Christoph Kapeller

A Steady-State Visual Evoked Potentials (SSVEP) based Brain-Computer Interface: Design of an Embedded On-Screen Stimulation Module

Master Thesis

Institute for Knowledge Discovery
Laboratory of Brain-Computer Interfaces
Graz University of Technology
Krenngasse 37/IV
8010 Graz, Austria
Head: Assoc.Prof.Dipl.-Ing.Dr.techn. Gernot Müller-Putz

Supervisor:
Assoc.Prof.Dipl.-Ing.Dr.techn. Gernot Müller-Putz

Dipl.-Ing.Dr.techn. Christoph Hintermüller

Evaluator:
Assoc.Prof.Dipl.-Ing.Dr.techn. Gernot Müller-Putz

Graz, May, 2012

Zusammenfassung

Ein Brain-Computer Interface (BCI), zu Deutsch Hirn-Computer-Schnittstelle, erlaubt es dem Benutzer ein Gerät oder ein Programm auf Basis seiner Hirnaktivität zu steuern. Ein solches System kann Personen mit eingeschränkter Kommunikationsfähigkeit helfen, neue Wege zur Interaktion mit ihrer Umwelt zu finden. Im Zuge dieses Projektes wurde ein BCI entwickelt, das durch Interpretation der Hinströme auf Grund unterschiedlicher visueller Reize Entscheidungen trifft und diese, über eine Netzwerkverbindung, an ein entferntes Gerät weiterleitet. Visuelle Reizfolgen mit konstanter Frequenz generieren ein so genanntes steady-state visuell evoziertes Potential (SSVEP), welches von der Kopfhaut abgeleitet und verstärkt wird. Das entwickelte BCI analysierte das Elektroenzephalogramm (EEG) des Benutzers und extrahierte, entsprechend der verwendeten Reizsequenz, Merkmale aus den Signalen. Diese Merkmale wurden online zu definierten Befehlen klassifiziert. In den durchgeführten Experimenten wurden drei unterschiedliche Konfigurationen getestet. Zwei Systeme basierten auf Stimulation mit konstanter Frequenz, wobei ein System Leuchtdioden (f-VEP LED) und das andere System einen Monitor als Lichtquelle (f-VEP on-screen) verwendete. Im dritten System erfolgte die Stimulation mittels pseudo-randomisierter Sequenz auf einem Monitor (c-VEP on-screen). Zur Stimulation mittels Monitor wurde ein Programm implementiert, das BCI Kontrollen visualisierte und über eine Netzwerkverbindung mit dem BCI verbunden war. Das BCI selbst wurde in einer Rapid-Prototyping Umgebung implementiert. In Summe nahmen elf gesunde Probanden teil, wobei jeder Proband Experimente zu jeder der drei Konfigurationen mit machte. Die Experimente umfassten einen Genauigkeitstest der Klassifikation und eine Robotersteuerung mit visuellem Feedback. Die c-VEP Konfiguration zeigte mit 94.51 % mittlerer Klassifikationsgenauigkeit den besten Wert. Die f-VEP LED Konfiguration erreichte durchschnittlich 83.64 %, während die f-VEP on-screen Konfiguration einen Durchschnittswert von 84.18 % zeigte. Ein Quade-Test zeigte signifikant bessere Klassifikationsgenauigkeit ($p = 0.0168$) und signifikant schnellere Steuerung ($p = 0.0187$) der c-VEP on-screen Konfiguration. In allen mit f-VEP basierenden BCIs durchgeführte Experimenten zeige die „on-screen“ Stimulation vergleichbare Ergebnisse und kann nun für weitere Anwendungen verwendet werden.

