Real-Time Gesture Recognition Based On Motion Quality Analysis

Céline Jost, Pierre De Loor,
Alexis Nédélec, Elisabetta Bevacqua
UEB, Lab-STICC, ENIB - France
Email: {jost,deloor,nedelec,bevacqua}@enib.fr

Igor Stanković
Dept. of Applied IT, LETStudio,
University of Gothenburg, Sweden
Email: igor.stankovic@gu.se

***Abstract*—This paper presents a robust and anticipative real-time gesture recognition and its motion quality analysis module. By utilizing a motion capture device, the system recognizes gestures performed by a human, where the recognition process is based on skeleton analysis and motion features computation. Gestures are collected from a single person. Skeleton joints are used to compute features which are stored in a reference database, and Principal Component Analysis (PCA) is computed to select the most important features, useful in discriminating gestures. During real-time recognition, using distance measures, real-time selected features are compared to the reference database to find the most similar gesture. Our evaluation results show that: i) recognition delay is similar to human recognition delay, ii) our module can recognize several gestures performed by different people and is morphology-independent, and iii) recognition rate is high: all gestures are recognized during gesture stroke. Results also show performance limits.**

Keywords*—*Gesture recognition, Quality motion features, Morphology independence

I. Introduction

In the past years, the development of virtual humanoids able to interact naturally with humans has gained more and more importance and interest in the Human-Machine Interfaces domain. A main research problem consists in making the interaction with these virtual entities as credible as possible, endowing them not only with the capability of communicating through the typical human multimodal communicative channels, but also with the ability to perceive the user and to adapt its verbal and non-verbal behaviors. Similarly, in human-human interaction, people continuously adapt and influence each other's behavior. This dynamical process has been thoroughly studied by several researchers in psychology and computer science who agree in considering it as fundamental for natural human interaction [1], [2], [3].

Within the French national project called Ingredible, we aim to model this dynamical mutual influence between users and virtual agents. This study focuses especially on the gestural behavior and gestural expressive quality shown by both agent and human. Interaction considered in the project belongs to the artistic domain, and we collaborate with a theatrical troup to define scenarios in which the whole interaction is based solely on gestural movements (neither speech nor facial expressions were considered). So, our final goal is to create a virtual character capable of maintaining a gestural coupling with a human during an artistic interaction. Movements, also called gestures in this paper, represent interaction vocabulary which

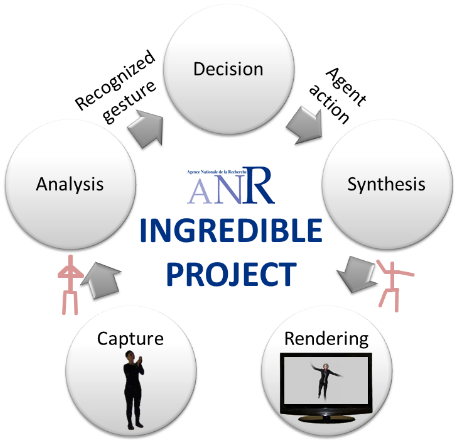


Fig. 1. Ingredible project framework. The Capture Module retrieves data from tracking devices and generates a unified 15-joints skeleton. Skeletons are being sent to the Analysis module which tries to recognize the current gesture and its expressivity. The recognition results are being sent to the Decision module which determines the agent's response depending on the user's behavior and the interaction scenario. The Synthesis module receives the choices obtained by the Decision module and computes the final skeleton animation displayed by the Rendering module.

can replace words. For example, gestures can be: waving hand to say hello, bow, applause, showing an object, showing enthusiasm and so on. Duration is rarely less than two seconds. All gestures described in this paper last between two and five seconds.

To realize such a virtual agent we propose a system able to: (i) recognize human's gestures and gestural expressivity, (ii) determine how the agent should adapt its behavior, and (iii) generate the final animation on the virtual humanoid representation. Our system is based on the framework shown in Figure 1. The entire framework is not described in this paper, which rather focuses on the progress done in the Analysis module, consisting of gesture and gestural expressivity recognition. To implement this module, the following requirements should be satisfied.

Anticipation. Firstly, keeping in mind that the desired interaction evolves continuously, the whole system must be fast enough to be interactive: the agent reactions must be triggered and displayed in a lapse of time that seems natural to the user. So, the gesture recognition module quickly and precisely detect the apparition of a new gesture at any time and to provide continuously information about the user's quality

INTETAIN 2015, June 10-12, Torino, Italy
Copyright © 2015 ICST
DOI 10.4108/icst.intetain.2015.259608

movements in order to help the Decision module to anticipate the user's behavior and to adapt the agent behavior quickly and consistently.

Full body. In interaction, our gesture recognition module should perform gesture recognition and monitor expressivity on a whole body, both the user's and the agent's.

Robustness. The gesture recognition rate should be high no matter who interacts with the virtual agent nor how the user performs gestures. People are very different and behave very differently. Even the same person does not move always in the same manner, that is showing the same expressivity. For example, waving to say hello is always the same gesture even if it is performed by shy people (barely moving, with the hand close to body) or by happy people (waving the whole arm above head, expressing enthusiasm). Therefore, our module must recognize a gesture independently of both the user's morphology and the way in which it is performed.

Number of gestures. In an artistic interaction, the number of gestures that could be performed could grow fast, depending on the topic and the length of the interaction. Given that we want our module to be adaptable to different performances and consequently able to differentiate dozens of gestures, which is not generally the case in the literature. Adding more and more gestures introduces some confusions in recognition and increases the computational cost.

Training efforts. The preparation of a theatrical performance needs some adjustment. The director makes different tests and try different scenarios. According to the director's choices, it could be necessary to modify the list of gestures to recognize. Unfortunately, to obtain a robust recognizer, many occurrences of the same gesture performed by different people should be learnt, which is time-consuming. Ideally, our Analysis module should recognize gestures from different users, having only few references gestures recorded from a single person.

Device independence. The system should work in different situations - anywhere, at any moment, and using even a simple motion capture device such as Microsoft Kinect. Input devices can be equipped with more sophisticated motion capture system which can provide more accurate skeleton-based data in real-time. For such a reason, the Analysis module must be independent from the input device by working on a minimal subset of skeleton joints.

