computergraphics tübingen

Funfair: EEG-based game control

Robert Eisele & Robert Geirhos

March 30, 2017

Praktikum Computergrafik WS2016/17
Wilhelm Schickard Institute of Computer Science

Funfair: EEG-based game control

Robert Eisele & Robert Geirhos

Figure 1: You can place a teaser here.

Abstract [insg.: 6 Seiten, mind. 4 Seiten reiner Text]

Electroencephalography (EEG) is a technique for measuring brain activity. Although widely used in neuroscientific research, it has thus far only rarely been used for the purpose of computer game control. Here we show that a leightweight EEG device, the Emotiv XXX, can be successfully used to play a broad range of simple yet challenging computer games. Furthermore, we demonstrate that rather than merely substituting a mouse or keyboard, an EEG device enables the development of a new type of games which go beyond traditional means of game control. We specifically developed six games, all of which can be played by controlling one's mental activity. We anticipate our work to be a starting point for ever more complex games and a new, keyboard-free gaming experience.

1 Introduction

Give an introduction to the problem. This might start from history, motivate the problem and end with the computational demands. Motivate why to use GPUs. You should also introduce CUDA in a way that makes an average computer science student understand what it is.

2 Description of the Solution

Describe your solution to the problem in detail. I propose to spend one chapter on a general, simple solution strategy so that the reader is familiar with how to solve the problem. This could also be a summary of the paper or document you mainly used to base your work on.

Subsequently, your particular solution should be described. Spend at least one chapter on describing the (parallel) improvements you made upon the naive implementation. It might be advantageous to split the implementation description to parts, each filling a subsection, and to give an overview over those parts first.

2.1 Sensor data analysis

All data, if not stated otherwise, were analyzed using R [R C16]. Our goal was to build several sensor data classifier which, based on an analysis of the data, would be able to tell apart several menal states (`no_action, action_1, action_2, ...`). In the process of building these classifiers, we faced two key challenges: Firstly, there was no prior mapping from the 16 sensors to a mental state - moreover, we did not even know which sensor would correspond to which mental signal (like α or β waves). Secondly, the sensor data - as is the case for almost any recorded data - seemed to be noisy. We therefore approached both challenges not in a knowledge-driven, but rather in a data-driven way: We recorded data for a variety of different actions and mental commands (Fig. 3) and carefully explored and visualized the sensor data recordings. As can be seen in the Figure, some commands such as `clench fist` were barely visible in the data, whereas other commands (e.g. `shake head, clench teeth`) produced considerably con-

sistent spike-like patterns. We therefore decided to focus on the actions that seemed to be visually distinguishable in the data.

As most actions lasted for a very brief period of time, roughly 1-2 seconds, we buffered all sensor data individually with a sliding window of 2 seconds, which corresponds to 256 data points per sensor since the sampling frequency was approximately 128 Hz, and took this sliding window as the basis for all further data analysis. We then implemented two different types of classification. The first one is a mapping based on all sensors to a continuous value indicating the degree of relaxation (from tense, agitated to very relaxed and calm). It was achieved by taking into account the variance of the sum of all sensor data, which is based on the observation that a very tense state cannot be traced back to one particular sensor, but elicits high variance in several sensor recordings. The second type of classification, detecting discrete events, directly focuses on specific sensors: `Y` and `X` for `shake head` classification, `F3` for `nodding` and `O1` for the classification of `clench teeth`. It would have been possible to classify `squint eyes` as well, however the signals appeared to be hardly distinguishable from `clench teeth` (and furthermore closing one's eyes during gameplay was found to be rather annoying). This second type of classification was achieved through thresholding the windowed sample variance of a targeted sensor. As a result, we were able to classify one continuous mental state (`relaxation`) as well as three discrete events (`clench teeth, nodding, shake head`).

2.2 Gyro data analysis

2.3 Highstriker

A highstriker is a classic game that can be found on most funfairs. We took the core idea of striking with a hammer to achieve a score on a vertical bar and made a few adaptions to it. Rather than a using a virtual hammer, we mapped current the mental state to a score: the more relaxed one is, the better the score (and the friendlier the status indicator, ranging from `Burnout candidate` to `Master of Yoga`). As this game is, in our experience, both quite fun and very easy to play, we made it the first game, enabling the user to get to know how he can control his mental activity to achieve a good score. This then prepares for more complex games such as the balancing game. So-called *brain self-regulation* based on visual feedback has been successfully used in clinical settings, e.g. to teach psychopaths to control their aggression [KVE+15]. It would be interesting to see whether some of our games (the ones that can only be won when one learns how to calm down and relax quickly) could be used to train children suffering from ADHD (attention deficit hyperactivity disorder) in a fun way.

2.4 Painter

The core idea behind the painter game can be described as follows: Wouldn't it be fun to take a snapshot of yourself (using your computer's camera) and then having this image painted in a style that matches your mood?

We achieved this by accessing the computer's camera through the browser, sending the plain image to the backend which forwards it, along with a style inferred from the current mood, to the Turbo-Deepart website[1] via the DeepArt API[2] to paint the image, which is then subsequently sent back to the game. DeepArt basically implements a neural network-based technique for painting an input image in the style of an arbitrary style image [GEB16]. For an illustration of how the Lena image looks like when painted in four different styles, see Fig. 2. We were able to contribute a commit to the DeepArt API (`rgeirhos`, commit `50f200f`, merged Dec 14, 2016), solving an early image retrieval issue.

As an additional feature, the painted image is not only shown but lies behind a veil first (low alpha value). One then needs to use either the mouse to hover over the image and reveal it piece by piece, or use a Leap Motion controller[3] to do so. A Leap Motion device measures hand gestures and translates them into a 3D hand model on the screen. We mapped the position of the hand model to a certain position on the canvas, which enables the user to simulate holding a paint brush in his or her hands and paint the image with painting-like movements.

