A Novel Approach to Diagnosing Motor Skills

Aitor Aguirre, Alberto Lozano-Rodero, Luis M. Matey, *Member, IEEE*, Mikel Villamañe, and Begoña Ferrero

Abstract—The combination of virtual reality interactive systems and educational technologies have been used in the training of procedural tasks, but there is a lack of research with regard to providing specific assistance for acquiring motor skills. In this paper we present a novel approach to evaluating motor skills with an interactive intelligent learning system based on the ULISES framework. We describe the implementation of the different layers that ULISES is composed of in order to generate a diagnosis of trainees' motor skills. This diagnostic process takes into account the following characteristics of movement: coordination, poses, movement trajectories and the procedure followed in a sequence of movements. In order to validate our work we generated a model for the diagnosis of tennis-related motor skills and we conducted an experiment in which we interpreted and diagnosed tennis serves of several subjects and which shows promising results.

Index Terms—Procedural tasks, motor skills, virtual reality, interactive learning environments

1 Introduction

Virtual reality interactive systems (VRIS) have been broadly used to enhance the learning process of procedural tasks. Sometimes they are even combined with adaptive educational technologies such as those from intelligent tutoring systems (ITSs) in order to provide trainees with the benefits of real one-on-one instruction within the simulation. This family of systems ranges from web delivered 3D environments with limited interaction capabilities to fully immersive VR environments where interaction is highly realistic. They can cover a broad spectrum of educational functionalities like adapted real-time feedback, instructional guidance, and recommendations. These systems, henceforth referred to as intelligent interactive learning systems (IILS), can correct trainees while they perform procedural tasks (e.g. STEVE ([1]), CanadarmTutor [2], [3]). However, it is difficult to find IILSs that provide specific assistance for acquiring motor skills. For example, an IILS for training ballet dancers should give advice on how to improve poses and the coordination of movements, in addition to reminding dancers of step sequences. This paper aims to endow IILSs with capabilities for diagnosing motor skills, which is the basis for providing learners with intelligent assistance. Therefore, our work is focused on IILSs equipped with a motion capture system, which can capture learners' movements so they can be automatically evaluated.

The main problem underlying this objective has to do with extracting the semantics of trainees' movements in the context of the procedural task; that is, an IILS has to recognize the actions and the meaning those actions have within the learning domain. Therefore, capturing and recognizing human actions is the first important step that has to be considered. Different kinds of motion capture systems, (MoCap) such as camera based (whether marker-based systems or markerless systems), inertial and magnetic systems, etc. [4], can be suitable for training motor skills. After the movement is captured, the action recognition process is carried out. Normally, action recognition systems follow three principal phases [5]: feature extraction, action segmentation and action learning and classification. However, depending on the features of the actions that need to be recognized (the punctual or short actions, periodic actions, and composition of simpler actions), different action recognition techniques are used. In fact, most of the existing action recognition techniques depend on the properties of the domain where the actions are being carried out, which means that if the type of captured movements is modified, the performance of the statistical methods used for learning and classification may worsen considerably. Given these characteristics and the results of our previous work [6], we believe that a classic action recognition procedure is not sufficient for defining a motor skills representation that is suitable for generating an accurate diagnosis in an IILS.

There are certain features that are common to any type of movement and that serve as a basis for generating a correct diagnosis. The first feature is the temporal relation between movements. As Allen [7] stated, actions cannot be considered punctual events; they are time intervals. What is more, different movements do not occur in a predictable manner, so qualitative temporal relations between movements have to be taken into account. There are some studies in the literature that refer to temporal relations between actions in activity recognition techniques [8], but it is not a common approach. Intentionality is another important characteristic of the movements. When trainees are learning or improving a movement, they might be trying to make one movement when in fact they are making a different one. In addition to the common features of a movement, it is also important to obtain a representation of the movement that is sufficient for making a successful diagnosis. For example, if we want to diagnose how a trainee runs, recognizing the action of

- *A. Aguirre and L.M. Matey are with CEIT and Tecnun, University of Navarra, 20018 San Sebastian, Spain. E-mail: {aaguirre, lmatey}@ceit.es.*
- *A. Lozano-Rodero is with CEIT, Manuel Lardizabal 15, 20018 San Sebastian, Spain. E-mail: alozano@ceit.es.*
- *M. Villamañe and B. Ferrero are with the Department of Computer Languages and Systems, University of the Basque Country UPV/EHU, San Sebastian, Spain. E-mail: {mikel.v, bego.ferrero}@ehu.es.*

Manuscript received 13 Nov. 2013; revised 18 June 2014; accepted 19 June 2014. Date of publication 23 July 2014; date of current version 15 Dec. 2014.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TLT.2014.2340878

running is not enough. The running movement has to be divided into various phases: footstrike, mid-stance, propulsion phase and swing phase. In this way, specific errors related to each phase can be identified in order to suggest corrections just as a real tutor would, for instance, via messages such as: “you have to lift your left knee more”, “your footstrike does not begin with the ball of your foot”, etc. In order to generate these messages, it is necessary to characterize each phase of the movement and the relations between them. This characterization could be achieved by parameterizing the movement phases that are relevant for the diagnosis with features such as duration, acceleration, speed, joint angles, direction of the joint trajectories, etc. Lastly, action diagnosis requires discerning the context in which the actions are happening. Continuing with the running example, lifting the knee during acceleration is not the same as during deceleration, because the meaning of lifting the knee is in a different context. Thus, context modeling must also be included. A generic movement model that considers these issues would allow us to obtain the actions’ semantics in order to diagnose motor skills in different learning domains.

In this paper, we first present some relevant related studies and an overview of our previous work, which is aimed at methodologically developing IILSs for different procedural domains and is therefore essential to understanding this work. Later we thoroughly describe the methodology for including the diagnosis of motor skills in an IILS. Next we demonstrate the operation of this methodology via the diagnosis of a tennis serve and the validation of the diagnosis. Then we provide a general discussion of the work and finally conclusions are drawn.

2 Related Work

A procedural task can be defined as one whose learning requires the integration of two types of human capabilities: intellectual skills and motor skills [9]. A machine maintenance task or making a tennis serve are examples of procedural tasks, but their cognitive and psychomotor demands differ. A machine maintenance task may demand great intellectual skills but lower psychomotor capabilities relative to a tennis serve. A machine maintenance task requires spatial perception capabilities in order to identify different parts of the machine and navigational capabilities in order to navigate during a maintenance task. It must be noted that a machine assembly task may require greater psychomotor capability than a simple monitoring task (e.g. watching the pressure level of a turbine). However, a tennis serve requires a much more precise quality of movement. What is more, if we compare it, for example, with javelin throwing, a tennis serve requires more cognitive skills given that javelin throwers have no opponent and their main challenge is moving efficiently in order to throw the javelin as far as possible.

In this regard, the higher the psychomotor demand of a task is, the less assistance current IILSs are able to provide to the learners when they train a certain task. In fact, there are few IILSs that provide specific assistance in training motor skills in comparison to the assistance given in other IILSs to train procedural tasks.

In this section, we will address the IILSs that assist trainees in procedural tasks that require high psychomotor capabilities. In this regard, three main domains can be discerned: the machinery manipulation domain, the surgical domain and the physical activity domain. Concerning the domain of machinery manipulation, to our knowledge there are two main works of considerable research interest: CanadarmTutor [10] and Tervo’s work [3].

CanadarmTutor is a simulation-based ITS whose aim is to train operators to operate the Canadarm2, a robotic arm. Its principal objective is to teach how to move this arm from one point to another. When carrying out this movement, there are many possible ways to move the arm and many strategies for doing it, and for that reason it is considered an ill-defined task. Nevertheless, the ill-definedness of a domain is an issue that is still under debate ([11], [12]), and it is not debated in this paper. When operators are executing a trajectory, there are many ways to move the arm: they have to avoid collisions with objects, there may exist blind spots where users will not see the arm, and singularities have to be avoided. What is more, the experience of the operators in carrying out the different movements that they can execute has to be taken into account. In order to overcome the problems related to the ill-definedness of operating the robotic arm, Fournier-Viger et al. proposed a multi-paradigm approach [12]. They use a cognitive model to cover well-defined parts of the task and spatial reasoning and a data mining approach for automatically generating a partial task model from user solutions for ill-defined parts of the task. In the case where the ITS cannot offer help via these two methods, a plan is generated with the path-planner. This multi-paradigm approach allows the different paradigms to be taken advantage of and some issues related to ill-definedness, such as the complexity of modeling the domain, are overcome. In addition, the system is able to offer tutoring services such as evaluating spatial reasoning, assessing the learner’s profile and proposing solutions to the learner. With regard to motor-skills-related assistance, the characteristic that is taken into account is the sequence of actions that the learners have to follow.

Tervo et al. proposed a different approach. In [3] they presented a general framework for the evaluation of machine operators’ skills based on task sequence recognition using a Hidden Markov Model (HMM). The work is evaluated in partly automated mobile working machines (a harvester). In order to evaluate operators’ skills, in this work four principal skill metrics are defined: task efficiency, task sequence complexity, the ability to plan and make decisions and task complexity. They present a methodology that evaluates operators based on the input signals from the controls buttons and levers. In order to undertake this evaluation, they propose an HMM with explicitly defined states. The role of the HMM is to recognize the intentionts of the operator based on the resulting actions, such as pushing buttons or moving the lever. HMM has been used in previous works ([13], [14]) to represent different skill levels. This kind of statistical model usually compares expert skill representation and learners activity. The results of these methods are the percentage of similarity between expert and learner activity. However, HMMs do not offer evidence about the cause of errors or incorrect performance, and therefore it is impossible to know why learners do not achieve the desired skill level. What is more, if the task that is going to be defined is ill-defined, the cost of defining the models can become very high. In light of this problem, Tervo et al. divide a task into different subtasks, where each HMM state corresponds to an operational phase of the task (e.g. engaging the clutch). The HMM stores the sequence state, which refers to the sequence of individual tasks that is needed to achieve the goal. That is to say, first the action that is probably occurring is detected, and then the most probable sequence of steps that can be happening is inferred. The principal drawback of this approach is that it requires that the task be repetitive and also each task has to be composed of a small number of individual tasks. If the number of individual tasks is too large, the capability of the classifier to generalize is reduced considerably [15]. What is more, if the sequence of tasks is large or if there are many possible ways of solving the sequence, the use of HMM for this purpose is not practical. Hence, it can be concluded that this approach is not well suited for ill-defined tasks.