These requirements introduce some challenges regarding the state of the art in gesture recognition. To rise these challenges, we developed a system based on the computation of several motion features that provide several information about a gesture, such as its expressivity. It also uses a sliding window to cut gestures in smaller sequential chunks, that allows us to recognize continuously gestures without detecting the start and the end of the gesture.

The paper is organized as follows. Next section presents related works about gesture recognition. In section III, the whole recognition module is described in detail. Section IV justifies some implementation choices. Finally, we present and discuss a set of evaluations in section V.

II. Related work

Most of the approaches on automatic gesture recognition tried to recognize gestures directly from video [4]. Even if high recognition rates have been obtained, features extraction remains quite a difficult task. Thanks to new tracking devices the features extraction process has become easier; low-level skeleton-based features, such as joints 3D position, angles and quaternions can be obtained in real-time. The model presented in this work uses skeleton-based features and for such a reason this brief review of related works focuses on models which use this type of data. As we will see, most of the previous studies rarely use motion features to recognize gestures. These features are mainly used to infer the user's internal state, such as emotional state [5].

Independently of the type of features considered, gesture recognition is often seen as a classification problem. A set of labeled gestures is recorded and a machine learning system is used to generate a classifier. To recognize a new gesture, its similarity to those in the classifier is computed and the closest class is selected as the most probable result. Several machine learning algorithms have been successfully implemented, among them Hidden Markov Models (HMM), Support Vector Machines (SVM), k-Nearest Neighbors (kNN), and Dynamic Time Warping (DTW) are surely the most commonly used. In [6], Gu et al. proposed a model based on HMM to recognize left arm gestures to interact in real-time with a robot. The model was implemented to recognize gestures consecutively, even if the user did multiple repetitions of the same gesture (all gestures last around one and a half second). They tested their model with five gestures performed by two subjects, of whom one was used to train the system. Recognition was around 85% for the person who provided the training data and 73% for the other one.

Consecutive upper body gestures are recognized also in [7]. Three-dimensional coordinates of seven joints were used as low-level features and two machine learning algorithms (SVM and Decision Tree) were applied and compared. Three subjects were involved in the evaluation, two were used for training purposes and one for testing. They had to perform eight aircraft marshaling gestures. Continuous recognition is assured by using a sliding window for each gesture, the window dimension corresponded to the average dimension of the gesture it was dedicated to. SVM scored better than DTW and was less subject dependent, however no information was provided about the subject morphology and the real-time performances of the system.

Ibañez and colleagues [8] proposed a tool which supports either HMM or DTW to classify and compare gestures trajectory. Low-level features are normalized to make recognition morphology independent and the tool allows user to select the important trajectory points, which implies that points selection depends on the user's expertise. With both techniques, HMM and DTW, the tool obtains good results when tested on 10 gestures, however to reach high accuracy (around 99%) many repetitions of a gesture (20 repetitions from four different subjects) are needed to train the system.

DTW technique is used also in the full body gesture recognition model proposed by Sempena et al. [9]. The model uses quaternions to avoid morphology dependence and has no learning phase: DTW is applied between each labeled gestures in the database and the new gesture to recognize. Evaluation has been done on 6 separated gestures, but no information is provided about performances.

In [10], a motion classifier for Kinect is presented. Joint position is converted to a position relative to the head and then their angle relative to the joints between the shoulders is computed. These angles are the features used for the classification which is based on the DTW technique. The system is tested on seven hand gestures of five seconds each and the recognition rate reaches 100%. Unfortunately, information about the number of repetitions used for training the system and about the samples used for testing is missing.

A real-time DTW based gesture recognition system is proposed in [11]. Raw data from an input device is reduced, utilizing PCA, to a single artificial signature generated in a 2D space. This signature is segmented to identify different phases of a gesture (start, stroke, and end). A multi-agent systems is used for real-time recognition: each time the beginning of gesture is detected, an observer agent is instantiated. When the end of the gesture is detected, the system collects all the ended observer agents and select the best one using the DTW algorithm. The system was tested on two different input devices: the Xsens Moven suit, which provides 23 human joints quaternions, and on Wiimote, which produces six acceleration values for the hand movement. In the first evaluation, a single person (performing several times) eight pointing gestures were used to train the system, with obtained recognition rate of 88%. In the second test, the system was trained with three repetitions of four gestures performed by seven participants. The recognition rate reached 75% and the system needed at least 2.3 seconds to recognize a gesture.

In their recognition of hand gestures and poses Yin and Davis [12] employed low-level skeleton features with velocity and acceleration, as well as the depth frame, to determine the hand shape. Each gesture phase (pre-stroke, stroke, and post-stroke) is represented by a HMM which allows the model to recognize gestures continuously and quite fast (0.3 seconds on average). Their study proves that richer feature vector increases the recognition speed.

Another work which exploited motion features for gesture recognition was conducted by Truong et al. [13]. Their model computes the whole gestures mid-level features based on Laban qualities, such as Shape and Effort [14]. Thus, each gesture is represented by a set of 81 features. For classification they use and compare SVM and Extra Trees. The model was tested on the 12 full body iconic and metaphoric gestures collected in the Microsoft Research Cambridge-12 (MSRC-12) database [15]. High level of performance was obtained, particularly with Extra Trees, showing the pertinence of Laban features for gesture recognition.

All gesture recognition systems have limits and none of them fit completely to our recognition constraints of dynamical coupling during gestural interaction between a human and a virtual character. HMM based recognition systems need a lot of training samples to obtain good recognition rates, while DTW underperforms with high within-gesture variance. For all systems, the number of recognizable gestures is limited. The gesture recognition is often based on upper body movements and constraint by the user's morphology. The recognition systems need to reach the end of gesture to obtain a good recognition rate and are not able to recognize gestures continuously in the data flow.

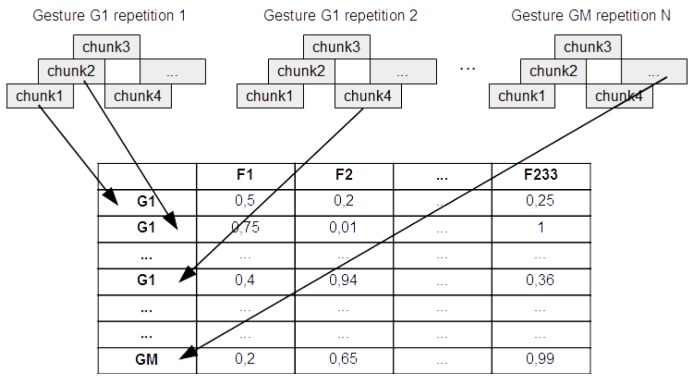
To our knowledge, Yin [12] and Truong [13] research is the closest system corresponding to our requirements. They show that taking different features into account improves speed recognition and robustness. However, Yin's system is based solely on upper body gestures and, even if it recognizes quite fast, it does not seem to take big variation of gesture expressivity into account. Truong's system computes features on the whole gesture, while we propose a use of sliding window in the learning phase, to capture features at different time of gesture execution. As explained in section III, this technique will allow our Analysis module to recognize a gesture fast and continuously, which is crucial in real-time interactive applications and one of our main challenge in human-virtual agent dynamical coupling.

