KU LEUVEN

FACULTY OF ENGINEERING TECHNOLOGY

CAMPUS GROUP T LEUVEN

Design and implementation of an application for controlling computer games using a Kinect camera as part of physical therapy

Christiaan VANBERGEN

Dimitri VARGEMIDIS

Promotor: prof. dr. ir. L. Geurts

Co-promotor: ir. K. Vanderloock

Master Thesis submitted to obtain the degree of Master of Science in Engineering Technology: Electronics-ICT, Internet computing

Academic Year 2016 - 2017

Acknowledgements

Designing and implementing a user interface utilizing the relatively new Kinect camera is a challenging, albeit fascinating subject. We encountered many problems along the way, for which we enjoyed coming up with creative solutions. In the process, we were supported by many people we definitely would like to thank.

This thesis would not have been possible to write without our supervisor professor Luc Geurts and co-supervisor Karen Vanderloock from E-media lab, who provided us with this subject, invested their time and shared their knowledge and experience with us, allowing us to come up with solutions that have lead to a fully functional result.

During the year, we've had several occasions to discuss the subject of machine learning and support vector machine (SVM) with our fellow students Victor Martens and Alexander Kerckhofs, who helped with making us aware of the pitfalls and shortcomings of the used techniques and joined us to discuss the requirements and come up with a creative solution concerning the use of SVM.

User testing is very important when designing an application with a specific audience in mind. As such, we are grateful for the useful feedback of physical therapist Dries Lamberts of Windekind, who took the time to meet us twice and evaluate both our prototype as well as the final result. But even more importantly, he shared with us his passion for taking care of children with disabilities, their need for physical exercise and the importance of making sure this is both meaningful and entertaining.

Last but not least, we would also like to thank our parents, family and friends, who not only have supported us during the course of past year while working on this master's thesis, but also for trying out our application and giving valuable feedback.

Abstract

- Motivating children to perform physical therapy exercises.
- Exercises are adjusted to the needs of the child.
- Economic necessity to support a multitude of existing games.
- SVM to learn new exercises.
- Graphical user interface focused around use by the therapist.
- Results

Key words: exergames, gesture recognition, Kinect, support vector machine, user interface

Contents

Chapter 1

Introduction

There are many different reasons for requiring physical therapy. They vary from recovering from a car crash to being born with physical disabilities. The goal of this therapy is respectively to make sure that the patient can recover as much as possible from his injuries or to train the muscles, preventing their situation from deteriorating.

Especially for children, the physical exercises performed during therapy can be demotivating. Combining these exercises with playing computer games leads to an increased willingness to continue with the exercises. However, the classic approach related to games that incorporate physical exercises is not economically sustainable.

The purpose of this thesis is to present a more sustainable solution to this problem in a way that offers more flexibility to the therapist, both in terms of the exercises that need to be done by the patients and the games they can control and play by performing the exercises.

1.1 Discussion of the problem

Patients often see exercises as a part of physical therapy as being fatiguing, monotonous and tedious. This problem is even more prominent with young patients and can quickly demotivate them, especially when the exercises are uncomfortable or painful to perform.

Computer games already exist that can support physical therapies. For instance, there are games that run on Microsoft's XBox console and accept user input through the Kinect 3D camera, or Nintendo's Wii console that use a controller with accelerometers. Additionally, there are also games that can run on a computer, using any combination of sensors for user input, and are specifically made for use by physical therapists and their patients. However, all these games have in common that they are very static by nature and are not often suited for the specific needs of a patient. For instance, a game that focuses on arm movement is not very effective at exercising the leg muscles of a patient. In other words, concerning the therapy, the game is meaningless for a patient that had leg surgery.

In addition, these static games have fixed input gestures the patients have to perform in order to control the action on screen. Even if the gestures perfectly fit the exercise requirements for a specific patient, the decrease in attractiveness of playing the same game over and over again is problematic and loses its long-term effect. Physical therapist Dries Lamberts has experience with children with disabilities and uses computer games as a form of exercise. The initial reaction of the children to these games is positive. They are more engaged in doing physical exercises and feel motivated while doing so. On the downside, this effect only lasts until the novelty of the game wears off. After that, the children lose interest in the game and, as such, in performing the exercises.

