No sweat, no Fun: Large-Gesture Recognition for Computer Games

Luc Geurts^{1,2}, Andy Van Woensel¹, Vero Vanden Abeele^{1,3}

¹e-Media Lab, Group T – Leuven Engineering College, (Association KU Leuven)

A.Vesaliusstraat 13, 3000 Leuven, Belgium

luc.geurts@groepT.be

²ESAT, KU Leuven – Kasteelpark Arenberg 10, 3001 Heverlee, Belgium

³CUO, IBBT/KU Leuven – Parkstraat 45, 3000 Leuven, Belgium

ABSTRACT

Controlling computer games and other applications with motion-based gestures has become common in the last decade. In this paper we present a method to recognize relatively complex gestures with a simple, fast and yet reliable classifier algorithm. More specifically, the purpose of the classifier is to recognize large gestures, as measured with the accelerometers and gyro sensors present in the Wii remote controller. The classifier is used and tested in the context of a computer game that forces the player to make large movements. The outcomes confirm that only large gestures are recognized correctly, leading to successful play. Therefore, the classifier might be particularly useful for designers of exertion games.

Author Keywords

Computer games, Game controllers, Gesture recognition

ACM Classification Keywords

H.5.2. User Interfaces

General Terms

Algorithms, Design, Measurement

INTRODUCTION

The use of motion-based gestures has become a popular control method for computer games, supported by the three big console manufacturers all having gesture based controllers at this time. In order to recognize these gestures, several algorithms exist, varying in accuracy, speed and complexity. An example of a simple algorithm is thresholding: an action is performed if certain sensor data exceed a preset value [3]. Such algorithms offer a low computational cost and a fast response, but also a lower accuracy. Simple algorithms might indeed classify a broad range of gestures as correct. Specifically problematic in a game context is the case where the player can fool the

system by performing the gesture in a way that was not intended by the designer. This can hamper the overall user experience, especially when aiming at physical exercise or exertion [7]. Not surprisingly, well-known among gamers are the mocking videos on the internet where popular ‘gesture’ games are played while barely moving. This inspired us to design a computer game where making large gestures are a prerequisite.

Examples of more complex pattern recognition systems are based on Hidden Markov Models (HMM) [5,9] or Dynamic Time Warping (DTW) [1,4]. While these pattern recognition systems might show high accuracy, these methods are also computationally intensive and might therefore be too slow to be perceived as real-time. This delay can cause a suboptimal experience for a large group of applications that depend on a swift response from the application. For real-time interaction, we need a gesture recognition system that works faster than the human perception threshold of 100 ms [6], while maintaining a high accuracy of detection.

In this paper we propose a fast method to recognize relatively complex (and large) gestures, based on a minimum distance classifier. Basically, the system extracts a limited set of features from the sensor data, and compares these with the features of the standard gestures to be recognized. The decision is based on the closest match between both. This approach is comparable to the L^2 -norm, as used e.g. by Slyper and Hodgins [11]. However, their system might be sub-optimal in efficiency, since all features were included in the comparison set. We propose a method to select a minimum amount of features, while still maintaining a high accuracy. Shiratori and Hodgins used accelerometers attached to the limbs in order to control a virtual character [10]. Their solution involved a continuous tracking of the user’s movements, and also recognizes these based on a limited set of features. However, these features were very specific for the application at hand. We propose a more generic system in which the most relevant features are selected based on gesture set. Finally, while the above papers report on the use of accelerometers for motion gesture recognition, we also include data from gyros sensors, which measure rotation speed along a certain axis.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

Fun and Games' 2012, Toulouse, France — September 4-6, 2012

Copyright © 2012 ACM ISBN 978-1-4503-1570-8 /12/09... \$10.00

Figure 1 The five steps for the design of a classification system (adopted from [12])

This way more information on the movements of the player is available for analysis.

In the first part of the paper, we will elucidate the five steps for designing the classifier [12] (see Figure 1). This classifier will be used in the specific case of a Tetris-based exertion game, i.e. a Tetris game that needs to be played via large gestures with the Wii remote controller. In the second part of the paper, we will validate the classifier via an experimental evaluation of the Tetris game. We asked eight experienced gamers to play the game making correct (large) gestures and next to find cheats and play without making these large gestures. The classifier is designed to recognize only large gestures correctly, so we hypothesize that the classifier will make more errors in the second case, and that the game scores will be lower.