III. Gesture recognition based on motion features

Gesture recognition and features analysis process is divided in two main parts: (i) an offline process in which the database of reference gestures is created and used to determine the most discriminant features able to differentiate the gestures, and (ii) an online process that runs during the user-agent interaction and that recognizes human's gestures and expressivity in real-time. For each gesture we collected ten repetitions performed by a single person. Indeed, one of our goals is to implement a module easy to train. Moreover, the person performs each repetition in very different ways to ensure gestures recognition is expressivity-independent. To assure expressivity variation in recordings, each gesture was performed twice in five different emotional states: neutral, happy, sad, stressed, and relaxed (note that gestures vary in expressivity more naturally when performed through emotional states than by asking the participant to perform gestures faster or slower, more or less fluid, etc.). Gestures are recorded as a sequence of unified 15-joints skeleton in 3D positions, which allows device-independent system, as explained in the introduction.

A. *Database creation and features selection*

The set of recorded gestures is divided in two subsets of the same size. Both subsets are used in the offline process - the first one is employed to create the reference database, which is used during the online process to recognize gestures in real-time; the second subset is utilized to create the support database, which is necessary to determine the minimal number of discriminant features. Both databases are created as follows. A sliding window of 60 frames is used to cut each repetition of a gesture in chunks with a shift of 30 frames. In each chunk, motion features are computed frame by frame and their mean, minimum, maximum, and standard deviation are calculated and stocked as a line of the database, together with the corresponding gesture class. This way each gesture is represented in the database by several lines, one line per gesture repetition chunk. Using several chunks for each repetition provides information about the gesture temporal evolution. Moreover, with a shift of only 30 frames, each chunk retains some information of



	F1	F2	...	F233
G1	0,5	0,2	...	0,25
G1	0,75	0,01	...	1
...	...	...	...	...
G1	0,4	0,94	...	0,36
...	...	...	...	...
...	...	...	...	...
GM	0,2	0,65	...	0,99

Fig. 2. Reference database creation: M gesture are recorded, each one repeated N times. Overlapping sliding windows of 60 frames cut gestures in chunks. Two hundred thirty three motion features are computed for each gesture chunk and recorded as a line of the database, together with the class name the gesture belongs to.

the previous chunk, avoiding any loss of potentially valuable information in between two frames.

To compute features, the 3D coordinates of the skeleton joints and the body position in the space (that corresponds to the root joint position) are employed. To assure independence of human's morphology and of input device coordinate system, all features are computed with normalized joints positions, expressed in a relative coordinate system. Hands and elbows joints are expressed in the coordinate system of the corresponding shoulder and normalized with respect to the arm length, while feet and knees joints are expressed in the coordinate system of the corresponding hip and normalized with respect to the leg length. The human position is considered to be X=0, Y=0, and Z=0 in order to be independent of input device coordinates.

Most of the features, such as body density, hands and feet symmetry, and head, hands and feet kinetic energy, fluidity, and directness, are computed following Piana and colleagues' work [5]. Other features, computed for head, and each arm and leg, are amplitude, speed, acceleration, velocity, and joints direction in the coordinate systems XY, YZ, and ZX. Finally, some arms and legs significant angles are computed: shoulder angle (between the trunk and the arm), elbow angle (between the arm and the forearm), leg angle (between the trunk and the thigh), and knee angle (between the thigh and the calf). In all, 233 features, normalized between 0 and 1, are computed. Database creation is shown in Figure 2.

All features are not necessary to discriminate gestures. To determine the minimal subset of discriminant features, PCA is computed. Features are considered to be PCA individuals, and gestures chunks are considered to be PCA variables. PCA reduces the original data space in a lower-dimensional space, transforming variables into new axes. From this new space, all the axes that together can represent 90% of the information are selected. Each individual (feature) has a value associated in this new space. All the couples *feature-value* are sorted in descending order, from the most important to the least important. It is used to determine the minimum features list, which is useful to discriminate database gestures. To deduce this list, several global theoretical recognition[1] rates, using the support database as the test set, are computed: the first recognition rate is obtained using only the first feature of the list, the second recognition rate is computed using the first two features of the list, the third one is obtained using the first three features of the list, and so on, until all the features of the list have been used. At the end a list of couples *number of features-recognition rate* is obtained and sorted in descending order. The first element (that is the best recognition rate) corresponds to the minimum features list which are needed to discriminate the gestures in the database.

A recognition rate is computed as follows. Each line of the support database (which corresponds to a gesture chunk) is compared with all the lines in the reference database to find the most similar one. To determine this similarity with the best possible reliability, three distance measures, based on different mathematics concepts, are computed:

- *Euclidean distance* – computes the sum of the distances between each couple of two n-dimension vectors (that is, two lines in the databases).
- *Cosine distance* – is the inverse of the similarity between two n-dimension vectors computed as the cosine of the angle between them.
- *Jaccard distance* – computes the dissimilarity between two sets.

The comparison between a gesture chunk in the support database and every gesture chunk in the reference database returns, for each distance measure, a list of couples *distance-gesture class* sorted in ascending order. This means that the first elements of the three lists correspond to the most similar gesture classes in the reference database. Since the module knows which was the real gesture class of the current chunk, it can determine which couples were correct and which were not, and consequently compute the recognition percentage as an average of correctly classified results. For example, if the current chunk in the support database belongs to a gesture in class G1 and if the comparison with all the lines in the reference database determines that the closest gestures class is G2 with Euclidean distance, G1 with Cosine distance, and G1 with Jaccard distance, then recognition rate is 66.6% (that is $(0\% + 100\% + 100\%)/3$).