The problems that have been addressed in the previous works can be moved to the surgical domain. Usually, the parameters that evaluate future surgeons' motor skills are not evident, so the tendency is to develop ad-hoc solutions for the required surgical operation. However, there are some systems that carry out a low level analysis of positions, forces and times recorded during training simulations ([16], [17], [18], [19]). Murphy et al. carried out a more exhaustive evaluation of dynamic tasks in the surgical domain using HMM classifiers [20]. They train each HMM independently for each movement based on the gestures extracted from the dynamic state of the system, and those movements compose an HMM network. At each moment, the most significant movement is recognized by evaluating the probability of the gesture sequence for each movement model. Skill evaluation is done based on the total number of movements and the time percentage used for each movement during the task. Rosen et al. ([21], [22]) describe a similar approach. They model the task using sequences of subtasks that are modeled via HMM. Forces and torques that are applied in the surgical tools are measured in 3D, and the HMM states are defined so they coincide with the tasks the surgeon executes during an operation. The evaluation of skills is based on the statistical distance to the expert mode, taking into account task execution times, the frequency of the executed tasks and the measurements of forces and tasks.

In comparison to the abovementioned works, there are other IILSs that are more focused on the physical part of the procedural task. As cited previously, there are plenty of IILSs that evaluate trainees' actions based on action recognition techniques, but there are some works that go further. reactive virtual trainer (RVT) [23] is an interactive virtual trainer capable of representing physical activity carried out by a human, monitoring trainees' activity and providing feedback at different levels; it is mostly focused on the psychological aspects of the training. The system is designed for repetitive tasks such as fitness or psychotherapy, and the avatar´s movements are modeled by defining three parameters: tempo (rhythm), the amplitude of the movement, and effort, which is measured with the acceleration of the movement. Movements are modeled by defining their trajectory in different ways. It is possible to model a trajectory by defining initial, intermediate and final poses or using a Hermite spline. However, the information about the trajectory is not used when correcting trainees' movements: they take into account the tempo and trainees' synchronization with respect to the avatar. There are other IILSs that are specific to training dance movements. In some of them, trainees learn by imitating the movements of a virtual agent [24], [25], [26]. Other IILSs make a deeper analysis of trainees' movement by using a motion-matching technique. The objective of this technique is to compute the similarities between motion templates and the input motion. In this regard, Hachimura [27] proposed the use of Laban movement analysis (LMA) for the analysis of the movements. This analysis includes weight (kinetic energy), space (direction), time (acceleration) and shape (overall shape) to analyze and evaluate dancing movements. These characteristics are extracted when movements are compared, and the differences between the movements can be computed anytime. Nevertheless, this method is not suited for detecting local errors; it just matches the postures globally. Other authors use the angles of different joints in order to match movements, which allows local or specific errors to be detected. For example, Qian et al. [28] and Kwon and Gross [29] carry out motion matching by measuring the statistical distance between two sets of joint angles. Chang and Leung [30] created an IILS that is capable of correcting trainees postures and synchronization by showing two different avatars to trainees. One of them shows trainees' movements, and depending on the correction to the movement, body-parts are shown in different colors. At the same time, the other avatar demonstrates of the correct movements to the learners. In this case, the degree of correction of the postures is calculated using the euclidean distance between the real joints' positions and the posture of the template.

To conclude, it is fair to say that there are not many IILSs that provide intelligent assistance to trainees when they train motor skills. However, there are many IILSs that are centered mostly on the intellectual part of the task, and we believe that the paradigms that are employed in those systems can be applied in the process of tutoring motor skills, too. With regard to the IILSs that are focused on the training of physical skills, the bases of the movement analysis are established. However, there is a lack of methodologies and proper data manipulation methods to provide the automatic evaluation of motor skills that could reveal learners' skill level. Generally, existing methods compare predefined templates and learner movements, giving as a result the percentage of similarity between the movements. However, we believe that proper motor skill evaluation should provide information about the causes of the errors made by trainees when they are learning a motor skill.

3 Previous Work: OLYMPUS and ULISES

OLYMPUS [31] is a generic platform for the creation of IILSs. As noted above, the objective of an IILS is to assist

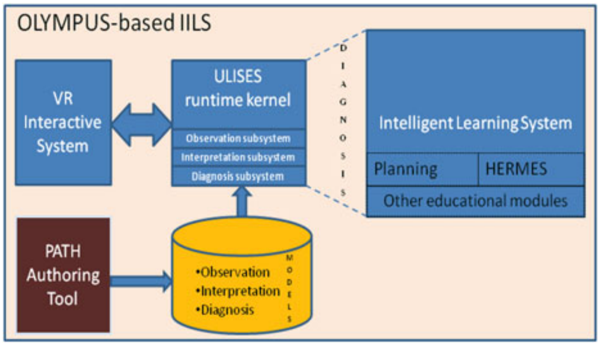

Fig. 1. The role of ULISES in the OLYMPUS platform.

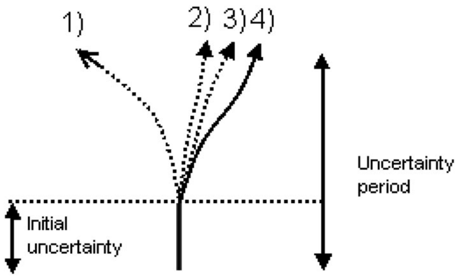

Fig. 2. The four joint trajectories start following the same direction, but it is not until the end when the real movement is discerned. Trajectory 4 represents the movement that a trainee intended to execute, while trajectories 2 and 3 are similar movements that a trainee could make.

trainees when they are learning a certain task, which can be achieved by integrating a VRIS with an ILS (e.g. [32]). To achieve this integration, OLYMPUS includes a framework called ULISES [33]. A VRIS generates real-time outputs or data streams that do not have educational meaning on their own, so it is difficult for an ILS to work directly with these outputs. The ULISES framework was created to solve this problem. This platform transforms data streams generated by a VRIS into educationally suitable data for an ILS. Fig. 1 shows the different modules and applications that compose the OLYMPUS platform.

1) *VR interactive system*. This is the Virtual Reality—Mixed Reality system where the trainees practice tasks. If the task to be learned involves motor skills, this is the place where the MoCap system is located.
2) *ULISES runtime kernel*. ULISES runtime kernel is the core of the ULISES platform. It sniffs data streams coming from the VRIS in real time and provides diagnostic results for the ILS. This diagnosis can be used, for example, to send appropriate feedback to the trainee.
3) *ILS*. This is the set of modules that gathers the data generated by ULISES in the form of diagnosis results and provides educational functionalities in the IILS. The diagnosis results can be input for a wide range of educational functionalities: SCORM compliant learning management systems, report generators, real-time feedback generation, etc.

The generic condition of ULISES is based on the domain-independence of its three levels: observation, interpretation and diagnosis. The observation level contains methods for fusing events and data streams and modeling their meaning. The domain-independent O**bservation** element is the result of this analysis, which represents a fact that has been observed over time. The Interpretation and Diagnosis levels model learner activity using **Step** and **Situation** elements, which are created from observations. These three elements are independent from the VRIS and allow any procedural activity to be represented, including tasks related to motor skills.

4 Methodology

This section describes how ULISES addresses the special characteristics of motor skills. The main objective of ULISES with regard to motor skills is to transform movement into language variables [34]. When experts are going to correct a trainee, they use qualitative descriptions [35] such as: "you have to extend your hips more", "you have not rotated your trunk fast enough", etc. In trying to imitate this behavior, an IILS has to be approached from a qualitative point of view, which means that the system has to be able to transform quantitative movement variables to a qualitative domain. In order to do that, the movement has to be decomposed into parts [36] and represented in an intuitive way. What is more, the way a movement is represented has to take into account another characteristic: *uncertainty*. When a trainee starts making a movement, the tutor does not know which movement the trainee is making until a change in the movement direction occurs or an interval of time passes. For example, if the trainee starts moving his hand upwards, the tutor will not know if he is trying to move diagonally (left or right) or straight (Fig. 2). Additionally, when a trainee makes a movement, the tutor's brain reasons in a fuzzy way that there are no established rigid boundaries between a movement that is fast or slow or between one that is diagonal or vertical. Therefore, *fuzziness* must be considered as well. Moreover, *intentionality* has to be added to uncertainty. When trainees are learning a new movement, they will not perform the exact intended movement but instead it will be an imperfect movement (Fig. 2, joint trajectories 2, 3 and 4), which adds a handicap to interpreting trainees' actions automatically. Taking all these into account, we decided to represent the movements using *arcs* complemented with *fuzzy logic*.