THE FIVE STEPS FOR THE DESIGN OF THE CLASSIFIER SYSTEM

The design of the classifier system involves several steps, as depicted in Figure 1. We will now discuss how we designed our classifier system for the Tetris game alongside the five steps.

Step 1 Gesture definition

In the first step, it is defined which gestures are to be recognized by the classifier system. The player has to perform the gestures holding the Wii remote controller in her or his dominant hand while playing. Each gesture is associated with an action in the game. We adopted several criteria for a proper choice of gestures: 1) the gesture should have a clear link with the action to be performed (natural mapping); 2) the gesture should evoke large exaggerated movements of the player (fun factor); and 3) the gestures should be distinguishable (limited similarity). We defined a set of eight gestures according to the above criteria. A description of these gestures and the according actions in the game are listed in Table 1.

Step 2 Feature extraction

The Feature extraction step encompasses the collection of data that result from making the gestures and the derivation of features that typify the gesture. The collected data are the sample values generated by the sensors in the Wii remote controller, including the Wii MotionPlus extension. This device measures acceleration in three dimensions and rotation speed along three axes. The sample values are read out every 10 ms by a PC using a Bluetooth connection [2]. The beginning and the end of the gesture are indicated by the user by pushing the B-button. Because of the natural affordance of the Wii remote controller, this action will

barely require mental effort from the player. The initial feature set consists of several statistical parameters of the recorded data (see Steps 4 and 5).

Gesture	Action
Sway arm horizontally from left to right	Move current piece 1 grid-position to the right
Sway arm horizontally from right to left	Move current piece 1 grid-position to the left
Circle arm clockwise in a vertical plane	Rotate current piece, clockwise
Circle arm counter-clockwise in a vertical plane	Rotate current piece, counterclockwise
Pirouette 360° to the right	Move current piece completely to the right in the playing field
Pirouette 360° to the left	Move current piece completely to the left in the playing field
Sway arm vertically from high to low position	Drop the current piece to the bottom of the playing field
Stab arm forward from bended to stretched	Store the current piece in the queue and continue with the piece that was in that queue

Table 1 Each gesture triggers an action in the Tetris game.

Step 3 Classifier design

In the next step, a method is selected to estimate the executed gesture from the feature set. For the sake of simplicity, and thus processing speed, we opted for a minimal distance classifier.

For each gesture, there is a reference point in an n -dimensional feature space, where n is the number of features in the feature vectors (combinations of multiple features). When a gesture is executed by the user, the feature vector x for that input is calculated. Then the distance is calculated between that input feature vector x and each feature vector of the reference gestures. The input gesture is then labeled as the gesture that resulted in the lowest distance between the input vector and the defined vector. In order to take into account the differences in variance and range both between different features and between different gestures, the Mahalanobis distance is used [12].

In order to define the reference gestures, data were collected during a user test. A group of six users was asked to perform all eight gestures ten times. This resulted in 60 iterations per gesture, 480 iterations in total. The initial demonstration was important to make sure we could train our classifier with data belonging to correctly performed gestures. The total of 480 iterations was now divided into two subsets: a training set and a test set. The odd-numbered iterations were extracted for the training set. This set was used to calculate the predefined feature vectors per gesture. The value for each feature in the predefined feature vectors was calculated by taking the average of the 30 values of the corresponding feature and gesture in the training set. The even-numbered iterations were extracted for the test set.