Taking into account only the first element in the lists of couples *distance-gesture class* is not as good as it could seem. We noticed, in fact, that when a gesture chunk is hard to recognize, several different gesture classes appear among the first elements of the lists, which means that there is too much uncertainty in order to determine the gesture class of the current chunk. In this case, it would be a mistake to consider only the closest gesture class determined by each distance measure, that could provide a high recognition percentage for a gesture class when, in reality, the module is far from recognizing correctly the chunk. For such a reason we decided to take into account the first five most similar gestures classes for each distance measure. Thus, the chunk recognition percentage is computed five times: that is considering just the first element in the lists, then considering the first two, then the

[1]For theoretical recognition we mean that gestures are recognized separately and not in continuous flow.

first three, and so on. Each time the recognition percentage is a weighted average of the good results provided by the distance computation. It is weighted in the sense that the closest correct gesture class (or classes) is more important than the most distant ones.

The offline process computes the five recognition percentages for each gesture chunk in the support database and determines the five global recognition rates as the mean of all chunk recognition percentages. As explained before, it computes these values several times, using just the most discriminant feature to compute the distance with the gesture chunks in the reference database, then using two features, and so on. At the end, it selects the higher recognition rate and returns the list of features associated as the most discriminant set of features, as well as the number of the closest gesture classes that must be considered in order to avoid false recognition.

B. Real-time recognition

Once the reference database calculated, the minimal set of discriminant features selected and the number of closest gesture classes to consider determined, the recognition module is ready to recognize in real-time. The system, working 30 frames-per-second (fps), receives a skeleton of 15-joints representing the user who is moving in front of the tracking device. The skeleton joints are normalized and modified to be represented in the body coordinate system, as explained in section III-A. Then recognition algorithm computes and stores features in a history (that is an array of 60 elements). This history represents the sliding window used to compute all discriminant features. The sliding window shift is one frame, that is, once the history is full, each new features sequence replaces the oldest one. That means that the discriminant features are computed at each frame, which allows the system to recognize gestures continuously. At each frame, the subset of discriminant features is compared with all the lines in the reference database using the three distance measures, as described above, and recognition percentage is computed taking into account the number of closest gesture classes to consider, which have been determined in the offline process. The Analysis module decides that a gesture is recognized only if this gesture obtained a recognition percentage of 100%.

IV. Implementation choices

Some of the choices we made to implement the Analysis module could appear arbitrary: (i) we use a sliding window of 60 frames with a shift of 30 frames to create the reference database, (ii) we compute PCA on all gestures and not on each gesture separately, and (iii) we use the same sliding window and shift than in reference database to create the support database (utilized in the features selection process). In the following subsections these choices are clarified.

A. Reference database sliding window and shift

To determine the best size of the sliding window for cutting gestures into chunks and the best size of the window shift, a test was conducted: we compared the recognition rate obtained with different sliding window and shift sizes. To perform this evaluation, we collected 10 repetitions of 10 gestures as described before (in 5 different emotional states)

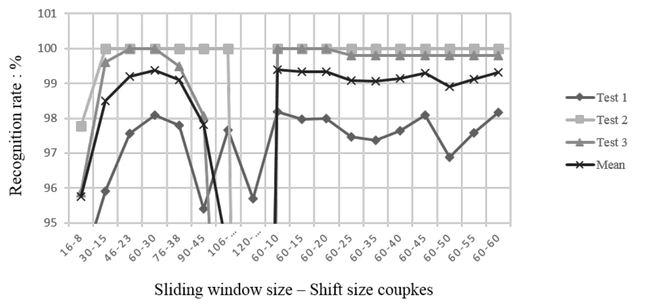


Fig. 3. Recognition rate for several window-shift couples

plus rest position[2]. We utilized five repetitions of each gesture to create the reference database and five for testing purpose. The recognition rates for the following couples of *sliding window size-shift size* were computed: 16-8, 30-15, 46-23, 60-30, 76-38, 90-45, 106-53, 120-60, and then 60-10, 60-15, 60-20, 60-25, 60-35, 60-40, 60-45, 60-50, 60-55, and 60-60; for each couple we conducted three tests:

- *Test 1* – The reference database and the recognition algorithm use the same couple of *sliding window size-shift size*. We perform a theoretical recognition for each gesture separately.
- *Test 2* – The reference database and the recognition algorithm use the same sliding window size, but a different shift size. The recognition algorithm shifts the sliding window of one frame. That is, we have a set of features that represents a chunk of the gesture at each single frame. Like in the first test, we perform a theoretical recognition for each gesture separately.
- *Test 3* – Similar to Test 2 but performed on a continuous flow of gestures. This test uses the whole module as it is actually used for real-time recognition.

Figure 3 presents the best recognition rates obtained in each test, as well as their mean, and only rates higher than 95% are considered. Best results are obtained for reference database created with couple of *sliding window size-shift size* 60-10, 60-30, 60-20, 60-15 and 60-60 (in descending order), and with a shift size of one frame for the recognition algorithm. In any case, best results are obtained with sliding windows of 60 frames. We chose a shift of 30 frames rather than 10 because it stores less information in the reference database (the third of the information that the couple 60-10 would generate). Indeed, we need to favor the smallest possible database in respect of real-time constraints and memory consumption.

B. PCA computed on the whole database

To determine the most discriminant features we utilized PCA and performed two tests. The same set of 10 gestures (+ rest position), described in the previous section, was used with the recognition algorithm on a continuous flow of gestures to recognize the most pertinent features obtained with PCA: first on the whole reference database (disregarding the gesture), and secondly on each class of gestures separately. A recognition

[2]Upright with the arms parallel to the body.

rate of 100% was reached in the first and 99,8% in the second test. The solution which leads to the best recognition rate was chosen.

C. Support database

To compute the minimum features list which can discriminate the gestures, we created a support database with a sliding window of 60 frames and a shift of 30 frames. Recognition rates would be more precise if we would have used a shift of a single frame, but computation would have increased significantly. For our 10 gestures (+ rest position), creating reference and support databases takes around 4 minutes, whatever the shift size. But features selection takes 12 minutes with a shift of 30 frames, and three hours and ten minutes with a shift of a single frame, with quite similar results. We favored the fastest solution because our system must respect the "training efforts" requirement described in section I, that is it must be trained easily and fast whenever new gestures need to be integrated.