As games are expensive to develop, and motion-based games in particular, there is no wide variety of games that are tailor made for people with specific needs and at the same time offer varied gameplay mechanics to remain interesting over long periods of time. On the other hand, producing these kind of games is not economically sustainable. From the patient's point of view, the right type of exercises need to be supported by the game and the game itself has to be considered fun as well by the patients for them to keep invested in it.

1.2 Purpose of the research

This thesis focuses on an application that allows for a more flexible and dynamic solution to the above discussed problem. It lets the physical therapist choose what exercises the patient has to do and how these can be used as an input to control any computer game of choice. All of this can be done without requiring the therapist to have any programming knowledge.

The goal is to have an application the therapist can control mainly using the Kinect camera. It needs to be both simple and efficient to do, in order to minimize the setup time before a patient can start playing. It is necessary to research the type of interface that is required to meet these objectives, in addition to finding out the easiest way to interact with an on-screen application using a camera for input.

Chapter 2

Literature study

2.1 Human-computer interaction

- Papers/artikels over framework, natuurlijke interfaces,...
- ...

2.2 Similar research

- Vergelijking van opzet met andere oplossingen
- Te trekken lessen uit deze oplossingen (vb: feedback voor de kinesist is belangrijk, opstellen van het systeem mag niet veel tijd in beslag nemen,...)
- ...

Chapter 3

Design

The properties of the application are chosen in function of solving the problem discussed in the introduction. The focus is on simplifying the setup process for the therapist, while still providing all tools needed for the patients to play a game using gesture input.

The developed application can roughly be split up into three major parts. Firstly, the graphical user interface allows the therapist to interact with the application, providing him with feedback and feedforward on inputting exercises for the patients. Secondly, gesture recognition is done as part of machine learning using support vector machines (SVM). Thirdly, all other back-end software connects the first two parts and provides a structure in which all data is managed and stored.

3.1 Properties of the application

The developed application can be seen as the link between the physical exercises and a game.

The physical therapist comes up with exercises that fit the needs of a patient. He uses the Kinect camera to interact with the application via a graphical user interface (GUI). Next, the therapist lets the application record what exercises need to be done by the patient. Using all of the recorded exercises, an SVM model is created, which is used to evaluate what exercise is performed by the patient while playing a game.

The therapist assigns a keyboard button to each of the exercises. This means that when the patient mimics one of the therapist's exercises, a keyboard button is pressed. If this application is running in the background while a computer game is opened in an active window, performing an exercise and thus indirectly pressing a button interacts with the game.

INCLUDE FIGURE SHOWING A GENERAL OVERVIEW OF
THE APPLICATION STRUCTURE IN BROADER CONTEXT

By mapping exercises to keyboard buttons, a vast majority of available games can be played using gesture-based input. It is however limited to button presses, pointing with the cursor like when using a mouse is not supported.

Application is for Windows only, C++

The application has to be flexible enough to support exercises that are unknown beforehand; same with the games played.

- Op voorhand ongekende oefeningen
- Op voorhand ongekende game
- Systeem = link tussen oefening en game
- ...

3.2 Graphical user interface

- Klassendiagramma voor GUI
- Algemene/schematische beschrijving van de voorgestelde oplossing.
- Procesbeschrijving, opties tussen verschillende types van interfaces (mime/dirigent)
- (Experimentele manier om tot prototype te komen door gesprek met kinesist (?))
- GUI + bespreking
- ...

3.3 Back-end software

- Klassendiagramma van de code (niet voor GUI)
- Uitleg structuur
- ...

3.4 Gesture recognition

3.4.1 Support vector machine

In order to provide enough flexibility concerning the type of exercises, SVM is used. SVM supports supervised machine learning and its use encompasses two modes: train and predict.

A model can be trained with a given set of data and a label to classify the gesture. The amount of numbers in one data set is referred to as the number of features. If it contains n features, the entire data set can be seen as a single point in a n-dimensional space. It is possible to train the same gesture more than once. In that case, the same label is used to indicate that the given data set is related to the same gesture. In other words, all trainings of the same gesture are linked to the same label. All of these trained gestures with the same label are seemingly similar, but actually contain variations due to noise during the measurement of the gesture or a slightly varying execution of the gesture.