Abstract

A Brain-Computer Interface (BCI) allows a user to control a device or a program on base of the brain activity. Such a system may help people, who have only limited or no possibility to communicate with their environment. Within this thesis a BCI system was developed that performs tasks based on specific brain waves caused by visual stimulation. Visual stimuli with constant stimulation cycles generate a so called steady-state visual evoked potential (SSVEP). A biosignal amplifier provided a signal that was used for further signal processing. The BCI analyzed the users' electroencephalogram (EEG) online and extracts features, which are related to the type of visual stimulation. Via network connection, the BCI sent the corresponding command to a remote device. Three configurations of the BCI were used for the experiments in this work. Two of the three configurations were based on visual stimulation with constant frequencies. The only difference between these configurations was the stimulating light source, where one configuration used LEDs (f-VEP LED) and the other one used a computer screen (f-VEP on-screen). The third configuration used pseudo-random sequences in combination with a computer screen for stimulation (c-VEP on-screen). For the on-screen stimulation, a software was implemented that visualizes the BCI controls and communicates with the BCI via a network connection. The BCI was developed within a rapid-prototyping environment. For each configuration an online classification accuracy test and a robot control in combination with visual feedback were performed. Eleven healthy subjects participated in experiments for each of the configurations. In all experiments the c-VEP on-screen configuration showed the best results. The mean online classification accuracy was 94.51 %. The f-VEP LED configuration showed 83.64 % average online classification accuracy. The f-VEP on-screen configuration showed quite similar results to the f-VEP LED configuration. In this configuration the average online classification accuracy was 84.18 %. A Quade-test showed that the c-VEP on-screen configuration provided significantly higher classification accuracy ($p = 0.0168$) and significantly faster movement of the robot ($p = 0.0187$) than the f-VEP LED and the f-VEP on-screen configuration. In all experiments performed with the f-VEP based BCIs, the on-screen stimulation module showed nearly similar results compared to LED stimulation. Therefore, the on-screen solution works and can be used for further experiments.

Table of Contents

1. Introduction

1.1. The VERE Project

As the work presented in this thesis is part of the VERE project, this section should give the reader an idea on what the project is about.

The VERE project deals with the virtual embodiment and robotic re-embodiment (VERE) of people. A user should be able to act and feel with a surrogate, also with the feeling that the foreign body or virtual avatar is the own body. Two principle concepts are followed within the project. The first idea is the embodiment in an avatar in a virtual reality environment. The robotic embodiment is the second concept in the VERE project. The user is embodied in a physical robot and controls the device from a remote position. This could give people who are not able to move the chance to participate in real world activities. The main research topics to fulfill these considerations are: the construction of an embodiment station that reads and sends signals from and to the user, the extraction of the operators' intentions (through monitoring of brain signals and physiological signals), an overall software platform (for the different streams of work) and the investigation of the philosophical and ethical principles concerning the embodiment (VERE, 2010).

VERE is an integrated Project funded under the European Seventh Framework Program, Future and Emerging Technologies (FET), Grant Agreement Number 257695.

1.2. The visual system

When light shines at the eye, it passes the cornea and the lens. At the back of the eye, the light shines at the retina. The retina is part of the central nervous system (CNS) and contains several types of neurons involved in the generation of electrical signals out of visual stimuli. Two of them are the photoreceptor cells, which are directly located on the epithelium at the back of the eye. Because of their shape, they are named rods and cones. However, the shape is not the only difference, they also have individual functions. Rods are more sensitive to light than cones. The temporal resolution is 55 Hz for cones and 12 Hz for rods (Tessier-Lavigne, 2000). Moreover, the wavelength dependence of cones allows color vision. In total, there are three types of cones, where each type is sensitive to a specific frequency range: short wavelength cones (blue), medium wavelength cones (green) and long wavelength cones (red)

(Purves, 2004). The number of rods is 1,2 * 10^8 and 20 times higher than the 6 * 10^6 cones (Faller et al., 2004, Tessier-Lavigne, 2000).

To reach a receptor, the light has to travel through layers of interneurons and ganglion cells. The cells in the retina have no homogenous distribution. On one hand, the so called fovea is an area with thinner layers of neurons and more cones, than anywhere else at the retina. Although, more rods exist in total, the spatial resolution in the fovea is higher for cones (due to the high concentration of cones in this area). On the other hand, the blind spot is an area of the retina with no photoreceptors. This blind spot is the path that is used by the optic nerve fibers to leave the retina. The bundle of nerve fibers leaving the eye is called the optic nerve (Tessier-Lavigne, 2000).