V. Evaluation

Several tests were performed in order to evaluate the Analysis module performance. These evaluations answer the following questions:

- *Recognition delay* – How fast does the module recognize a gesture compared to a human?
- *Recognition plasticity* – Can the module recognize gestures performed by different people? Can the module recognize different gestures?
- *Recognition rate* – What is the module recognition rate?
- *Performance* – How many gestures can theoretically be added to the reference database without the computational process exceeding 33.33 ms (time between two consecutive skeletons in 30 fps)?

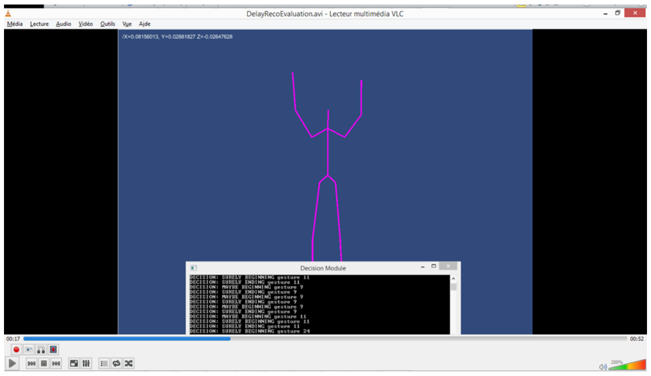
Implementations and evaluations were realized with a computer ASUS N56V Series with proc Intel(R) Core(TM) i7-3630QM 2.40 Ghz, RAM 6.00 Go, NVIDIA Geforce 740M.

A. Recognition delay

It is important that the recognition delay of the Analysis module is not longer than that shown by a human. If such a delay was longer, effective and satisfying human-agent interactions would be very hard to obtain, as the user would sense the unnatural movements/reactions of the virtual agent. An evaluation was made using three gestures (two of them start in a similar way and can be easily mixed up). We tracked with a Kinect a human performing these three gestures ten times, in a random order. The flow of skeletons is sent to the Analysis module which prints the name at the beginning and at the end of the gestures it recognizes. We created a video, as shown in Figure 4, which shows a virtual character performing these gestures as well as the Analysis module results.

The video is then annotated in three different ways:

- *Expert annotation* – an expert annotates the beginning (when the virtual character starts to move from rest position) and the end (when the virtual character comes back to the rest position) of each gesture. To be accurate, the expert can view the video frame-by-frame and rewind it whenever necessary.
- *Natural annotation* – twenty participants annotate the video. Obviously, the Analysis module printing is hidden. They have to annotate their own perception of the gestures beginning and end. They cannot rewind the video. They are only allowed to pause the video in order to write their annotation.
- *Analysis module results annotation* – an expert annotates gestures' beginning and end printed by the Analysis module. This is necessary to synchronize all the annotations done in the video.


DelayRecoEvaluation.avi - Lecteur multimédia VLC
Média
Lecture
Audio
Vidéo
Outils
Vue
Aide
/X=0.08156013, Y=0.02881827 Z=-0.02647628
Decision Module
00:17
00:52

```
DECISION: SURELY BEGINNING gesture 11
DECISION: SURELY ENDING gesture 11
DECISION: MAYBE BEGINNING gesture 9
DECISION: SURELY ENDING gesture 9
DECISION: MAYBE BEGINNING gesture 9
DECISION: SURELY ENDING gesture 9
DECISION: MAYBE BEGINNING gesture 9
DECISION: SURELY ENDING gesture 9
DECISION: MAYBE BEGINNING gesture 11
DECISION: SURELY BEGINNING gesture 11
DECISION: SURELY ENDING gesture 11
DECISION: SURELY BEGINNING gesture 24
```

Fig. 4. Gestures recognition video example

Annotations are made using the ELAN software [16].

1) Participants: Twenty students (19 men and 1 woman) aged from 20 to 23 (mean age: 21.25, SD: 1.04) were recruited at the Brest National Engineering School for the annotation purposes.

2) Experimental setting: An experimenter was present during all the evaluation to check if the participants obeyed the rules and to help them to use the annotation software if needed. Before showing the video to annotate, the experiment invited the participant to watch a video which showed the three gestures to annotate with their corresponding names. Since it was important that participants correctly memorized the three gestures, they were allowed to re-watch this video as many time as they wanted. Then, the experimenter taught them how to make an annotation and, finally, determined the participant reaction time. She showed them a video where the virtual character raised its right arm three times; the participants had to pause the video as soon as they could see the arm moving. Reaction times were subtracted to the participants recognition delay.

Eventually the test could begin. The experiment showed the participant the video to annotate, reminding them that they had to pause the video only when they recognized a gesture. At any moment, the participant could consult a reminder sheet where the name and the number of the gestures were written. For each gesture, they had to annotate bX for "beginning gesture X", and eX for "ending gesture X", X being the gesture number.

3) Analysis and results: To analyze the annotations, we considered that the expert annotation provided the real beginning and end time of each gesture. So, all delays were computed with respect to them. Firstly, we computed the average human recognition delay for the beginning and the end of each gesture. Then we compared them with our module recognition delay always for the beginning and the end of each gesture. We performed a Wilcoxon test using the Minitab software.

Figure 5 shows our module and the participants' average recognition delay for each gestures beginning. The average human delay is 0.48 seconds (SD: 0.04). The average module delay is 0.41 seconds (SD: 0.19). The Wikcoxon test indicates a significant difference between human recognition times and module recognition times. Our module detects faster gesture beginning than humans (Wilcoxon test=170, P<0.01, Median=0.49).

Figure 6 shows our module and the participants' average recognition delay for each gesture end. The average human delay is -0.29 seconds (SD: 0.23). The average module delay is 0.99 seconds (SD:0.45). The Wikcoxon test indicates a significant difference between human recognition delay and module recognition delay. Our module is slower than humans to detect the end of a gesture (Wilcoxon test=0, P<0.001, Median=0.1).

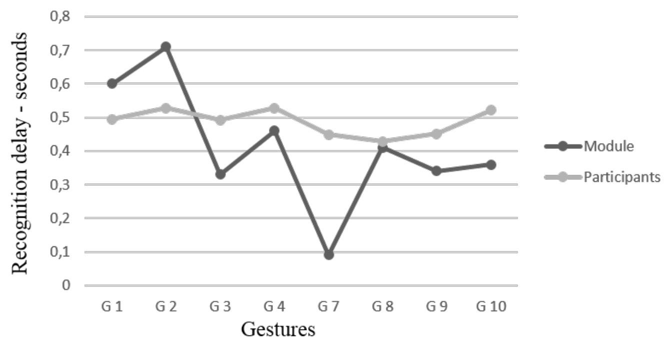


Fig. 5. Beginning detection delay in seconds. 0 corresponds to the expert annotation. Gestures G5 and G6 were deleted since mixed up by our module.