[1] `http://turbo.deepart.io/`
[2] `https://github.com/deepart-io/deepart-api`
[3] `https://www.leapmotion.com/`

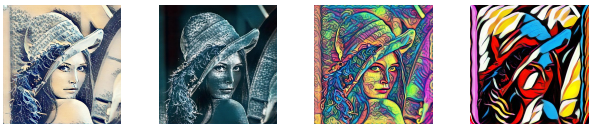

Figure 2: Four different style examples for the painter game. Images are sorted from a relaxed, calm style (left) to a more excited, agitated style (right).

2.5 Magic Duel

2.6 Mastermind

2.7 Wireloop

2.8 Balancing

Our balancing game is perhaps the most challenging game, as it requires to balance a little man on an inherently instable seesaw: if tilted to one side, one needs to strain oneself in order to get it back to balance - but then, it tilts to the other side and one instantly needs to switch to 'zen mode', to total relaxation... if one is able to do this for 30 seconds, the game is won, otherwise it is re-started.

In some relaxation techniques such as *Progressive Relaxation* [Jac38], being able to consciously change from straining oneself to relaxation mode (and vice versa) is a core element: if one learns and trains how to change between these states, this can be effortlessly applied in situations where one is naturally stressed and tense, such as ahead of an exam or a job interview. It would be interesting to explore whether a game like our balancing game could be used in a similar fashion - which would then enable one to teach relaxation techniques to a broader variety of people than those attracted by, say, adult evening classes.

2.9 Architecture & Backend

3 Possible Extensions

Describe possible extensions and discuss why they could be useful.

Viele Daten -> LSTM
weitere Spiele beliebig hinzufuegbar
Spiele fancier machen (3D, ...)

A General Infos

You don't need an appendix. The two appendix sections are just there to give some additional or general information.

This document describes roughly, what the documentation of the Praktikum project should look like.

Note that your final documentation of your project should contain 6 text pages using this template plus the cover and the empty sheet at the beginning. Within the 6 pages, at least 4 pages should be only text.

References

[GEB16] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2414–2423, 2016.

[Jac38] Edmund Jacobson. Progressive relaxation. 1938.

[KVE+15] Lilian Konicar, Ralf Veit, Hedwig Eisenbarth, Beatrix Barth, Paolo Tonin, Ute Strehl, and Niels Birbaumer. Brain self-regulation in criminal psychopaths. *Scientific reports*, 5, 2015.

[R C16] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2016.

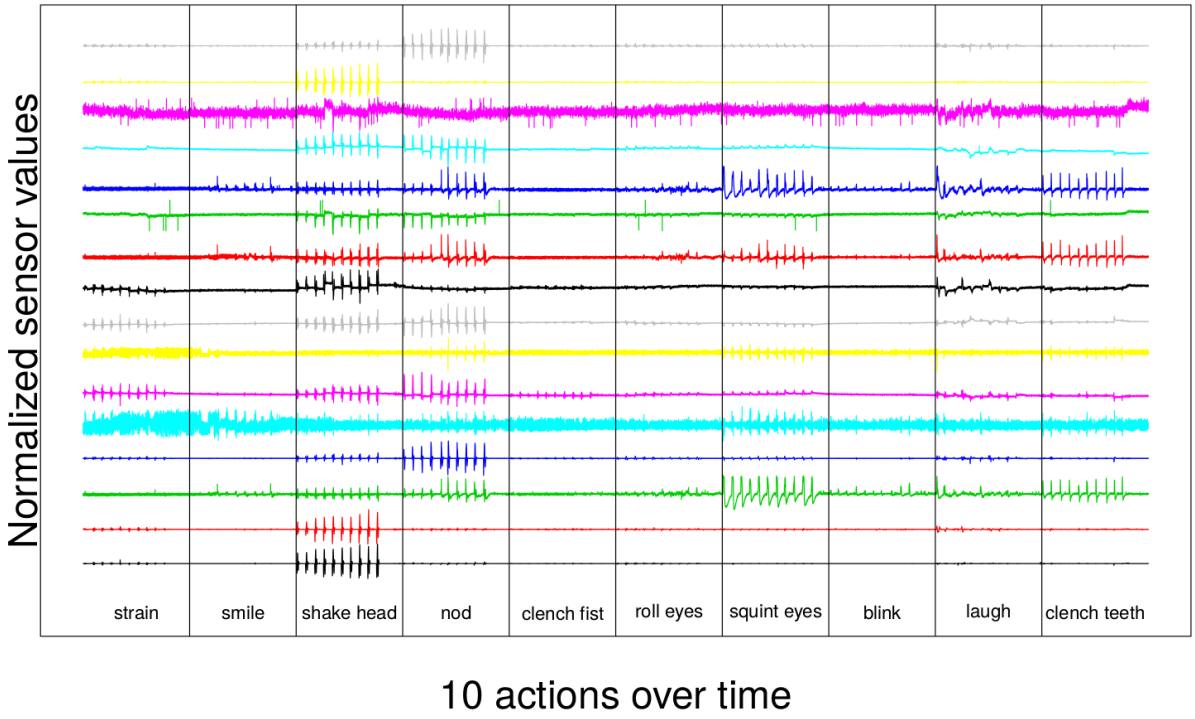

Figure 3: Raw sensor data for ten different actions and mental commands. All data are normalized to lie within [0, 1]. Every sensor has a different offset here for better visibility. Sensors from top to bottom: `Y, X, F4, FC6, AF4, F8, T8, P8, O2, O1, P7, T7, F7, AF3, FC5, F3`. Every action corresponds to one minute of data recording, during which over the first 45 seconds the action was briefly executed every five seconds, followed by a short break until the next type of action began.