This approach assumes that relevant body landmarks and joints can be tracked with a MoCap system. Therefore, the movement of each landmark forms a *trajectory* that can be segmented in smaller parts. In our approach we express each segment of a trajectory as an ***arc***, which stands for *"a part of a circumference of a circle or other curve"*. The decision to choose arcs was taken based on an educational point of view. We believe that each part of the movement has to be represented in a simple manner, but at the same time it has to contain enough information such that semantically meaningful data can be extracted from it. Using arcs represent a movement segment allows it to be accurately represented without losing semantic information. An arc allows the direction, length, speed, acceleration to be described and it permits the degree of the arc (or curvature) of the trajectory to be discerned. With these variables, it is possible to correct a movement using verbalizable elements. What is more, the description of an arc can be done using a few variables. In

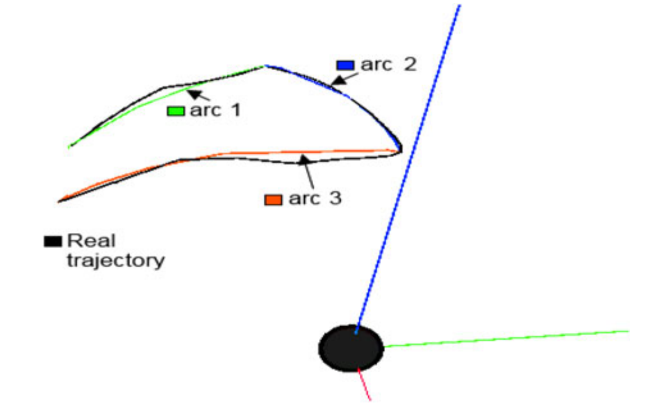

Fig. 3. Black lines represent the real movement, while the color lines represent the approximation of each movement described with an arc.

comparison to other representations of trajectories (such as splines), arcs allow movements to be defined in a sufficiently simple way, making verbal corrections possible.

Segmenting movement into simpler parts is also important in this context. We have pointed out that a way of recognizing a new movement is by the change in direction of a joint's trajectory. Therefore, the trajectory of a point during a movement is segmented in every possible change of direction (depending on the precision that is needed). As each segment is expressed as an arc, the result will derive a set of arcs (Fig. 3). Moreover, each arc will be temporally related to the rest of the arcs, including other joints' arcs. In short, we define a movement as a *set of arcs that are interrelated by constraints*.

Furthermore, an arc's direction can be represented with fuzzy logic in order to obtain a qualitative description of the movement. In the following sections, we detail how this approach has been implemented within ULISES's observation, interpretation and diagnosis levels.

4.1 Observation Level

In terms of representing a movement, the first step in the observation level is to segment the movement of a single tracked joint in order to approximate the trajectory as a sequence of arcs. As already noted, the segmentation is done based on two criteria: a change in direction (x-, y-, or z-axis) or the elapsed time since the end of the last segmented arc. The change of direction is detected by finding the local maxima and minima in real time. A threshold for defining "significant" changes in direction is calibrated depending on the domain in order to adjust accuracy as needed. In addition, if the MoCap system is noisy, a smoothing filter is used in order to avoid false direction changes. Segmentation is also done based on elapsed time. Elapsed time is useful for detecting poses and slowly executed movements. When a certain amount of time passes, there is no need to wait for a direction change to know which movement or pose is being performed. In this work, we have determined empirically that a value of 1,500 milliseconds is usually adequate, although this parameter can also be configured as needed.

Once the segmentation is done, the segment is classified according to its movement direction. The movement space is discretized into 26 classes that are separated by 45 degree

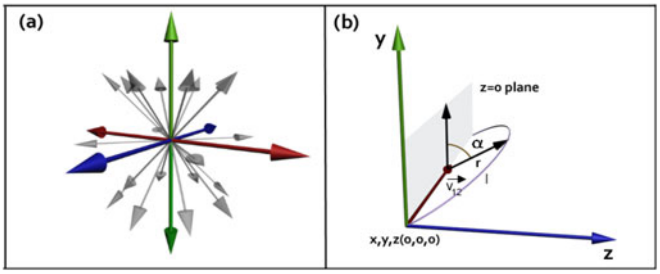

Fig. 4. (a) The 26 classes that are used to represent movement. (b) A vector v_{12}that represents the direction of movement.

(Fig. 4a). This discretization is similar to the one used in Labanotation [37]

$$\begin{aligned} &\mathrm{Classes} = \{\mathrm{P} = (\mathrm{i},\mathrm{j},\mathrm{k}) \,|\, \mathrm{P} \neq (0,0,0) \quad \forall \mathrm{i},\mathrm{j},\mathrm{k} \in \mathrm{I}\} \\ &I = \{1, 0, -1\}. \end{aligned} \tag{1}$$

In order to handle the uncertainty and the imprecision of a movement, we use a fuzzy approach. In other words, the movement is classified into one class but the similarity (distance) to the rest of the classes is stored as a special property of the observation. The first step in the classification process is the calculation of the angle (β) between the movement vector (Fig. 4b) (local coordinate system based on the hip center) and each of the classes in Fig. 4a (2).

$$\beta = \cos^{-1}(\overrightarrow{v12}x\overrightarrow{vClass}. \tag{2}$$

Then, the degree of membership in each class is calculated in order to obtain the fuzzy representation of the movement, which is calculated based on (3):

$$f(class) = \begin{cases} 0, & \beta \geq \dfrac{\pi}{4} \\ \dfrac{\frac{\pi}{4} - \beta}{\frac{\pi}{4}}, & 0 \leq \beta < \dfrac{\pi}{4} \end{cases} \tag{3}$$

Once the segment is classified into a specific class, it is transformed into an arc, which is expressed as:

$$Arc(\alpha, r, l). \tag{4}$$

An arc's parameters are calculated in a local coordinate system which is obtained as follows: the origin of the coordinate system is placed at the starting point of the movement. The vector that defines the movement (v_{12}) represents the x-axis of the coordinate system. Then, the y-axis is obtained based on the projection of the x-axis in the global coordinate's ground plane. Similarly, all the movements are defined using a static reference as a base. Based on this coordinate system (Fig. 4b), α expresses the angle between the $z = 0$ plane and the movement. The radius and the length of the arc are expressed with r and l. These three elements are a sufficient number of arguments to define an arc and to define a movement in a simplistic model that can later be expressed in words. In case two or more similar adjoining segments belong to the same class, they are treated as a unique arc, so its properties are recalculated based on the new arc formed by the adjoining segments.

Therefore, the observations generated at this level are represented as an interval of time and defined with a fuzzy function and by an arc, which is parameterized by an angle (α), radius (r) and length (l). Observations are generated in real time, and their attributes are *updated in each simulation cycle* while the data is being gathered from the VRIS:

- *Start*. Initial instant of the observation.
- *End*. Final instant of the observation.
- *OpenInterval*. This indicates whether the observation is finished or not at each simulation cycle.
- *ConfirmedUntil*. While the interval is opened, this attribute indicates that from a Start instant to a ConfirmedUntil instant there is no uncertainty. From ConfirmedUntil to End, the interval is uncertain.
- *Duration*. This indicates the duration of the observation at each simulation cycle. When the interval is opened, more data coming from the VRIS could lengthen the duration of the interval.
- *FuzzyList*. This contains the list with the degree of membership in each class (the 26 types of movements).
- *Properties*. This is the list of properties defined in the observation model. Arc is a compulsory property, while others like Speed and Acceleration are optional.
    - *Arc*. This property stores the representation of the trajectory as an arc (angle, radius and length).
    - *Speed*. Movement speed at each simulation cycle.
    - *Acceleration*. Movement acceleration at each simulation cycle.
    - ...

Due to the segmentation algorithm, sometimes it cannot be guaranteed that an observation is or is not happening; that is, there is uncertainty in some simulation cycles. Because of that, the interpretation subsystem is notified of all the observations that are being generated in each cycle, whether they are confirmed until the end or not (under uncertainty). Then, the interpretation subsystem takes this uncertainty into account in order to interpret what actions the trainee is actually performing. The interpretation level of the ULISES metamodel describes the necessary elements to achieve this aim.

4.2 Interpretation Level

The interpretation subsystem (or interpreter) decides which steps and situations are being carried out based on the observations generated by the observation subsystem. These two elements are modeled in the interpretation level using constraints: when constraints are satisfied, steps and situations are "detected". Steps and situations are modeled with three sets of constraints: *general*, *start* and *end*. Firstly, general constraints must be satisfied in order to interpret that a step or situation is happening. Start constraints indicate that the step or situation is going to begin, while the satisfaction of end constraints indicates that a step or a situation is going to finish, regardless of the general constraints. Usually, only general constraints are necessary. The other two are used when the start or end are slightly different from the rest of the step in order to facilitate the interpretation. The interpretation level specifies the different types of constraints that can be combined to model steps and situations. Constraints adjust the acceptable range of values that properties can have so a step is interpreted without being identical to the expert's solution:

- *Temporal qualitative constraints*. These are used for defining temporal relations between time intervals (observations, steps or situations) without needing to numerically quantify the instants of starting, ending or the duration of the intervals [38]. The following example represents that the Footstrike step is carried out before the Propulsion step:

$$Footstrike\ [Precedes]\ Propulsion$$

- *Punctual temporal qualitative constraints*. These define qualitative temporal relations between the starting or ending instants of two intervals. In this example, the Footstrike step ends before Propulsion starts:

$$Footstrike.End < Propulsion.Start$$

- *Temporal quantitative constraints*. These constraints are used to establish conditions over the duration of one interval or over the time elapsed between two intervals. The following is an example of this type of constraint, where the time elapsed between the start of the Footstrike and Propulsion steps has to be less than 1,000 milliseconds:

$$Duration(Footstrike.Start, Propulsion.Start) < 1{,}000\ ms$$

- *Constraints over properties*. Such constraints establish conditions over the values of observation properties during an interval. Arcs and fuzzy classes are also properties of an observation, so constraints are also used to establish conditions over them. The first example shows two constraints applied over the observation named RAnkleDirection010, which is observed when the right ankle moves up. The first constraint (Arc) constrains concavity, curvature and length of the trajectory. The second one (Fuzzy) constrains movement direction, indicating how similar the movement can be to neighboring classes (represented with one Logic Constraint per class that constrains similarity with percentage thresholds). The second example represents a constraint over the movement velocity:
    - $RAnkleDirection010.Arc(>90, <0.3, <\ 0.5).$ $Fuzzy(>=0.0, >=0.0, >=0.0, >=0.0, >=0.0, >=0.0, >=0.0, >=0.0, >=0.0, >=0.0, >=0.0, >=0.0, >=0.0, >=0.0, >=0.0, >=0.0, >=0.0, >=0.0, >=0., >=0.0, >=0.0, >=0.0, >=0.0, >=0.0\ AND <100.0, >\ =50.0)$
    - *RAnkleDirection010.Velocity* > 1.0
- *Logical constraints*. They are used to establish logical relations between intervals by means of the operators AND, OR and NOT. Of course, observations not

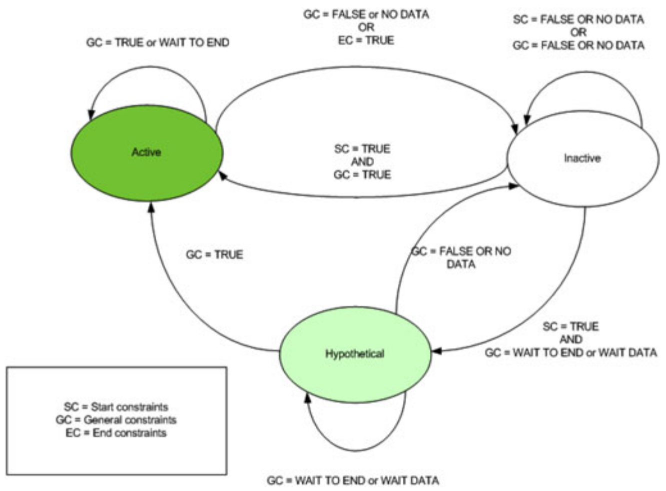

Fig. 5. State diagram of the interpretation level.

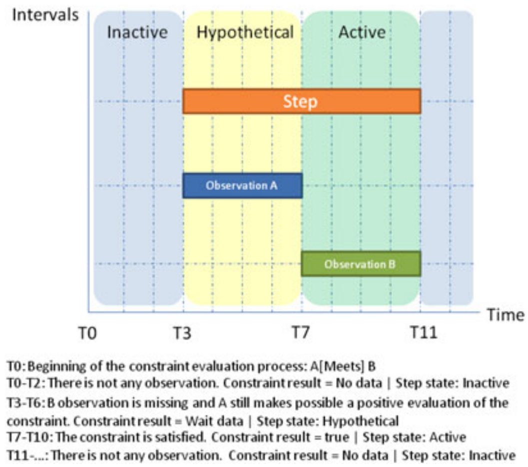

Fig. 6. Example of the evaluation of a temporal constraint.

related to movements can also be combined. Example:

- *Duration(FootStrike) > 1000 AND Duration(FootStrike) < 3000 AND Runner.HeartRate < 180*

The conceptual meaning of each category of constraints is the usual one that is found in literature. However, the particularity of this approach lies in the temporal management of the observations and the constraint satisfaction process itself, since observations are time intervals containing properties as data streams, while usually only punctual events are managed. Furthermore, this approach allows different types of constrains to be combined within the same model.

Given a set of observations, the interpreter evaluates constraints in each interpretation cycle in order to determine whether or not those observations satisfy them. It holds in memory, cycle by cycle, the state of all the constraint evaluations and the observations that they involve in order to maintain temporal cohesion. Besides the constraint evaluation results, the interpretation subsystem also needs to determine the interpretation state where a step or situation is, as a real tutor would do. For example, when a trainee is making a movement, the tutor can think that the trainee is no longer making the movement that she was doing some milliseconds ago, so the tutor can "discard" this movement from his mind. Or going the other direction, the tutor can confirm that in fact the trainee is making a movement that some milliseconds ago did not clearly appear to be that exact movement. Based on this schema, the interpreter decides if an interval is started, ended or expanded based on the possible constraint evaluation results:

- *True*. The constraint is completely satisfied.
- *False*. The constraint is not satisfied.
- *Wait data*: There are not enough observations to evaluate the constraint.
- *Wait to end*. It is not possible to know if the constraint is satisfied until the observation ends or the uncertainty in the observation subsystem is resolved.
- *No data*. None of the existing observations are referenced in the current constraint evaluation.
- *Abort*. The evaluation of this constraint has to wait because one or more intervals that are needed to evaluate this constraint have not been evaluated yet.

Based on these evaluation results of the Start, General and End Constraints, the interpreter determines the state of every step and situation in the interpretation model, denoting it as Active, Inactive or Hypothetical. If a step or situation is being executed, it will be in the Active state; otherwise, it will be in the Inactive state. However, there are moments where the interpretation cannot confirm that a step or situation is active because of the uncertainty or because the observations that have been extracted on the current instant do not provide enough information. In this case, those intervals are placed in a hypothetical state. In this regard, the general process that the interpreter executes in each cycle consists of controlling the state of each step and situation and to change it if it is required. Fig. 5 illustrates the different state transitions of steps and situations:

Fig. 6 shows the interpretation process for a simple step modeled with the General Constraint A [Meets] B. In order to satisfy this constraint, observation A has to finish at the same instant that observation B begins, and the interpreter has to hold in memory the interpretation state of the constraint and the observations detected during the full interval.

Besides interpreting step and situation states, the interpretation subsystem also decides when to transmit the interpretation result to the diagnosis subsystem. In order to do this, steps are defined with another set of constraints called *action constraints*. When these constraints are satisfied, the interpretation subsystem notifies that a diagnosis is needed at that moment. If the step has not defined action constraints, the interpreter notifies the diagnostic system of every change in the state of steps and situations. This possibility is useful if a step needs to be diagnosed only when a specific condition is fulfilled, but not every time that the step is carried out. In addition, if an active step changes its state to inactive or if its situation is finished, it does not need to be diagnosed anymore.

4.3 Diagnosis Level

The diagnosis level specifies how trainees have to be evaluated when they are practicing their motor skills, which involves detecting their errors. To achieve this aim, the diagnosis subsystem receives from the interpreter the steps that are being carried out and also the situations that are associated with them. The steps and situations that have to be diagnosed are specified in the *diagnosis model*. When real tutors define a training session, they firstly think about the situations trainees are going to face, *which actions* are going to be diagnosed and how those situations will be solved. As Section 2.3.1 described, ULISES accepts the integration of different diagnosis techniques to diagnose solutions to situations. However, regardless of the diagnosis technique, they are all composed of the following three elements that define the task model:

1) Situations that the trainees have to face.
2) Solutions for each situation: Incorrect and correct solutions can be defined for each situation, and it is possible to use a different diagnosis technique for each of the solutions.
3) Steps that trainees have to carry out in order to solve each situation. Depending on the context (situation) where steps are executed, the solution for the step can be different so steps are contextualized for each of the solutions.

In this study, we decided to integrate a constraint-based diagnosis technique. The domains where motor skills are diagnosed are considered ill-defined domains, so constraint-based modeling is a suitable approach [39]. The reason why constraint-based modeling is used in this type of domain is because rather than defining a way to solve a problem, constraints allow for the definition of how certain actions should be performed in order to detect mistakes. Other approaches, such as plan recognition, are not as suitable for this kind of domain [11].

In our implementation of a constraint-based diagnosis technique, solutions for situations are composed of durative steps. The algorithm checks the correctness of the steps cycle by cycle, maintaining temporal cohesion during the full duration of the step similar to the way that the interpreter does. Each step has a set of conditions that need to be satisfied; that is to say, conditions are the elements that make it possible to identify contextualized learner mistakes. They contain the set of constraints that must be satisfied plus additional information (such as numerical weights that represent the importance of not respecting the constraints within the situation). Constraints can specify permissible values for an observation's properties (direction of trajectories, curvature, velocity, etc.), as well as the qualitative and quantitative temporal relations between multiple steps during the situation. Therefore, temporal cohesion between constraint evaluation results is also maintained within the situation. Specific examples of constraints for diagnosing motor skills are provided in the next section. Finally, constraint evaluation results, together with the additional data from conditions, are used to calculate the correctness of the step and the weighted average note of each situation is obtained. This way of defining solutions can be used, for example, to generate positive feedback, which is important in learning environments [40].

5 Implementation of Motor Skills in ULISES

In order to diagnose a motor skill, we currently focus on four features of the movements:

1. *Coordination*. This is achieved when different movements of body parts are combined in a manner that is well timed.
2. *Poses*. Refers to the intentional configuration of the human body.
3. *Movement trajectories*. The path that a moving joint (or tracked element) follows during a period of time.
4. *Procedure of a sequence of movements*. This specifies how different steps have to be sequenced.

In view of these features, we have chosen different tennis movements to validate our work. In this section, we first explain how we modeled the observation, interpretation and task models. Then we describe the experiment we carried out, followed by a report of the experimental results and a discussion.