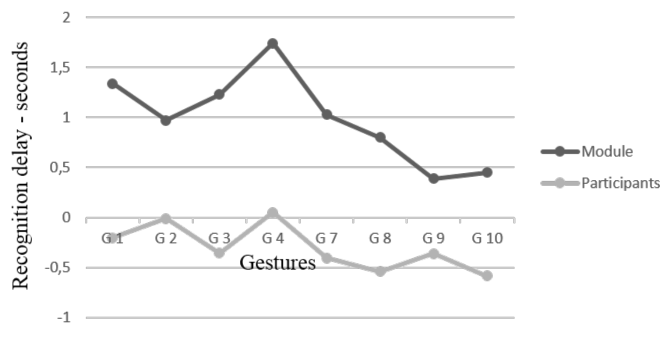


Fig. 6. End detection delay in seconds. 0 corresponds to the expert annotation. Gestures G5 and G6 were deleted since mixed up by our module.

4) Discussion: Gestures beginning are recognized faster by our module than by participants. However, analyzing Figure 5 and the standard deviation, we noticed that ther module recognition delay is quite irregular and depends on the type of the gesture. When gestures are quite different, our module recognizes them more easily and faster, so the recognition delay could be improved if dissimilar gestures were considered, consequently it could get worse if similar gestures were used. Thus, we consider that the Analysis module can recognize a gesture at the same time and speed as human, with possible hesitation if gestures are too similar. The end of a gesture is recognized faster by participants than by our module. Clearly, participants anticipated the end of the gesture. The expert annotator considered that a gesture was over when the virtual character went back to the rest position, while participants considered that a gesture ended when its stroke was finished. Concerning the Analysis module, it does not detect gesture end, it only detects that "something else" is beginning. We decided to interpret this change as the end of the previous gesture. Finally, it is important to notice that for people it is more difficult to detect the end of a gesture (SD: 0.23) than the beginning (SD: 0.04), which could indicate that the end of a gesture depends on the context.

B. Recognition plasticity

This evaluation aims at verifying if the Analysis module can recognize several gestures performed by people with different morphology. We remind that one of our module requirements is that training efforts should be limited, and that gestures collected using just a person should be enough to create a reference database which allows morphology-independent gesture recognition. We want also to show that the recognition rate is not correlated to the person who performed the gestures to train the system.

Ten gestures are used in this evaluation. Five of them come from the MSRC-12 Kinect gesture data set [15]: "Crouch/hide" (G16), "shoot with a pistol" (G17), "throw an object" (G18), "kick to attack an enemy" (G20), and "wind up the music" (G24). Other gestures are provided by the Magician scenario from Ingredible project: "say hello with the right hand" (G01), "magician bow" (G05), "show enthusiasm" (G07), "no big deal" (G09), and "incite to go faster" (G10). To test our module plasticity, we added following:

- Gesture "no big deal" is a chunk of "wind up the music" and "show enthusiasm" - it is supposed to be badly recognized because confusion between the both gestures is expected.
- Gesture "kick to attack enemy" is recorded as a perfect art martial kick (perhaps badly performed by inflexible people).
- Gesture "show enthusiasm" is recorded with wide amplitude and strong energy (perhaps badly performed by shy people).

1) Participants: Twenty participants with very different morphology were recruited. There were 10 men and 10 women, aged from 22 to 58 (mean age: 32.65, SD: 10.84). Their size varied from 1.53 meters to 1.83 meters (mean size: 1.68, SD: 0.1). Their weight varied from 44.2 kg to 114.8 kg (mean weight: 67.74, SD: 16.41). We computed each participant body mass index (BMI) which varied from 17.7 to 46.57 (mean BMI: 24.05, SD: 6.44).

The person who performed the gestures used to train the system is a 39 year old woman measuring 1.61 meters, weighting 51.2 kg, and with a BMI of 19.75.

2) Experimental setting: This evaluation was conducted in a room dedicated to gestures capture where each participant was located in front of the capture device (a Microsoft Kinect). The evaluation was performed in two steps:

- *Gestures learning* – Each gesture was showed to the participant, who could reproduce it several times in order to be at ease with it. Gestures order was different for each participant in order to avoid a bias due to gestures learning order.
- *Gestures performing* – The participant was asked to perform all ten gestures. He had to go back to rest position after each gesture and wait for five seconds. Gestures order was different for each participant. This session was recorded to be analyzed.

3) Analysis and results: Each recording session was provided to the Analysis module which printed frame by frame the name of the gesture it recognized (it printed "unknown" otherwise). A Mann-Whitney test was computed to determine whether there was a statistical difference concerning the results obtained by men and women. A Pearson correlation test was computed to determine whether age, size, weight, or BMI influenced the recognition results. Since the woman who performed the gestures to train the system participated also to the evaluation, these tests aimed to verify if the module is really morphology independent. Statistical tests were realized with the Minitab software.

Figure 7 shows, for each gesture, the number of participants whose gesture was correctly recognized. A Friedmann test is computed to verify whether recognition rate is different between gestures. There is a significant difference (Friedmann test: 71.75, $p<0.001$).

Figure 8 shows, for each participant, the number of gestures which were correctly recognized. The Mann-Whitney test indicates that there is no significant difference between results obtained by men and women ($p>0.05$). The Pearson correlation test indicates that there is no significant difference between results with respect to age (ρ: 0.334, $p>0.05$), size (ρ: 0.034, $p>0.05$), weight (ρ: 0.214, $p>0.05$), and BMI (ρ: 0.221, $p>0.05$).

4) Discussion: Results show that our module can recognize several gestures performed by people with different morphology. It confirms that we succeed in normalizing features, and in removing information related to the morphology of the person used to train the system. We deliberately recorded only 5 variations of each gesture to test our module. But, it is possible to improve recognition by recording more variations of the same gesture.