After training all required gestures, a SVM model is created. Given a data set of a gesture, this model can predict which of the trained gestures has the biggest resemblance to the given data set. As a result, the label of the most similar trained gesture is returned. This also explains the necessity of having multiple trainings recorded for each gesture. Errors in a single training due to noisy measurements of the Kinect camera can lead to wrong predictions. Having multiple trainings minimizes the influence noise has on the prediction and keeps into account that the user executes all gestures with slight variations. As a result, the model can predict more accurately which gesture is performed.

However, there are some things to keep in mind when applying this strategy. Firstly, while inputting multiple repeats of the same gesture helps with predicting the gesture after a model is created, it takes more time for the therapist to do this. Secondly, when trying to predict a gesture, the generated SVM model always returns the label of the most similar gesture, even if they are not related at all.

These problems are tackled as part of the approach to using SVM to predict gestures.

3.4.2 Approach

As stated in section 3.3, a gesture consists of multiple frames. Each frame contains 25 joints and each joint consists of an x, y and z component. This results in a total of 75 features that are being considered for each frame.

Two approaches are considered when it comes down to learning how to recognize gestures. By directly comparing these approaches, it is easier to identify their advantages and disadvantages.

The first approach is to add a time stamp to each frame as an extra feature, which indicates the time relative to the first frame of the gesture. This results in having 76 features per frame. If performing a certain gesture takes about t seconds and is captured at a rate of f frames per second, this amounts to 76 tf features. This additional feature can be used to make sure that the gesture isn't just similar in space, but also in time. For a 3-second gesture recorded at 30 frames per second, which is the highest sampling speed of the Kinect camera, this amounts to 6840 features for a single training of the gesture. In other words, the number of features of one training depends on the duration of the gesture. The entire gesture is classified as a single gesture. During prediction, a gesture with the same number of features can be input to verify if that gesture matches any of the trained gestures.

There are several problems with this first approach. If the number of features of the gesture used for predicting does not match the number of features of the training gestures, the prediction is not accurate and should be discarded. Discarding is necessary as SVM always returns the label of a gesture, even if the predicted gesture is completely unrelated to any of the trained gestures. This poses a problem as different gestures can have a different duration. Even different trainings of the same gesture can take for instance 3 seconds and 3.1 seconds, implying that the recordings have a different number of frames and thus a different number of features. A possible solution is to recalculate the entire gesture and use interpolation to convert the set of frames to a new set with a known, fixed size and preferably with equidistant time stamps.

Furthermore, not just the trainings of one gesture, but all of the gestures need to have the same number of frames. The gesture to predict is not known beforehand and can only be predicted accurately if the number of features for both the predicted gesture and trained gesture are equal. As a result, it is required that all gestures have an equal amount of features. This means that short gestures need to be mathematically extended with additional frames to match the size of longer gestures. Another side effect is that all gestures need to have the same duration, not just the same amount of features. This limits the therapist in choosing exactly what exercises he wants the patient to perform. Also, predicting a gesture can only be as fast as the trained gesture with the longest duration. More in particular, assume the longest gesture is 5 seconds in length. It then follows that it takes about 5 seconds to predict any gesture. This also means that patients that control the game can only perform one action every 5 seconds. As the games to be played are unknown beforehand, a slow reaction time of the application can render some games unplayable. As such, this approach is not viable.

The second approach is to split up one gesture into n smaller gestures and classify each of them differently, assigning a different label to each part of the gesture. Assume that a gesture is split up into 4 smaller parts so that each part contains approximately an equal amount of frames. Consider a 4-second gesture sampled at 30 frames per second. The complete gesture consists of 120 frames. By splitting it up into 4 parts, each part contains 30 frames. All first 30 frames are labeled with the same label, for instance 1. The next 30 frames receive a label 2, and so on. To put it differently, the considered gesture is split up into 4 postures with each having 30 slightly varying trainings. In contrast to the first approach, as described above, prediction does not happen for an entire gesture, but for separate postures. The condition for having executed the entire gesture is that each of the postures are predicted correctly and in the right order. To put it differently, n *pictures* are taken of the gesture and if at some point during prediction all pictures are executed in the right order, the application acknowledges the execution of the gesture.