5.1 Practical Case Study: Diagnosis of Tennis Skills

In this section we describe how we modeled the observation, interpretation and task models in order to diagnose two tennis skills: the serve and the forehand tennis shot. In both cases, we only took into account upper body movements.[1] The capturing system used is the Microsoft Kinect Sensor. Since the objective of this study is to demonstrate that ULISES is in fact suitable for observing, interpreting and diagnosing motor skills, rather than undertaking an exhaustive quantitative analysis of the movements, we consider the Kinect Sensor's accuracy to be sufficient for this study. Nevertheless, OLYMPUS allows easy integration of other MoCap systems. The observation, interpretation and task models were defined using PATH, the authoring tool in the OLYMPUS platform that makes the creation of the models easier. In order to create these models, the first step was to define the situations and the steps to be diagnosed. For the diagnosis of tennis skills, we defined the following steps and situations (Table 1).

5.1.1 Observation Model

The observation model defines the observations that are needed in the interpretation and task models. With regard to the observations that are representative of movements, they are calculated based on a coordinate system placed on the center of the hip. In Table 2, we explain the observations that are necessary to interpret and diagnose the steps of this practical case. Overall, the model is composed of 15 properties.

5.1.2 Interpretation Model

In the interpretation model all the steps that need to be interpreted are defined. Some steps are aimed at diagnosis, but others can be defined as a part of the definition of other steps, using the same syntax shown in Table 4 (STEPS::). Table 3 explains the definition of each step of the interpretation model.

5.1.3 Task Model

In Table 4, the situations (game and serve) and the steps that compose the solution of each situation are explained. Each

1. The definitions of movements have been retrieved from ITF Coaching (http://en.coaching.itftennis.com/home.aspx).

TABLE 1
Steps and Situations

Step	Situation	Description
Forehand swing	Game	This is a special movement to add spin to the ball. The higher the curvature of the racket's trajectory is, the bigger the spin added to the ball will be.
Forward swing	Serve	This is a movement where the racket is moved in front of the body. In good serves, the racket movement should start by moving the racket behind the body. If this movement is done with the racket in front of the body the movement is incorrect.
Ball toss	Serve	This represents the step of tossing the ball. Trainees should start tossing the ball at the same time that they start moving the racket. When the ball is tossed the arm has to be extended.
Follow-through	Serve	This represents the movement that is made after hitting the ball. The racket motion should continue moving to the side of the tossing arm and the body should be parallel to the net as a consequence of the body's forward rotation.
Back swing	Serve	The hitting arm comes down to point the racket to the floor and it is raised behind the back side. The body should be more or less perpendicular to the net in order to take advantage of the forward rotation of the trunk.
Lifting tossing arm	Serve	The Tossing arm has to be lifted higher than head height.
Trophy pose	Serve	This is the completion of the service preparation. This is the position where the tossing arm is extended straight up into the air and the racket and hitting arm are in an "L" position. The prolongation of this step results in the loss of elastic energy. When the arm is in the "L" position the elbow should not be too low.

Steps, situations and a short description of the step and how it should be carried out.

step has one or more conditions that have to be satisfied, and these conditions are defined using constraints.

5.2 Authoring Process and Scalability

The PATH authoring tool guides instructional designers in defining ULISES' knowledge model. It promotes the methodology for defining training tasks and it provides a set of tools to facilitate the process of creating the knowledge model.

The authoring methodology for creating tasks with PATH is iterative: Firstly, the steps and situations that need to be solved by the trainees are defined. Then, an expert demonstrates in the VRIS how the task should be performed and PATH captures its activity. The instructional designer uses the visual tools provided by PATH to analyze the captured data. This way the behavior of the observations can be analyzed rigorously so the patterns and relations among the observations are identified, in order to create constraints for steps and situations. Once a version of the models is created, PATH reproduces the demonstrations and it generates interpretation and diagnostic results using the defined models. Instructional designers visualize these results in the monitoring tool included in PATH, in order to verify whether the results fit expert criteria. This iteration is

TABLE 2
Observations and Its Properties

Observation	Properties	Description
Left hand movements (HandLeftClass)	• Position • ZNodePos • YNodePos • XNodePos • Arc	These observations are used to represent the 26 classes of movements that can be made with the left hand (Fig. 4 a). The properties of each observation are the position of the left hand and the arc that describes the movement.
Right hand movements (HandRightClass)	• Position • ZNodePos • YNodePos • XNodePos • Arc	These observations are used to represent the 26 types of movements that can be made with the right hand.
Left arm flexion	• Angle	This observation indicates that the left arm is bent. Its property indicates the flexion angle of the arm.
Body	• Angle	The angle property of the body observation indicates the body`s angle relative to the Kinect. This is used to know if the trainee is parallel or perpendicular to the net.
Head	• Height	This observation represents the head's position.
Toss hand high	• Height	This observation is active when the toss hand is higher than the head. Its property represents the exact height of the toss hand.
Left Elbow	• Angle	The angle property of the left elbow observation represents the angle between the left arm and the body. If the angle is low, it means the elbow is in a low position.
Tracking		This observation indicates if the skeleton is being tracked.

Observations and properties that define the observation model.

TABLE 3
Definition of the Steps

Step	Observations	Instances	Constraints
Hand behind	HandLeftPosition	hl	*General:* hl.PosNodeZ<0.0
Description: This step is active when the hand is behind the body			
Toss begin	FUZZY::HandRight010 FUZZY::HandRight011 FUZZY::HandRight-111	handUp handFU handRFU	*General:* handUp.Arc(>0.0, < 1.0, >0.4) OR handFU.Arc(>0.0, < 1.0, >0.4) OR handRFU.Arc(>0.0, < 1.0, >0.4)
Description. This step begins when the trainee starts the movement to toss the ball. The Arc(>0.0, < 1.0, >0.4) property indicates that its radius has to be less than 1 meter and the linear distance has to be higher than 0.4 meters. This assures that small hand movements will be ignored and that only significant movements will be taken into account. *Note. The FUZZY prefix is used to indicate that the observation is of the fuzzy type.*			
Trophy pose	HandLeftPosition LeftArmFlexion	hp laf	*General:* laf AND hp.PosNodoZ < 0.0
Description. The Trophy pose is detected when the arm is bent (~90°) and behind the body.			
High toss ball	TossHandHigh	hbh	*General:* hbh
Description. This step is detected when the tossing hand is higher than the head			
Hand ahead	HandLeftPosition	hl	*General:* hl.ZNodePos > 0.0
This step is detected when the hand is in front of the body.			
Follow through	FUZZY::HandLeft0-10 FUZZY::HandLeft-1-11 FUZZY::HandLeft0-11 FUZZY::HandLeft-1-10	b c d e	*General:* b.Arc(>0.0, < 1.0, >0.28) OR c.Arc(>0.0, < 1.0, >0.28) OR d.Arc(>0.0, < 1.0, >0.28) OR e.Arc(>0.0, < 1.0, >0.28) *Action: b OR c OR d OR e*
Description. As this step is the last step within the serve, we define an action constraint in order to assure that the step is diagnosed before the serve situation is finished.			
Back swing	HandLeft11-1 HandLeft110 HandLeft01-1 HandLeft-11-1 HandLeft010 HandLeftPosition	hlUB hlUF hlUN hlUB2 hlU hp	*General:* (hlUB OR hlUF OR hlUN OR hlU OR hlUB2) AND hp.PosNodoZ < 0.0
Description. Back swing starts when the racket starts moving backwards			
Forward swing	FUZZY::HandLeft010 FUZZY::HandLeft011 HandLeftPosition	handUp handFU hl	*General:* handUp.Arc(>0.0, < 1.0, >0.3) OR handFU.Arc(>0.0, < 1.0, >0.3)
Description. Forward swing movement can be executed behind or in front of the body. Even if the execution of this step from behind the body is considered incorrect in the serve situation, it has to be interpreted without considering its correctness or incorrectness.			
Forehand swing	FUZZY::HandLeft-111 FUZZY::HandLeft011 FUZZY::HandLeft-101 FUZZY::HandLeft001	hl hl2 hl3 hl4	*General:* hl1.Arc(>0.0,>0.0,>0.35) OR hl2.Arc(>0.0,>0.0,>0.35) OR hl3.Arc(>0.0,>0.0,>0.35) OR hl4.Arc(>0.0,>0.0,>0.35)
Description. The Arc(>0.0,>0.0,>0.35) property indicates that its linear distance has to be higher than 0.35 meters. This way small hand movements are ignored.			

repeated until successful results are obtained. PATH is an essential tool both for defining new tasks and extending existing ones.

The model's growth does not necessarily imply a negative impact on the success rates of the interpretation and diagnostic systems. From our experience, potential problems might arise when two similar movements are interpreted. In this case, the transformation of movements into sequences of arcs would produce collisions if the constraints of the models were ambiguous. For example, the movements performed on a tennis serve are usually similar to a smash. In our opinion, the easiest way to avoid misinterpretations is to include observations in the constraints about the context of the actions. In this case, adding an observation that represents the phase of the game where the two movements are being carried out (serve and game), would be sufficient for differentiating them.

Another issue related to scalability is the level of diagnostic detail that is desired: a very detailed diagnosis involves increasing the number of constraints, which makes the models more complex.

Taking these problems into account, PATH offers different features for dealing with them. Firstly, the iterative authoring process defined in PATH and its visual tools facilitate the detection and management of ambiguities as the models grow. Moreover, it also encourages the reusing of previously created models in order to make the task creation faster.