Steps 4 and 5 Feature selection and Classifier evaluation

In this stage, the most relevant features are selected, based on the result of the evaluation of the classifier. The purpose is to reduce the number of features required for the classification, in order to reduce the required processing power and time, whilst still maintaining a low classification error rate. The target error rate was set at 5% or lower – or alternatively – 95% or more (correctly performed) gestures should be correctly classified. In order to find a good combination of features, a selection method was used that is a mix of a sequential forward search and a floating search [12]. The target 95% success rate was reached for several vectors with 3 features. The search was still expanded to 4-feature vectors to see how close to 100% success rate the classifier could get and to find a feature vector that included at least one accelerometer feature, so that the classifier would take translations into account. With these translations included, users would be less likely to be able to cheat the system by not performing the gestures with the intended exaggeration. From the feature selection step, the highest success rate was achieved for the following four features: the sum of the yaw data, the sum of the roll data, the minimum of the cumulative sum of the pitch data, and the sum of the acceleration along the z-axis. These features correspond to the following physical factors: final yaw and roll angles from the starting orientation, minimum pitch angle from the starting orientation and velocity along the z-axis. When evaluating the test set with this feature vector, a classification success rate of 99.5% was attained.

In order to evaluate the speed of the system (running on a standard PC with Windows OS), we recorded the time between the release of the B-button (end of the gesture) and the moment the gesture was classified. The average speed across all gestures is 3.4 ms, with a maximum of 17 ms. This value is linked to the pirouette gesture, which is a relatively long gesture. The large sample size requires more calculation time. So, all time intervals are well below the aforementioned threshold of 100 ms.

VALIDATION OF THE CLASSIFIER

Rationale and hypothesis

The classifier obtained a very high recognition rate (99.5%) on the test set, i.e. in an isolated setting. This high score is also due to the fact that the test set resembled the training set to a high extent: the same users made the same gestures at the same moment. In order to validate the classifier in a more natural setting, it was embedded in a computer game, and played by other users than those who generated the training set and the test set. The game to be played is a variant on the well-known Tetris game (provided in a Qt-version known as Tetrix [8]). Blocks are handled by making (large) gestures as explained before. The first goal is to evaluate the performance of the classifier if large gestures are made. Note however that the classifier will always make a decision when the B-button is released, also e.g. when the Wii remote controller is held still, or when a totally

different gesture is made. Lousily performed gestures should not be recognized correctly, resulting in an action that is not expected by the user and consequently, a worse score on the game. So, the second goal of the experiments is to evaluate the performance of the classifier if the players put less physical effort in their motions.

Methods

A within-subject design ($n=8$) was adopted to compare both conditions: large gestures (condition 1) vs. small ones (condition 2). All participants were experienced gamers that had played Tetris before, as well as games with the Wii remote controller. At the start of the experiment, it was explained how the gestures had to be performed. Next, the subjects had the opportunity to practice for a few minutes. This was enough to learn how to make the gestures correctly. During the first test, the participants were instructed to try to achieve a high score, while performing the gestures as demonstrated; so, with large movements. They played the game for two minutes and were also asked to indicate verbally whether the computer recognized their gesture correctly (e.g. by saying ‘good’ or ‘bad’). Besides the score, the number of completed lines, the number of successful gestures and failed gestures, also the heart rate was monitored in order to evaluate the physical effort they put in playing the game.

The purpose of the second test is to examine to what extent the game can be played successfully if the player puts less physical effort in her or his motions. Therefore, during the second test the participants were asked to make as small movements as possible, while still trying to obtain a high score. Everything else was kept identical, and the same data were recorded as in the first test.

Results

Across all participants, the average recognition rate, i.e. the ratio of successful gestures versus total number of gestures dropped from 92% (SD 8.1) for the first test to 63% (SD 16.1) for the second test (see Figure 2). This is a significant decrease ($t(7)=5.07$, $p<0.01$). The average score also dropped significantly from 319 (SD = 101.6) to 164 (SD=95) ($t(7)=3.36$, $p<0.05$). The average heart rate dropped from 111 (SD = 15) to 98 (SD = 13), again a significant decrease ($t(7)=4.44$, $p<.01$). The latter result is an indication that the players indeed put less physical effort in the game. The decrease in heart rate was significantly correlated with the decrease in score ($r=.674$, p (one-tailed) $<.05$).

After the tests, some participants reported that moving the block one grid position to the left or the right was possible with a smaller movement, as long as there was enough rotation in the movement. For circling and pirouetting no good alternatives were found requiring less effort. Several users also reported that their gesture was often confused with a “stab forward”. This can be explained by the fact that for this gesture the least movement is required, and so other “small” gestures are likely to be judged as a “stab forward”.