Results show, however, that some people were recognized better than other. A reason could be that participants had a brief introduction to gestures, and did not necessarily memorize correctly them. Sometimes, they did not make the correct gesture, like for the gesture G07 ("show enthusiasm'), or were not flexible enough, like for the gesture G20 ("kick"), which requires the leg to be horizontal, which was not realizable by

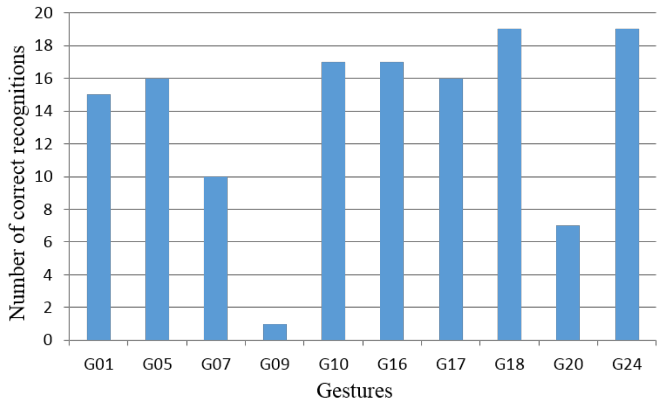


Fig. 7. Number of correct recognized gestures

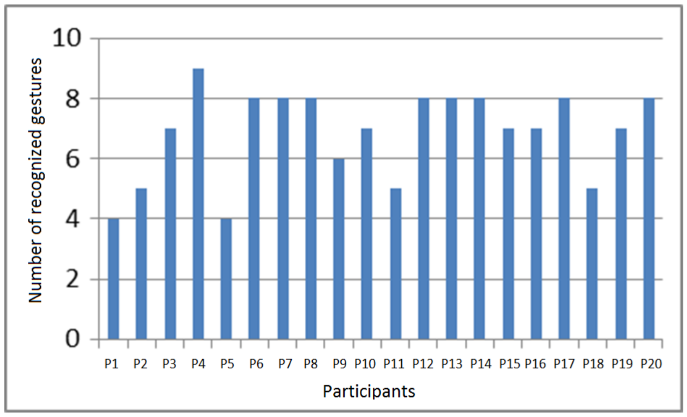


Fig. 8. Number of recognized gestures by participants

everyone. Concerning gesture G09 ("no big deal"), it was not recognized even when the person who recorded the database performed it. It confirms our hypothesis that a gesture which is also a trunk of others gestures is not well detected by our module.

C. Recognition rate

This evaluation aims at measuring our module raw recognition rate in a real-time context, which means that the module computes motion features for each received skeleton and indicates the gesture it recognizes every time the recognition percentage reaches 100%. This evaluation uses ten gestures, which are almost the same than in the evaluation described in section V-B. The gesture G09 ("no big deal')' is replaced by G08 ("applause')'.

Again, the ten gestures used to train our module were performed by a single person, we call her person A. Another person, person B, performed the gestures used for testing. Person B knew the ten gestures quite well, she was at ease to perform them. Person B did not have to purely copy gestures from person A, but has to perform gesture in her own way; for example, person A performed the gesture "shoot with a pistol" shooting just once, while person B shot three times.

Person B was recorded twice:

TABLE I. RECOGNITION ACCURACY

	Test 01	Test 02	Test 03
Total number of recognition	576	576	2175
Number of correct recognitions	575	575	2092
Number of incorrect recognitions	1	1	83
Accuracy	99.83%	99.83%	96.18%

TABLE II. RECOGNITION PRECISION AND RECALL

	Precision			**Recall**		
	Test 01	**Test 02**	**Test 03**	**Test 01**	**Test 02**	**Test 03**
G01	100%	100%	100%	100%	100%	100%
G05	100%	100%	100%	97%	97%	100%
G07	100%	100%	98%	100%	100%	100%
G08	100%	100%	94%	100%	100%	96%
G10	98%	98%	100%	100%	100%	77%
G11	100%	100%	93%	100%	100%	100%
G16	100%	100%	100%	100%	100%	100%
G17	100%	100%	100%	100%	100%	89%
G18	100%	100%	100%	100%	100%	100%
G20	100%	100%	100%	100%	100%	100%
G24	100%	100%	86%	100%	100%	100%

- *Gesture by gesture* – Each gesture was recorded in a file, starting and finishing with rest position. Within this session, we obtained 11 files (10 gestures + rest position, called G11).
- *Continuous flow* – The whole gestures were recorded in a single file. In this sample, gestures order was different than that in the precedent one.

With these records, we tested recognition rate three times:

- *Test 1* – The Analysis module receives the flow of skeletons deriving from the "Gesture by gesture" recording, in the same order in which the gestures were recorded.
- *Test 2* – Similar to Test 1, but the gesture order was different.
- *Test 3* – The Analysis module receives the flow of skeletons deriving from the "Continuous flow" recording.

1) Analysis and results: Comparing Test 1 to Test 2 determines whether the gestures order has an influence on the recognition rate, while comparing Test 1 to Test 3 shows the difference between pure gestures recognition and real-time gestures recognition. To compute recognition rate, we used the same measures proposed in [8]: accuracy, precision, and recall.

Table I shows details about recognition accuracy for the three tests. Table II shows precision and recall for each gesture. There is not difference between Test 1 and Test 2 results, which confirms that gestures are properly recorded independently in the database.

For the three tests, 100% of gestures were correctly recognized during stroke. The bad recognition always appeared between two gestures, during transitions.

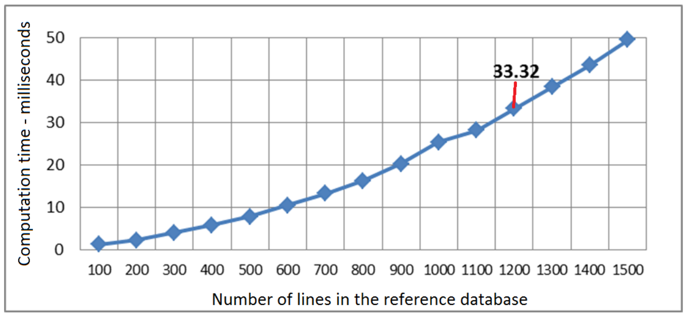


Fig. 9. Module recognition computation times according to database number of lines

2) Discussion: Results show a good recognition rate. In fact, all gestures were recognized during stroke, without errors. Errors appeared during transitions between two gestures. That was the only difference between Test 1 (gesture by gesture) and Test 3 (continuous flow). This result cannot be generalized because recognition rate depends on recorded gestures. If we record some gestures which have similar parts, the confusion level will increase, so will the error rate. On the other hand, if we record gestures which employ different parts of the body, or which are quite different, confusion level will decrease, and so will the error rate. The interpretation of these transitions depends on the Decision module according to the scenario.