TABLE 4
Situations and Its Solutions

Situation: Step	Conditions over	Observations:instances	Constraints
Game: Foreh and Swing	Racket trajectory	FUZZY::HandLeft-111: hl1 FUZZY::HandLeft011: hl2 FUZZY::HandLeft-101: hl3 FUZZY::HandLeft001: hl4	Hl1.Arc(>90.0, > 0.1, >0.0) OR hl2.Arc(>90.0, > 0.1, >0.0) OR hl3.Arc(>90.0, > 0.1, >0.0) OR hl4.Arc(>90.0, > 0.1, >0.0)
Description. In order to add considerable spin to the ball, the arc of the movement has to be greater than 0.1 meters. In addition, the movement's trajectory has to be concave. *Features.* Trajectory			
Serve:Forward swing	ArmPosition	HandLeftPosition: hp	hp.PosNodoZ < 0.0
Description. The racket has to move behind the body until the ball is hit. If the forward swing move is carried out with the racket in front of the body the step will be considered incorrect. *Features.* Trajectory			
Serve:Trophy pose	1) Elbow height 2) Duration	1) LeftElbow: le 2) LeftArmFlexion:laf	1) le.Angle > 80.0 2) Duration(laf) < 500
Description. When the arm is in trophy pose, the elbow must not to be very low because efficiency is lost and it can cause injuries. If the arm stays in the pose too long, elastic energy is lost. *Features.* Pose, trajectory			
Serve:**toss begin**	1) Coordination	1) STEPS::TossBegin: rtb STEPS::BackSwingStart: bss	1) Duration(rtb.Start,bss.Start) < 200 OR Duration(bss.Start,rtb.Start) < 200
Description. This step has to begin at the same time as the back swing begins. *Features:* Coordination *Note. The STEPS prefix is used to indicate that the instance is a step, not an observation.*			
Serve: Follow through	1) Body position 2) Racket position 3) Lift hand 4) Procedure	1) BodyAngle: ba 2) FUZZY::HandLeft0-11: hl FUZZY::HandLeft-1-11: hlb 3) STEPS:: HighTossBall: ma STEPS::FollowThrough: ft 4) STEPS::TrophyPose: tp STEPS::FollowThrough: ft	1) ba.Angle < 150.0 AND ba.Angle > 100.0 2) hl(>= 0.0, >= 0.0,>= 0.0, >= 0.0,>= 0.0, >= 0.0, >= 0.0, >= 0.0, >= 0.0,>= 0.0, >= 0.0, >= 0.0,>= 0.0,>= 0.0,>= 0.0,>= 0.0,>= 0.0,>= 0.0, >= 0.0,>= 0.,>= 0.0,>= 0.0,>= 0.0,>= 0.0,>= 0.0 AND <100.0,>= 50.0) OR hlb 3) ma [Precedes] ft 4) tp [Precedes] ft
Description. 1) When the follow through is executed, the body has to be more or less parallel to the net. 2) The racket motion should continue until it arrives on the side of the tossing arm. That means that the movement is diagonal (handleft-1-11 movement) or similar: the observation HandLeft0-11 has to be similar to HandLeft-1-11 by more than 50 percent. 3) Before the follow through step, the HighTossBall step has to be executed (when the ball is tossed the hand has to be lifted higher than the head). 4) Before the follow through step, the TrophyPose step has to be executed (the arm has to be flexed). *Features.* Pose, procedure, trajectory *Note. The FUZZY prefix is used to indicate that the instance is a special type of observation: an observation with an arc property or fuzzy logic*			
Serve: Back swing start	1) Body position 2) Coordination	1) BodyAngle: ba 2) STEPS::TossBegin: rtb STEPS::BackSwingStart: bss	1) ba.Angle > 195.0 AND ba.Angle < 225.0 2) Duration(rtb.Start,bss.Start) < 200 OR Duration(bss.Start,rtb.Start) < 200
Description. 1) When the serve begins (back swing starts) the body has to be placed perpendicularly to the net. 2) This step has to be coordinated with the toss beginning. *Features.* Pose, Coordination			
Serve: High toss ball	1) Extend arm	1) RightElbowAngle: rea	1) rea.Angle > 140.0
Description. When the ball is tossed the arm should be extended. *Features.* Pose			

The condition name represents the characteristic of the movement that the constraints are applied over.

6 Experiment

We performed an experiment in order to evaluate the accuracy of the motor skills diagnosis generated with ULISES. For this purpose, we conducted a performance study of tennis serves with 15 right-handed participants, none of whom had mastered tennis. Each of them carried out 10 serves with the left hand, a condition we established to equalize the skill conditions between the 10 subjects. In this experiment we used a Kinect Sensor to capture subjects' motion and also a tennis net to provide a static reference in the serves. In order to validate our approach, we measured the number of successful interpretations and diagnostics for each step of the serve situation (Table 5). As the objective of the experiment was to diagnose tennis serves, we only took into account steps where the subjects were carrying out serves; situations where subjects were positioning themselves were discarded. Similarly, we have taken into account those situations where occlusions of the MoCap system did not allow a correct interpretation of the steps being made. Table 5 shows the results of the experiment.

TABLE 5
Experiment Results for Each Step

Step name	Number of executions of the step	Number of executions where the step was diagnosed as incorrect	Interpretation accuracy	Diagnosis accuracy
Backswing start	128	50	97.3%	100%
Trophy pose	109	73	96.3%	98.5%
Forward swing	101	51	100%	100%
Toss begin	155	117	96.9%	94%
High toss ball	101	87	100%	100%
Follow through	147	140	98%	98.52%
GLOBAL	741	518	98.1%	98.5%

6.1 Results and Discussion

We obtained the results according to the following procedure: We recorded all the serves, storing the skeleton data generated with the Kinect Sensor. We reproduced each recorded serve and an expert provided his own interpretation and diagnosis results, i.e. the expert reported for all the experiments which steps were carried out and whether they were carried out correctly or incorrectly. The expert applied the same criteria that were used to define the interpretation and task models, ignoring any other subjective assessment that was not considered in the models. These results were compared with the evaluation results generated by ULISES for trainees and instructors (Fig. 7). The evaluation report shows an interactive chronogram with all the executed steps, which represents their correctness with different colors. Moreover, clicking on each step shows the conditions that were satisfied and not satisfied and the cause of mistakes. Evaluation marks are calculated based on this, although we have neither calibrated nor validated them in this experiment since our focus was on error detection and a different experiment would be necessary to do that.

Table 5 shows the experiment results for each of the steps that were to be diagnosed in the serve situation. Interpretation accuracy expresses the percentage of correctly interpreted steps out of the total steps that the trainees have executed. Diagnosis accuracy represents the number of step diagnoses that coincide with a real expert's diagnosis out of the total diagnosed steps. The total diagnosed steps take into account only those steps that were interpreted correctly. In addition, these results are also broken down by the motor skill features that we wanted to evaluate (Table 6).

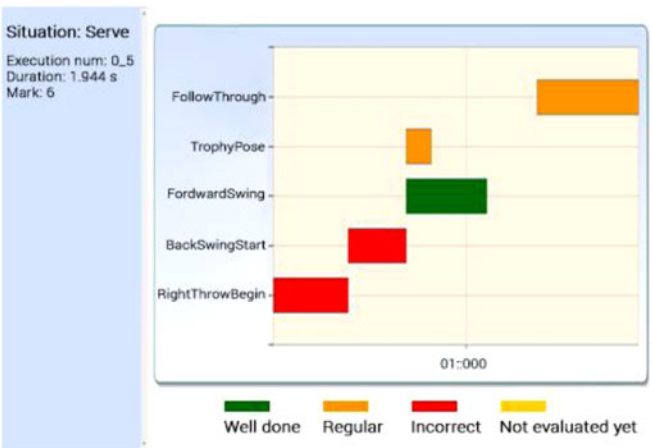

Fig. 7. Partial view of the evaluation results generated with the monitoring tool in OLYMPUS.

The experimental results show that it is possible to successfully interpret (with 98.1 percent accuracy) and diagnose (with 98.5 percent accuracy) the steps we have modeled to evaluate a tennis serve using a constraint-based technique. The "Trophy pose" and "Toss begin" steps were the least accurate steps in terms of interpretation. The interpretation constraint for "Trophy pose" was modeled well enough to detect the correct pose correctly, so it was detected more times than it should have been. However, the problem with the "Toss begin" step was that sometimes this step was not detected or that it was detected late. With regard to diagnosis accuracy, the "Toss begin" step scored the lowest accuracy. "Toss begin" was modeled to diagnose the coordination between the "Toss begin" and "Backswing start" steps; however, our model did not contemplate all the different relations between those two steps, e.g. the cases where each step was carried out more than once. This circumstance therefore influenced the diagnosis result for the "Toss begin" step, which in turn influenced the diagnosis result of the coordination feature (92.5 percent, Table 6). The results show that the best performed step was the Backswing start (50 incorrect executions out of 128 executions), while the worst was the Follow Through step. This is logical because the Follow Through step is composed of four conditions that need to be satisfied, so it was easy to make at least one error per each execution of this step.

In order to obtain these results, interpretation and task models were modeled following the iterative authoring process described in Section 5.2 by capturing the movements of

TABLE 6
Experiment Results for Each Motor Skill Feature

Feature name	Number of times that a feature has been diagnosed	Number of times that a feature was diagnosed as incorrect	Diagnosis accuracy
Coordination	156	119	92.5%
Pose	473	299	99.7%
Trajectory	332	147	99.4%
Procedure	293	99	99.3%
GLOBAL	1254	664	97.7%

two expert volunteers. In this case, instructional knowledge was retrieved from the ITF Coaching webpage and the models were calibrated until we achieved successful results prior the experiments with the fifteen subjects. The interpretation and task models were defined to diagnose only steps involved in tennis serves. We did not take into account other movements that were not related to a tennis serve, such as the movements that were carried out while the trainees were positioning themselves. Those movements were of no interest to the analysis of the movements and they would have unnecessarily made the models more complex. Therefore, we discarded those movements, of which there were 49 altogether. Nevertheless, our models can be completed to take into account those movements. Besides the positioning-related discards, 75 steps were discarded because of the occlusions generated by the Kinect sensor. Most of the occlusions were generated because the racket-holding arm was behind the body. These occlusions would not be a problem in a game, but they would be if a correct interpretation and diagnosis needed to be done. A possible solution would be to use more than one Kinect sensor to obtain a better field of vision or to use another MoCap system with cameras surrounding the full capture region, which would not involve any change in the models.