Figure 2 Results of the two tests: large movements (test 1) vs. small movements (test 2). Both the individual results (connected squares) as the average results (bars) are shown for: (a) the percentage successful gestures, (b) the scores on the game, and (c) the average heart rate during the game.

DISCUSSION

The results confirm the claim that the used classifier system only recognizes the gestures correctly if they are large. When the players tried to play the game making smaller movements, the recognition rate dropped significantly and a lower score was obtained. Perhaps this result is not what one would expect from a typical classifier. Indeed, in many applications a classifier has to estimate the intention of the user. For example in speech recognition systems: If a user says a word, the classifier should recognize the word, even if it is pronounced unclear or with an accent.

On the other hand, the approach of designing a classifier that does not try to estimate the intention of the user, but that even penalizes the user if her or his input is not conform certain requirements seems to work well in a game context. Note that this penalty is not the result of an extension of the classifier to judge the quality of the gesture, but simply the result of an error of the classifier, or its inability to estimate the user's intention. This supports the claim that less accurate classifiers can be used in a game context, as long as an integral approach is adopted by a proper choice of gestures and game mechanics. In our case, the inclusion of the "Stab forward" gesture turned out to be a good choice, since many small gestures are confused with this one.

The advantages of the presented classifier system are its accuracy, as long as the gestures are executed correctly, its speed, offering real-time experience, and its low computational cost, enabling applications in low-cost, low-power and/or wearable computing systems.

CONCLUSION

We presented a motion-gesture recognition system to be used in computer games. With a relatively low computational cost, a fast system is obtained that is able to recognize relatively complex gestures. We believe this

finding is especially useful for developers of exertion games or physical therapy games. As opposed to common classifiers, whose quality is determined by their ability to estimate the user's intention, we propose that "unaccurate" classifiers that only yield a high recognition rate when the gestures are performed correctly can actually be beneficial. This was illustrated via the Tetris exertion game where large gestures are needed to play a computer game successfully.

REFERENCES

1. Ashbrook, D. and Starner, T. MAGIC: a motion gesture design tool. *Proceedings of the 28th international conference on Human factors in computing systems*, ACM (2010), 2159–2168.
2. GL.tter. Wii yourself, Native C++ Wiimote library. <http://wiioyourself.gl.tter.org/>.
3. Hartmann, B., Abdulla, L., Mittal, M., and Klemmer, S.R. Authoring sensor-based interactions by demonstration with direct manipulation and pattern recognition. *CHI 2007 Proceedings*, ACM Press (2007), 145–154.
4. Holt, G.A. ten, Reinders, M.J.T., and Hendriks, E.A. Multi-Dimensional Dynamic Time Warping for Gesture Recognition. *proceedings ASCI07*, (2007).
5. Kela, J., Korpipää, P., Mäntyjärvi, J., et al. Accelerometer-based gesture control for a design environment. *Personal Ubiquitous Comput.* 10, 5 (2006), 285–299.
6. Moran, T.P., Newell, A., and Card, S.K. The model human processor: an engineering model for human performance. *Handbook of perception and human performance* 2, (1986), 41–45.
7. Mueller, F. "Floyd," Gibbs, M.R., and Vetere, F. Taxonomy of exertion games. *Proceedings of the 20th Australasian Conference on Computer-Human Interaction: Designing for Habitus and Habitat*, ACM (2008), 263–266.
8. Nokia Corporation. A qt version of the classic tetris game. 2011. <http://doc.qt.nokia.com/4.7/widgets-tetrix.html>.
9. Schlömer, T., Poppinga, B., Henze, N., and Boll, S. Gesture recognition with a Wii controller. *Proceedings of the 2nd international conference on Tangible and embedded interaction*, ACM (2008), 11–14.
10. Shiratori, T. and Hodgins, J.K. Accelerometer-based user interfaces for the control of a physically simulated character. *ACM SIGGRAPH Asia 2008 papers*, ACM (2008), 123:1–123:9.
11. Slyper, R. and Hodgins, J.K. Action capture with accelerometers. *Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, Eurographics Association (2008), 193–199.
12. Theodoridis, S. and Koutroumbas, K. *Pattern Recognition, Fourth Edition*. Academic Press, 2008.