Incoming light activates pigment molecules in the photoreceptors. Instead of generating an action potential, the activated receptor generates a hyperpolarization. Interneurons transmit the signal to ganglion cells, which generate trains of action potentials leading to the optic chiasm in the diencephalon. The ganglion cells receive information from circular receptive fields consisting of a group of photoreceptors. These receptive fields have a center area that is directly connected to the receptors (through bipolar interneurons). Neighboring receptor cells build a surrounding area around the center. The ganglion cells receive signals from the surrounding area via lateral pathways. The connection to neighboring receptors is realized with horizontal cells and amacrine cells (Figure 1).

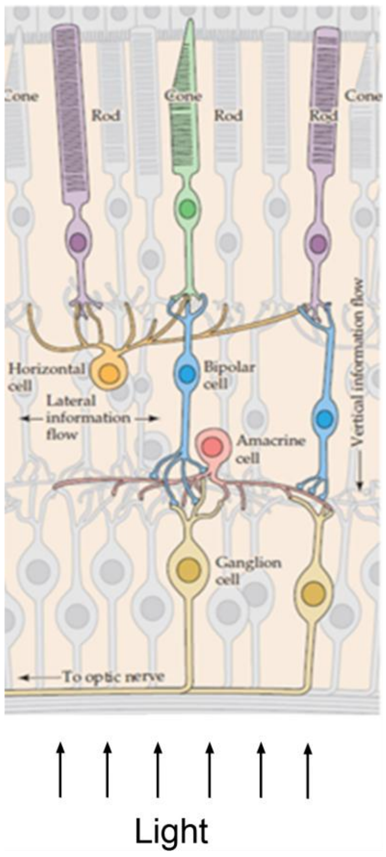

Figure 1: Network of the photoreceptors and the corresponding neurons. Ganglion cells are connected to the optic chiasm. Horizontal cells build connections between photoreceptors. Amacrine cells build connections between ganglion cells. Bipolar cells provide a direct connection from photoreceptors to ganglion cells. Modified from from (Purves, 2004).

There are two types of receptive fields: on-center and off-center. In on-center fields the ganglion cells are excited, when light hyperpolarizes the receptors in the center. Exposure of the surrounding area would inhibit the ganglion cells. Off-center ganglion cells fire when the light at the center receptors is turned off or when the neighboring receptors are illuminated. The number of on-center fields and off-center fields is nearly the same. Furthermore, every receptor cell sends signals to both types of receptive fields (Tessier-Lavigne, 2000). Such an organization of receptive fields allows the extraction of useful information like contrasts or edges and directional movement in the image.

The axons of the ganglion cells leave the eye through the optic disk and lead to the optic chiasm in the diencephalon. Here, about 60 % of the nerve fibers cross the fibers from the other eye (Purves, 2004). This crossing causes a splitting of the visual field. The left area of the visual field is projected to the right half of the brain and vice versa. After the optic chiasm, the fiber bundles form the optic tract. Main targets of the ganglion cells are: the lateral geniculate nucleus of the thalamus, the pretectum and the superior colliculus in the midbrain. The midbrain controls eye and head movements. The pretectum, which lies between the midbrain and the thalamus, is responsible for the pupillary light reflex. Most important destination of the ganglion axons is the lateral geniculate nucleus. It is part of the primary visual pathway and leads to the primary visual cortex (also called striate cortex) lying on Brodmann's area 17. Nerve fibers, which come from the fovea, are projected to the posterior end of the visual cortex. An interesting fact is that the nerve fibers of both eyes are mixed in the visual cortex for the first time. Before reaching the cortex, the nerves run along distinct paths (Figure 2).