As we have explained throughout this paper, detecting the intentionality of the movements is a key part of our work. Movement intentionality is detected by the interpretation subsystem, whose result (98.1 percent) indicates that our approach works successfully. The steps "Toss begin", "Follow through" and "Forward swing" were modeled using arcs, and they were suitable for detecting intentionality. In addition, by establishing a constraint in the arc's length, the interpreter was able to discard movements that were insignificant. In the "Back swing" step, intentionality was modeled by defining the possible movements that trainees could execute when they were trying to make a "Back swing". In this case, the detection of intentionality was also successful (97.3 percent). These are two ways of detecting intentionality in the movements, but defining the interpretation model is a subjective task that depends on expert criteria. Using these criteria as a base, there are several ways to detect intentionality, and the constraint-based modeling used in ULISES provides enough flexibility to model it.

In terms of diagnosis, results show that our approach was able to correctly diagnose the four basic features of the movements: coordination, poses, movement trajectories and the procedure that has to be followed in a sequence of movements (the last column in Table 4 describes the features that are to be diagnosed in each step). Firstly, coordination was diagnosed by establishing temporal relations between steps: The "Toss begins" step requires that "Back swing start" starts at the same time. The procedure that has to be followed in the serve was successfully diagnosed (99.3 percent accuracy) in the "Follow through" step, which checks if the right hand has been lifted and the arm flexed before the serve is finished. The correct diagnosis of these three steps demonstrates that temporal uncertainty has been handled correctly, because the relation between steps in both coordination and procedural diagnosis cannot be established until a time period elapses. This means that the temporal context has been handled correctly in both the interpretation and diagnostic subsystems. Secondly, poses were correctly diagnosed in "Forward swing" (checks if the hand is being moved behind body), "Trophy pose" (checks the correct L position of the arm during the movement), "Follow through" (checks if the toss hand has been lifted correctly), "Back swing" and "High toss ball" (they check the correct position of the body). Lastly, trajectories were correctly diagnosed in the "Follow through", "Forward swing" and "Trophy pose" steps. Besides correctly diagnosing these steps in terms of the four movement features, the experiment showed that the diagnosis results generated with ULISES can be qualitatively expressed. Table 7 shows some different feedback messages that we generated for the different steps:

TABLE 7
Feedback

Situation: Step	Feedback
Game: ForehandSwing	Give more spin to the ball, lift up the racket.
Serve:Forward swing	Wrong serve. You have to move your left arm backwards, not forward.
Serve:Trophy pose	1) Your elbow was too low when you flexed your arm. 2) You have lost a lot of elastic energy in preparing the shot.
Serve: Toss begin	Start throwing the ball at the same time you move your left arm backwards.
Serve: Follow through	1) You have to end the serve parallel to the net. 2) You have finished the serve correctly. You have to make the complete movement of the racquet, bringing it to the right side of your body. 3) You have not lifted your hand enough when you tossed the ball. 4) You have not flexed your arm enough before you hit the ball.
Serve: Back swing start	1) You have to start the movement from a position that is perpendicular to the net. 2) Start throwing the ball at the same time you move your left arm backwards
Serve: high toss ball	You have to stretch your arm more when you are tossing the ball.

List of feedback generated from the diagnosis results.

Even though we saw successful results, further work must be done. The objective in this experiment was to demonstrate that our system is able to discern what actions the trainees perform, in addition to correcting them, from an educational point of view by dealing with some classic problems related to Motion Capture, such as uncertainty, intentionality, action recognition, skill modeling, semantic extraction and so on. However, more experiments must be done in order to demonstrate how much ULISES contributes to the training of motor skills. Lastly, although we used an authoring system, we believe that it is not easy to define the arc and fuzzy properties of a movement. As we noted earlier, we needed several iterations during the authoring process to achieve these results, and more iterations would be needed to improve the interpretation and diagnosis accuracy rates even more. We think that the cost of defining arcs with constraints can be reduced significantly by using an automated model generation approach. At the moment, we are working on a module for PATH that is able to define partial solutions based on expert examples. A similar approach was proposed by Fournier-Viger and Nkambou [41], but they have not yet made use of this approach in the area of motor skill evaluation. Nevertheless, according to our knowledge, the cost of defining a model and adding new movements to a model with our approach is easier than doing it with existing action recognition techniques. When classifiers are used, if a model is going to be redefined or a new movement is going to be added, the classifiers need to go through the learning process again. What is more, our approach does not depend on the properties of the domain where the actions are being carried out, unlike most of the existing action recognition techniques.

7 Conclusion

In this paper we have presented a technique for providing IILSs with the capability to give assistance to trainees while they learn motor skills. We have adapted the observation, interpretation and diagnosis subsystems of the ULISES framework in order to deal the special characteristics of motor skills. This has allowed us to transform the quantitative variables of a movement to a qualitative domain and generate diagnosis results that can be used by other educational components. We have validated our work by creating models that provide diagnosis for tennis skills, taking into account coordination, poses, movement trajectories and the procedure that a tennis serve should follow. The initial evaluation results were positive, and they serve as proof that our approach is effective. Additionally, constraint-based modeling has been shown to be effective for use in both the interpretation and diagnosis of motor skills. What is more, our approach allows the knowledge of an expert to be represented by adapting the observation, interpretation and task models to his experience. However, the definition of the models can become complex, so an automated model generation approach would considerably reduce the effort that goes into generating models for the diagnosis of motor skills.

Apart from motor skills, ULISES has been tested in a truck driving domain and in the development of an IILS for gardening tasks for mentally challenged people [31], demonstrating that the framework is independent from the domain. However, we are interested in expanding the model to diagnose another set of motor skills and to use a different MoCap system for further validation of ULISES in the domain of motor skills.

Acknowledgments

The authors want to thank the Spanish Ministry of Education and Science and Spanish Ministry of Industry, Trade and Tourism and CDTI and for making this work possible with their research grants.

References

[1] J. Rickel, J. Gratch, R. HIll, S. Marsella, and W. Swartout, "Steve goes to Bosnia: Towards a new generation virtual humans for interactive experiences," in *Proc. AAAI Spring Symp. Artif. Intell. Interactive Entertain.*, 2001, pp. 2–3.

[2] P. Fournier-Viger, R. Nkambou, and E. M. Nguifo, "ITS in Ill-defined domains: Toward hybrid approaches an hybrid model in canadarmtutor," in *Intelligent Tutoring Systems*. Berlin, Germany: Springer, 2010, pp. 318–320.

[3] K. Tervo, L. Palmroth, and H. Koivo, "Skill evaluation of human operators in partly automated mobile working machines," *IEEE Trans. Autom. Sci. Eng.*, vol. 7, no. 1, pp. 133–142, Jan. 2010.

[4] V. Medved, *Measurement of Human Locomotion*. Boca Raton, FL, USA: CRC Press, 2001.

[5] D. Weinland, R. Ronfard, and E. Boyer, "A survey of vision-based methods for action representation, segmentation and recognition," *Comput. Vis. Image Underst.*, vol. 115, no. 2, pp. 224–241, Feb. 2011.

[6] O. Mena, L. Unzueta, B. Sierra, and L. Matey, "Temporal nearest end-effectors for real-time full-body human actions recognition," in *Articulated Motion and Deformable Objects*. Berlin, Germany: Springer, 2008, pp. 269–278.

[7] J. F. Allen, "Maintaining knowledge about temporal intervals," *Commun. ACM*, vol. 26, no. 11, pp. 832–843, 1983.

[8] J. C. Niebles, C.-W. Chen, and L. Fei-Fei, "Modeling temporal structure of decomposable motion segments for activity classification," in *Proc. 11th Eur. Conf. Comput. Vis.*, 2010, pp. 392–405.

[9] R. M. Gagné, *The Conditions of Learning and Theory of Instruction*. New York, NY, USA: Holt, Rinehart and Winston, 1985.

[10] F. Kabanza and R. Nkambou, "Path-planning for autonomous training on robot manipulators in space," in *Proc. 19th Int. Joint Conf. Artif. Intell.*, 2005, pp. 1729–1731.

[11] A. Mitrovic and A. Weerasinghe, "Revisiting Ill-definedness and the consequences for ITSs," in *Proc. Conf. Artif. Intell. Educ. Build. Learn. Syst. Care Knowl. Represent. Affect. Model.*, 2009, vol. 200, p. 375.

[12] P. Fournier-Viger, R. Nkambou, E. M. Nguifo, A. Mayers, and U. Faghihi, "A multiparadigm intelligent tutoring system for robotic arm training," *IEEE Trans. Learn. Technol.*, vol. 6, no. 4, pp. 364–377, Oct.–Dec. 2013.

[13] J. Yang, Y. Xu, and C. S. Chen, "Human action learning via hidden Markov model," *IEEE Trans. Syst. Man Cybern. A, Syst. Humans*, vol. 27, no. 1, pp. 34–44, Jan. 1997.