AFBEELDING NODIG

The biggest advantage of this approach is the flexibility of it. Each posture corresponds to a single frame, which contains 75 features. This eliminates the need of having gestures with the same length or having to modify training data to have gestures with an equal number of features. It is even possible to not just link a gesture, but also a posture to a keyboard button, which allows the physical therapist to choose the exercises that fit the needs of a patient the best. Additionally, the application reacts immediately to an executed gesture, not after a fixed amount of time. This allows for a wider variety of games being playable using this application.

Another advantage is that it solves the issue where SVM always returns the label of a predicted gesture, even when the patient is not doing anything. A gesture is only recognized when all parts of it are executed. In other words, gestures are not recognized involuntarily and, as such, actions like button presses that influence the game are not executed by accident. This also means that no neutral or *wrong* gestures are required to link a certain gesture to no in-game action.

The disadvantage of this second approach is that there is no strict requirement for the entire gesture to be executed in about the same time as the gesture recorded during training. If a gesture is executed faster during prediction compared to during training, it is recognized as the same gesture being executed. However, it is possible to solve this timing issue to some extent when the gesture during predicting is performed slower. The gesture can be ignored if more than a certain amount of time passes. This time threshold takes the gesture with the longest duration into account and is chosen higher than this in order to allow all gestures to be executed and successfully recognized.

Chapter 4

Implementation

- Bespreking van werking van belangrijkste softwareonderdelen
- Belangrijkste deel/delen uit code
- ...

4.1 Requirements and setup

List of requirements and setup due do design and implementation decisions.

These are Kinect requirements:

64-bit (x64) processor
Physical dual-core 3.1 GHz (2 logical cores per physical) or faster processor
USB 3.0 controller dedicated to the Kinect for Windows v2 sensor or the Kinect Adapter for Windows for use with the Kinect for Xbox One sensor
4 GB of RAM
Graphics card that supports DirectX 11
Windows 8 or 8.1, Windows Embedded 8, or Windows 10

Other requirements:

Install Kinect drivers
Position of the Kinect camera
Start application, setup gestures, start browser with game or similar

4.2 Graphical user interface

4.3 Back-end software

4.4 Gesture recognition

```
bool Model::isGestureExecuted(int checkLabel, int posInBuffer, int recursiveCounter, int
    badCounter)
{
        //The base label exists and is linked to a posture, so the posture has been executed.
```

```
        if (activeProject->containsLabel(checkLabel) &&
            activeProject->getGestureClass(checkLabel)->getGestures().front()->isPosture())
                return true;

        //Done enough recursive checks to confirm the gesture has been executed.
        if (recursiveCounter >= NB_OF_LABEL_DIVISIONS)
                return true;

        int nextLabelToCheck = checkLabel - 1;
        for (int i = posInBuffer; i >= 0; i--)
        {
                if (labelsBuffer.at(i) == nextLabelToCheck)
                        return isGestureExecuted(nextLabelToCheck, i, recursiveCounter + 1,
                            0);
        }

        //A label that is one less than the last label found cannot be found in the buffer,
            but the
        //gesture may still have been executed, so keep checking for the next one.
        if (badCounter > 0)
                return false;

        return isGestureExecuted(nextLabelToCheck, posInBuffer, recursiveCounter + 1,
            badCounter + 1);
}
```

Chapter 5

Results

- Korte samenvatting van bereikte resultaat
- [belangrijker] Resultaten gebruikerstest
- Interpretatie van de resultaten
- ...

Chapter 6

Discussion

- Reflectie behaalde resultaat
- Reflectie proces
- Voorstellen voor verbetering (efficiëntie,. . .)
- Voorstellen voor toekomstige projecten (vb: focus op besturing door kinderen ipv kinesisten)
- . . .

Chapter 7

Conclusion

- Samenvatting probleemstelling, implementatie en resultaten
- ...

Appendix A

Explanation about the appendix.

FACULTY OF ENGINEERING TECHNOLOGY
CAMPUS GROUP T LEUVEN
Andreas Vesaliusstraat 13
3000 LEUVEN, Belgium
tel. + 32 16 30 10 30
fax + 32 16 30 10 40
fet.groupt @kuleuven.be
www.fet.kuleuven.be