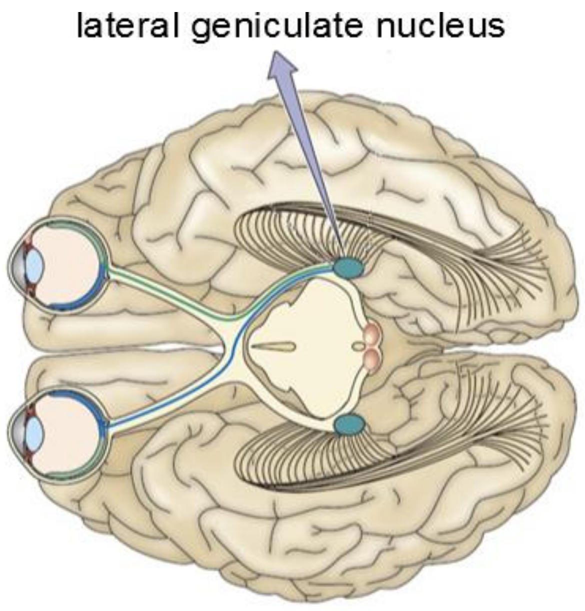

Figure 2: The primary visual pathway consists of the optic nerve from the eye to the optic chiasm. After crossing nerve fibres from the other eye the optic tract leads to the lateral geniculate nucleus in the thalamus. This nucleus consits of six layers, where each layer is destination of fibres from one eye. From the lateral geniculate nucleus, the neurons mainly project along the optic radiation to the layer IV of the striate cortex. Modified from (Purves, 2004).

Within the lateral geniculate nucleus the cells are not only separated by their origin, but also by size. There are magnocellular and parvocellular layers, where each layer is connected to different types of ganglion cells. Therefore, the primary visual pathway consists of parallel streams providing different information. First, the magnocellular pathway transmits transient signals after visual stimuli. Second, the parvocellular stream sends durable signals, which also contain information about color. Third, the koniocellular pathway also provides color information, but only with respect to cones sensitive to short wavelengths.

At the visual cortex, the signals from the thalamus are combined, which allows the extraction of depth information, based on the binocular information. The primary visual cortex projects to other areas of the cortex, where visual information is interpreted together with signals from other senses.

1.3. Electroencephalogram (EEG)

Electroencephalography is the non-invasive derivation of brain waves over the cerebral cortex. Therefore, at least two scalp electrodes are used to measure the voltage related to a reference or another scalp electrode (Olejniczak, 2006).

1.3.1. Signal Generation

Action potentials are the common way to transmit information between neurons. Amplitudes of up to 100 mV are the highest potential differences in the CNS (Zschoke and Hansen, 2002). However, the trans-membrane potential difference only exists about 1-2 ms and there is hardly any chance to measure the corresponding electrical field variations outside the cell. Therefore, the derived EEG signals must origin from another source: the synapses. They which act as potential generators within the cerebral cortex (Li and Jasper, 1953). An activated synapse causes a local change of the membrane potential of the receiving neuron. The remaining part of the cell surface, which is not affected by the synapse is the post-synaptic membrane. This leads to another potential difference between sub-synaptic and post-synaptic membrane. Such a post-synaptic potential lasts about 10-100 ms, which is much slower than the corresponding action potential (Zschoke and Hansen, 2002). Synapses can either have inhibitory or excitatory behavior. The corresponding potentials are the inhibitory post-synaptic potentials (IPSP) and the excitatory post-synaptic potentials (EPSP), respectively.

A post-synaptic potential causes the movement of ions through the intercellular space. Due to the tissue specific impedances the ionic current causes a field of potential differences, which is called cortical field potential. This field potential can lead to potential differences even at the scalp, which are the base for EEG measurements.

During the presence of a post-synaptic potential, the synapse sets up an electric dipole related to the rest of the cell. Neurons (mainly pyramidal cells) can have up to 10000 synapses per cell and every single synapse generates such a dipole (Zschoke and Hansen, 2002). Therefore, the sum of the synchronized dipoles defines the strength of the resultant field. However, the orientation of the dipoles is quite more important for EEG derivations. Only normal to the scalp oriented dipoles generate a measureable signal on the skin surface. Groups of pyramidal cells, which are oriented the same way, are called open fields. About 30 % of the neurons in the cerebral cortex are aligned normal to the scalp (Zschoke and Hansen, 2002). The EEG signal generated by a post-synaptic potential depends on the type of the synapse and also on the location of the synaptic connection (Figure 3).