[14] J. Solis and A. Takanishi, "Enabling autonomous systems to perceptually detect human performance improvements and their applications," in *Proc. IEEE Int. Conf. Autom. Sci. Eng.*, 2008, pp. 259–264.

[15] S. Theodoridis and K. Koutroumbas, "Pattern recognition," *CA Acad.*, San Diego, CA: Academic Press, 1999.

[16] S. Cotin, N. Stylopoulos, M. Ottensmeyer, P. Neumann, D. Rattner, and S. Dawson, "Metrics for laparoscopic skills trainers: The weakest link!," in *Proc. 5th Int. Conf. Med. Image Comput. Comput.-Assisted Intervention*, 2002, pp. 35–43.

[17] J. Rosen, M. Solazzo, B. Hannaford, and M. Sinanan, "Task decomposition of laparoscopic surgery for objective evaluation of surgical residents' learning curve using hidden Markov model," *Comput. Aided Surg.*, vol. 7, no. 1, pp. 49–61, 2002.

[18] L. Vemer, D. Oleynikov, S. Holtmann, H. Haider, and L. Zhukov, "Measurements of the level of surgical expertise using flight path analysis from da vinci^{TM} robotic surgical system," *Med. Meets Virtual Real. 11 NextMed Heal. Horiz.*, vol. 94, p. 373, 2003.

[19] Y. Yamauchi, J. Yamashita, O. Morikawa, R. Hashimoto, M. Mochimaru, Y. Fukui, H. Uno, and K. Yokoyama, "Surgical skill evaluation by force data for endoscopic sinus surgery training system," in *Proc. Med. Image Comput. Comput.-Assisted Intervention*, 2002, pp. 44–51.

[20] T. E. Murphy, C. M. Vignes, D. D. Yuh, and A. M. Okamura, "Automatic motion recognition and skill evaluation for dynamic tasks," in *Proc. Eurohaptics*, 2003, pp. 363–373.

[21] J. Rosen, B. Hannaford, C. G. Richards, and M. N. Sinanan, "Markov modeling of minimally invasive surgery based on tool/tissue interaction and force/torque signatures for evaluating surgical skills," *IEEE Trans. Biomed. Eng.*, vol. 48, no. 5, pp. 579–591, May 2001.

[22] J. Rosen, L. Chang, J. D. Brown, B. Hannaford, M. Sinanan, and R. Satava, "Minimally invasive surgery task decomposition-etymology of endoscopic suturing," *Stud. Hehtmlalth Technol. Informat.*, vol. 94, pp. 295–301, 2003.

[23] Z. Ruttkay and H. Van Welbergen, "Elbows higher! performing, observing and correcting exercises by a virtual trainer," in *Proc. 8th Int. Conf. Intell. Virtual Agents*, 2008, pp. 409–416.

[24] D. Davcev, V. Trajkovic, S. Kalajdziski, and S. Celakoski, "Augmented reality environment for dance learning," in *Proc. Int. Conf. Inf. Technol.: Res. Educ.*, 2003, pp. 189–193.

[25] K. Hachimura, H. Kato, and H. Tamura, "A prototype dance training support system with motion capture and mixed reality technologies," in *Proc. 13th IEEE Int. Workshop Robot Human Interactive Commun.*, 2004, pp. 217–222.

[26] A. Soga, B. Umino, and M. Hirayama, "Automatic composition for contemporary dance using 3D motion clips: Experiment on dance training and system evaluation," in *Proc. Int. Conf.*, 2009, pp. 171–176.

[27] K. Hachimura, K. Takashina, and M. Yoshimura, "Analysis and evaluation of dancing movement based on LMA," in *Proc. IEEE Int. Workshop Robot Human Interactive Commun.*, 2005, pp. 294–299.

[28] G. Qian, F. Guo, T. Ingalls, L. Olson, J. James, and T. Rikakis, "A gesture-driven multimodal interactive dance system," in *Proc. IEEE Int. Conf. Multimedia Expo.*, 2004, vol. 3, pp. 1579–1582.

[29] D. Y. Kwon and M. Gross, "Combining body sensors and visual sensors for motion training," in *Proc. ACM SIGCHI Int. Conf. Adv. Comput. Entertainment Technol.*, 2005, pp. 94–101.

[30] J. Chan and H. Leung, "A virtual reality dance training system using motion capture technology," *IEEE Trans. Learn. Technol.*, vol. 4, no. 2, pp. 187–195, Apr.–Jun. 2011.

[31] A. Aguirre, A. Lozano-Rodero, M. Villamañe, B. Ferrero, and L. Matey, "OLYMPUS: An Intelligent interactive learning platform for procedural tasks," in *Proc. Int. Conf. Comput. Graph. Theory Appl. Conf. Inf. Vis. Theory Appl.*, 2012, pp. 543–550.

[32] J. Clemente, J. Ramírez, and A. de Antonio, "Applying a student modeling with non-monotonic diagnosis to intelligent virtual environment for training/instruction," *Expert Syst. Appl.*, vol. 41, pp. 508–520, Jul. 2013.

[33] A. Lozano-Rodero, "Metodología de desarrollo de sistemas interactivos inteligentes de ayuda al aprendizaje de tareas procedimentales basados en realidad virtual y mixta," Univ. Navarra, Pamplona, Spain, 2009.

[34] L. A. Zadeh, "The concept of a linguistic variable and its application to approximate reasoning-I," in *Inf. Sci.*, vol. 8, no. 3, pp. 199–249, 1975.

[35] A. Panjkota, I. V. O. Stančić, and T. Šupuk, "Outline of a qualitative analysis for the human motion in case of ergometer rowing," in *Proc. WSEAS Int. Conf. Math. Comput. Sci. Eng.*, 2009, pp. 182–186.

[36] L. A. Zadeh, "Toward a theory of fuzzy information granulation and its centrality in human reasoning and fuzzy logic," *Fuzzy Sets Syst.*, vol. 90, no. 2, pp. 111–127, 1997.

[37] L. Loke, A. Larssen, and T. Robertson, "Labanotation for design of movement-based interaction," in *Proc. 2nd Australas. Conf. Interact. Entertain.*, 2005, vol. 2005, pp. 113–120.

[38] J. F. Allen, "Towards a general theory of action and time," *Artif. Intell.*, vol. 23, no. 2, pp. 123–154, Jul. 1984.

[39] S. Ohlsson, "Constraint-based student modeling," *Artif. Intell. Educ.*, vol. 3, pp. 429–447, 1992.

[40] A. Mitrovic, S. Ohlsson, and D. K. Barrow, "The effect of positive feedback in a constraint-based intelligent tutoring system," *Comput. Educ.*, vol. 60, no. 1, pp. 264–272, Jan. 2013.

[41] P. Fournier-Viger and R. Nkambou, "Exploiting partial problem spaces learned from users' interactions to provide key tutoring services in procedural and ill-defined domains," in *Proc. Artif. Intell. Educ.: Building Learn. Syst. Care: From Knowl. Representation Affective Model.*, 2009, pp. 383–390.

Aitor Aguirre received the BEng and MEng degrees in computer engineering in 2006 and 2009, respectively, from the University of Mondragon, Spain, and the PhD degree from the University of Navarra in 2013, while with the Simulation Department at CEIT, Spain. His research interest include interactive learning environments, and in particular, the application of virtual reality systems in the learning of procedural tasks and motor skills.

Alberto Lozano-Rodero received the degree in computer science from the University of the Basque Country in 2001 and the PhD degree in intelligent tutoring systems from the University of Navarra. He is a researcher in the simulation area at the Department of Mechanics, CEIT. He has been doing research on intelligent tutoring systems and virtual reality since 2000, when he was with the Group for Adaptive Teaching-Learning Environment at the University of the Basque Country. Since 2002, he has been a researcher at CEIT, developing both research and industrial projects mostly focused on applying artificial intelligence techniques in virtual reality-based environments. He has supervised two Doctoral Thesis.

Luis M. Matey Mechanical Engineer from the University of Navarra in 1991 and the PhD degree from U. Navarra in 1994. He is a professor at the University of Navarra, a senior researcher in the Simulation Area at the Department of Mechanics, CEIT. This area works on research projects on mechanism analysis, design and optimization, simulation of multidomain systems, human motion reconstruction, VR/AR applications, human-machine interfaces, and haptics. His research interests focus on HCI, visualization, virtual reality and ambient intelligence. He has collaborated on four books, more than 12 papers in journals, more than 40 international presentations, and directed nine PhD theses. He has collaborated with many companies, mainly in the aeronautic, automotive and educational fields, and he has participated in more than 40 research projects, directing many of them. He is a member of the ACM, IEEE, Euromech, and Eurographics.

Mikel Villamañe received the BEng and MEng degrees in computer engineering in 2002 and 2004, respectively, from the University of the Basque Country, Spain. He is currently a lecturer in the Computer Languages and Systems Department at the Bilbao College of Industrial Engineering (EUITI). He teaches courses in computer security, design and analysis of software and mobile application development. He is a part of the organizing committee for the Agile Spain Conference. His research interests include all aspects of computer-aided learning, but in particular the application of virtual reality and haptic systems in the learning of procedural tasks.

Begoña Ferrero received the PhD degree in 2004 in computer engineering from the University of the Basque Country, Spain. She is currently a lecturer in the Computer Languages and Systems Department at the Bilbao College of Industrial Engineering (EUITI). She teaches courses in software engineering, artificial intelligence and, also, in intelligent learning environments both the undergraduate and postgraduate levels. Her research interests include all aspects of computer-aided learning, but in particular the learning of procedural knowledge and tasks. She is member of the AIED Society and a board member of Association for the Development of Computer Education (ADIE). She has several publications and has presented many papers at national and international conferences.