the task of generating Labanotation. Besides plain gesture recognition and eventing, the engine provides to record detailed descriptions of the performed movements. These recordings provide standard human-readable movement scores based on Labanotation, which can be an essential part of motion-based interaction documentation. Two context delivery options were implemented, namely the observer pattern and client-server communication, for eventing for further processing. Both options are accessible via a dedicated application programming interface (API).

CONCLUSION

Within the scope of this work, the Ambient Movement Analysis Engine for recording and analysing live off-shelf motion capture data streams in Labanotation movement scores was designed, implemented and evaluated. To the best of our knowledge, our approach is the first to provide this type of recognition and generation of context events. The engine addresses the growing need for simple ways to create, customise and handle user interactions, especially through demonstration and declaration. The engine dramatically reduces the complexity of programming, and customisation of such systems for end-user programming and non-professional interaction developers. In addition to a detailed description of the engine and our recognition approach, we have provided a preliminary evaluation of the engine with a reasonable recognition rate of 78%, with strict gesture definition, and up to 87% with a slight deviation for recognising gestures modelled in Labanotation from the live data streams of an off-shelf budget depth sensor.

FUTURE WORK

The Ambient Movement Analysis Engine can be improved and extended to cover more Labanotation, such as detecting steps, turns and body weight. Moreover, another aspect that could potentially be implemented is a feedback loop for the skeleton data based on the generated Labanotation score. Besides the improvement of skeleton data, a visualiser for missing movements of a defined sequence would be useful. Further evaluations could be carried out with multiple depth sensors and larger test groups. Despite the successful and wide usage of Labanotation to model movement amongst non-technical users, for instance dancers, artists and choreographers [14], we believe that an evaluation of the end-user programming aspects using Labanotation for modelling gestures, is a very relevant theme. Hence, an evaluation study in this direction is currently being planned as an own contribution to be followed.

ACKNOWLEDGEMENT

This work is partially sponsored by the The German Federal Ministry of Education and Research (BMBF) funded project Ensembles (Code - 16SV6369).

REFERENCES

1. Altakrouri, B. *Ambient assisted living with dynamic interaction ensembles*. PhD thesis, Universität zu Lübeck, Institut für Telematik, June 2014.
2. Altakrouri, B., Gröschner, J., and Schrader, A. Documenting natural interactions. In *CHI '13 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '13, ACM (New York, NY, USA, 2013), 1173–1178.
3. Altakrouri, B., and Schrader, A. Describing movements for motion gestures. In *1st International Workshop on Engineering Gestures for Multimodal Interfaces (EGMI 2014) at the sixth ACM SIGCHI Symposium on Engineering Interactive Computing Systems (EICS'14)* (Rome, Italy, June 2014).
4. Bishop, C. M. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.
5. Chen, H., Miao, Z., Zhu, F., Zhang, G., and Li, S. Generating labanotation from motion capture data. In *Proceedings of the 2013 International Conference on Culture and Computing*, CULTURECOMPUTING '13, IEEE Computer Society (Washington, DC, USA, 2013), 222–223.
6. Fanello, S., Keskin, C., Izadi, S., Kohli, P., Kim, D., Sweeney, D., Criminisi, A., Shotton, J., Kang, S. B., and Paek, T. Learning to be a depth camera for close-range human capture and interaction. In *Journal ACM Transactions on Graphics (TOG)*, ACM – Association for Computing Machinery (July 2014).
7. Fogtmann, M. H., Fritsch, J., and Kortbek, K. J. Kinesthetic interaction: revealing the bodily potential in interaction design. In *Proceedings of the 20th Australasian Conference on Computer-Human Interaction: Designing for Habitus and Habitat*, OZCHI '08, ACM (New York, NY, USA, 2008), 89–96.
8. Gamma, E., Helm, R., Johnson, R., and Vlissides, J. *Design Patterns: Elements of Reusable Object-oriented Software*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1995.
9. Hachimura, K., and Nakamura, M. Method of generating coded description of human body motion from motion-captured data. In *Robot and Human Interactive Communication, 2001. Proceedings. 10th IEEE International Workshop on* (2001), 122–127.
10. Hutchinson Guest, A. *Labanotation : the system of analyzing and recording movement*. Routledge, New York, 2005.
11. Kahol, K., Tripathi, P., and Panchanathan, S. Documenting motion sequences with a personalized annotation system. *IEEE MultiMedia 13*, 1 (Jan. 2006), 37–45.
12. Konečný, J., and Hagara, M. One-shot-learning gesture recognition using hog-hof features. *J. Mach. Learn. Res. 15*, 1 (Jan. 2014), 2513–2532.
13. Laban, R., and Lawrence, F. *Effort*. MacDonald and Evans, London, 1947.
14. Lepczyk, B. Labanotation revisited. *Journal of Physical Education, Recreation & Dance 82*, 6 (2011), 5–6.
15. Loke, L., Larssen, A. T., and Robertson, T. Labanotation for design of movement-based interaction. In *Proceedings of the Second Australasian Conference on Interactive Entertainment*, IE '05, Creativity & Cognition Studios Press (Sydney, Australia, Australia, 2005), 113–120.
16. Miranda, L., Vieira, T., Martinez, D., Lewiner, T., Vieira, A. W., and Campos, M. F. M. Online gesture recognition from pose kernel learning and decision forests. *Pattern Recognition Letters 39* (april 2014), 65–73.
17. Raptis, M., Kirovski, D., and Hoppe, H. Real-time classification of dance gestures from skeleton animation. In *Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA 2011, ACM (New York, NY, USA, 2011), 147–156.
18. Yu, T., Shen, X., Li, Q., and Geng, W. Motion retrieval based on movement notation language. *Computer Animation and Virtual Worlds 16*, 3-4 (2005), 273–282.
19. Zacharatos, H., Gatzoulis, C., Chrysanthou, Y., and Aristidou, A. Emotion recognition for exergames using laban movement analysis. In *Proceedings of Motion on Games*, MIG '13, ACM (New York, NY, USA, 2013), 39:61–39:66.