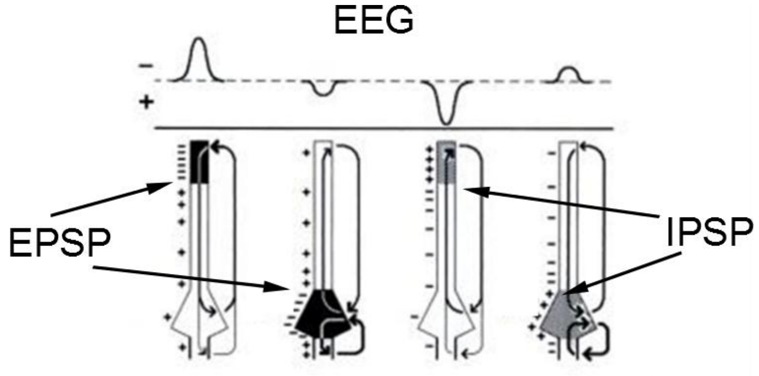

Figure 3: Both, the location and the excitatory or inhibitory behavior of the post-synaptic potentials influence the resultant field potential. An axodentric EPSP results in the same signals in the EEG as an axosomatic IPSP. This is why the behavior of the synapse cannot be extracted out of the EEG derivations. Modified from (Zschoke and Hansen, 2002)

The field potentials are influenced by the glia cells too. During neuronal activity the extracellular concentration of potassium can increase up to three times of the normal value (Zschoke and Hansen, 2002). The permeability of potassium ions through the glia cells membrane is very high. So glia cells act like a potassium buffer, which avoids a too high depolarization of the surrounding neurons. In unison, the glia cells themselves get depolarized. Because of inter-cell connection or also called gap junctions, the depolarization expands to other glia cells. Together, the connected cells can form electric dipoles and so they are possible generators for field potentials.

1.3.2. Signal Derivation

The most important carriers of charge on the skin are sodium and potassium ions. To reduce the skin impedance, an electrolyte containing ions is used. Then, an electrode provides connection to the EEG device. Usually, the electrodes consist of Au or Ag/AgCl. During connection the positive metal ions of the electrode diffuse to the electrolyte and set up a double layer containing metal ions and ions from the electrolyte (e.g. Cl^-). This double layer causes a permanent DC offset voltage and also parasitic current at the beginning of the connection. In Au electrodes the ions in the double layer underlie only marginal movements, which is a problem for the measurement of low frequencies. In contrast the Ag/AgCl electrodes allow the chloride ions to pass the border between electrode and electrolyte. So the signal of this electrode type depends less on frequency, than Au electrodes. The impedance of the electrodes should be equal for all electrodes (a value about 5 – 10 kΩ is desirable) (Zschoke and Hansen, 2002).

In principle, a differential amplifier is used to derive EEG signals, caused by the low amplitude of the derived signals and the high influence of noise (e.g. 50/60 Hz power line). Requirements for EEG amplifiers are linearity, high common mode rejection ratio (CMRR), a high input impedance and flatness over the used frequency range (Epstein, 2011).

There are some conceptual techniques to derive the EEG. In every case the signals are extracted through the potential difference of one to another or one to more derivation points. First principle is the unipolar derivation, which uses a common reference. In this configuration each derivation point gets the same reference potential. The reference point can be chosen anywhere on body. In general, the best location of the reference is as far away from the scalp as possible. So, no cortical signals would be derived with the reference electrode. Unfortunately, the reference could record other artifacts, which are caused by large skeletal muscles or the heart and do not occur on the scalp. To avoid muscular and ECG artifacts, it is advisable to choose a reference point that is not far away from the brain (e.g. the earlobe). Of course, any other position on the scalp can be the reference point. But a reference placed on the head and also on the earlobe, is affected by cortical signals, which may cause a distortion of the differential signal (Zschoke and Hansen, 2002).

Another possible type of derivation is a bipolar setup. Here the signal is the potential difference of two recording electrodes. The great advantage of such a configuration is the robustness against artifacts. Usually, artifacts appear similar in neighboring derivations. This leads to a cancellation of the artifacts in a differential signal. Also field potentials of neighboring recordings are quite similar, so that the derived signal is affected by cancellation too. Therefore, only phase shifted signal components can be measured with a bipolar setup. The obtained signals have a decreased amplitude compared to unipolar derivations and also inversed polarity (Zschoke and Hansen, 2002).