D. Performance

The recognition process compares real-time features with all lines of the database. That means that the bigger the database, the longer the computation time. This is one of the limits of our module. This evaluation aims at measuring the maximum number of gestures which can be added in the reference database without computation time exceeding 33.33 ms. Our objective is to find the module's computation limit.

1) Results and discussion: As explained in section III, each line in the reference database represents a gesture chunk. The longer gestures to recognize, the bigger the database will be. Thus, this evaluation compares computation times according to the database number of lines. We use database with several number of lines: 100, 200, 300, and so on, until computation times reaches 33.33 ms. For each database, the Analysis module computes times between skeleton reception and gesture recognition. This time is computed for 100 skeletons, and the average time is stored in a file.

Figure 9 shows computation times in milliseconds according to the database number of lines. A computation time of 33.33 ms is reached when the database contains 1200 lines. Analyzing the gestures we collected for the evaluations described in the previous sections, we noticed that the average number of lines for a gesture was 37 lines. So, in theory we could collect up to 32 gestures to recognize before that the computation time becomes too long.

VI. CONCLUSION

The Analysis module, a component of the Ingredible project framework which recognizes continuous gestures performed by a user, was described in this paper. We propose to represent gestures as a set of motion features computed on a sliding window of 60 frames with 30 frames overlapping. Thus, a gesture is cut into chunks and, since sliding windows overlap, each chunk retains some information of the previous chunk.

During real-time recognition, motion features are computed at each received skeletons and they are compared to those in the reference database to determine the most similar class of gesture. This allows continuous recognition. Three different distance measures are used which allows us to determine when the module is 100% certain of its recognition and when confusion appears, which means that the module cannot provide an answer. In continuous real-time recognition this confusion appears mainly during transitions. The evaluations we conducted allowed us to provide answers to the questions we rose in section V and to show that most of the the requirements we described in section I are satisfied. Evaluations confirmed that the recognition delay is similar to that shown by humans, then our module is fast enough to be used in interactive applications.

Moreover, the Analysis module has been proven quite robust: it can recognize several gestures performed either with different expressivity or by people with different morphology. This is true even if the module was trained with just 5 very different repetitions of each gesture performed by a single person. Such a result satisfies our "training efforts" requirement.

The number of gestures in the database could increase till 32 without loosing real-time performances, however code optimization is still possible and we think that this number could be higher. Last but not least, the recognition rate in a continuous flow is quite good, when gestures are not too similar the recognition rate is 100% during the stroke. Gestural transitions are still a cause of confusion, however this information could be used by the Decision module to anticipate that something has changed in the user's behavior.

Acknowledgment

This work was funded by the ANR INGREDIBLE project: ANR-12-CORD-001. The authors would like to thank Marine Grandgeorge for help analyzing the results.

References

[1] J. K. Burgoon, L. A. Stern, and L. Dillman, *Interpersonal Adaptation: Dyadic Interaction Patterns*. Cambridge University Press, 1995.

[2] A. Fogel and A. Garvey, "Alive communication," *Infant Behavior and Development*, vol. 30, no. 2, pp. 251–257, 2007.

[3] P. De Loor, E. Bevacqua, I. Stanković, A. Maatallaoui, A. Nédélec, and C. Buche, "Utilisation de la notion de couplage pour la modélisation d'agents virtuels interactifs socialement présents," in *Conférence III*. France: Oxford University Press, 2014.

[4] S. Mitra and T. Acharya, "Gesture recognition: a survey," *IEEE Transactions on Systems, Man, and Cybernetics - Part C: Applications and Reviews*, vol. 37, no. 3, pp. 311–324, 2007.

[5] S. Piana, A. Staglianò, A. Camurri, and F. Odone, "A set of full-body movement features for emotion recognition to help children affected by autism spectrum condition," in *IDGEI International Workshop*, 2013.

[6] Y. Gu, H. Do, Y. Ou, and W. Sheng, "Human gesture recognition through a Kinect sensor ," in *2012 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, Guangzhou, China, 2012, pp. 1379–1384.

[7] S. Bhattacharya, B. Czejdo, and N. Perez, "Gesture Classification with Machine Learning using Kinect Sensor Data," in *2012 IEEE International Conference on Emerging Applications of Information Technology (EAIT)*, 2012, pp. 348–351.

[8] R. Ibañez, A. Soria, A. Teyseyre, and M. Campo, "Easy gesture recognition for kinect," *Advances in Engineering Software*, vol. 76, pp. 171–180, 2014.

[9] S. Sampena, S. Nur Ulfa Maulidevi, and M. Peb Ruswono Aryan, "Human action recognition using Dynamic Time Warping," in *2011 IEEE International Conference on Electrical Engineering and Informatics*, 2011.

[10] C. Waithayanon, C. Aporntewan, "A motion classifier for Microsoft Kinect," in *2011 6th International Conference on Computer Science and Convergence Information Technology (ICCIT)*, 2011, pp. 727–731.

[11] R. Billon, A. Nédélec, and J. Tisseau, "Gesture recognition in flow based on pca analysis using multiagent system," in *ACE '08: Proceedings of the 2008 International Conference on Advances in Computer Entertainment Technology*. New York, NY, USA: ACM, 2008, pp. 139–146.

[12] Y. Yin and R. Davis, "Real-time continuous gesture recognition for natural Human-Computer Interaction," in *2014 IEEE Synposium on Visual Languages and Human-Centric Computing (VL/HCC)*, 2014, pp. 113–120.

[13] A. Truong, H. Boujut, and T. ZAHARIA, "Laban descriptors for gesture recognition and emotional analysis," *The Visual Computer*, 2015.

[14] R. Laban, *Espace Dynamique*. Contredance, 2003.

[15] *Instructing People for Training Gestural Interactive Systems*. ACM Conference on Computer-Human Interaction, 2012. [Online]. Available: http://research.microsoft.com/apps/pubs/default.aspx?id=157326

[16] P. Winttenburg, H. Brugman, A. Russel, and A. Klassmann, "Elan: A professional framework for multimodality research," in *Proceedings of the 5th Langage Resources and Evaluation Conference*, 2006, pp. 1556–1559.