In addition, there are setups with a virtual reference. The reference signal can for example be the average of all recorded EEG or just the surrounding electrodes, as for the Laplacian reference (Hjorth, 1975).

1.3.3. International 10-20 System

Although every human brain differs in shape and size, the relative position on the cortex of the functional areas is nearly constant. An asymmetric shape of the head leads to asymmetries in the brain. This is why a relative positioning system is used, according to the international 10-20 system. Starting from the nasion and inion, the electrodes are placed in relative distances of 20 %. The nasion is the intersection of the frontal bone and the nasal bones. The inion is a projection of the occipital bone. From each landmark, the initial step to the first electrode is 10 %. This leads to a setup of 19 signal deriving electrodes. Also configurations with more electrodes are possible (see section 0, Figure 18) (Zschoke and Hansen, 2002).

1.3.4. Visual Evoked Potentials

After stimulation of the retinal receptor cells in the eye, the corresponding signals run through the visual pathway to the occipital lobe. The resulting potentials generated by the visual cortex can be recorded at the positions O1, Oz and O2 of the 10-20 system. For this derivation, the earlobe or the Fpz position can be chosen as reference point (Bach et al., 2005). There are two general forms of visual stimulation: stimulation with short light flashes or pattern reversal stimulation (Paulus, 2005). For flash stimulation either light emitting diodes (LED) or a stroboscope, with a flash duration of about 10 µs are used (Bach et al., 2005). Usually, pattern reversal stimulation is performed with a screen using inverting checkerboards. The optimal check-size is in the range of 0.1° to 0.3° of visual angle (Harter, 1970). As an alternative, VEPs may be evoked using the so called pattern on/off stimulation (Bach et al., 2005). Thereby, the pattern appears out of a homogenous background. Figure 4 shows an evoked potential, generated by flash stimulation. The latencies of the peaks can vary with age and stimulus frequency. The brightness of the visual stimulus should be more than 100 cd/m² (Bach et al., 2005). Also the size of the total stimulating area affects the amplitude of the VEP. Usually, larger stimulation area leads to higher amplitudes in the signal. However, (Bartl et al., 1978) showed that the enhancement of the measured signal decreases for larger stimulation areas. The VEP amplitude strongly decreases for stimulation areas less than 7.5° x 7.5° visual angle. A visual stimulus should have a rise time from dark to bright less than 10 ms ((Bach et al., 2005).

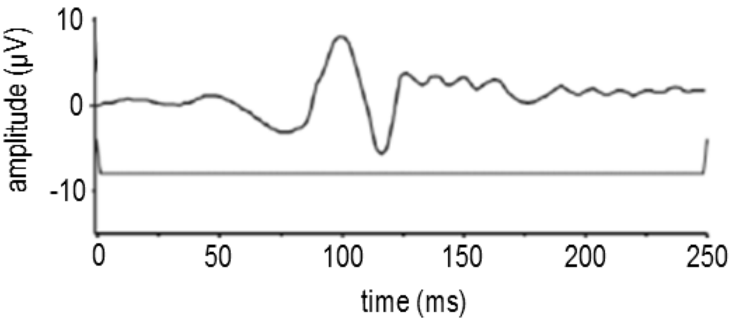

Figure 4: Visual evoked potential after flash stimulation. In diagnostic the P100 peak at 100 ms is very important, because this peak is used for latency and amplitude measurements. The amplitudes of VEPs lie in the range of 5 µV – 10 µV. Modified from (Russo et al., 2002).

If the visual stimulation frequency is less than 3.5 Hz the VEP is called transient. Stimuli over 6 Hz are leading to a sinusoidal shape of the corresponding EEG (Regan, 1989) (Figure 5). This phenomenon is called steady state visual evoked potential (SSVEP). The spectrum of an SSVEP shows a strong peak at the fundamental stimulation frequency and its corresponding higher level harmonics. The group of (Müller-Putz et al., 2008b) investigated the optimal electrode position for recording the SSVEP and the higher level harmonics. When pattern reversal stimulation is used, two inversions per cycle occur. This is why the resulting SSVEP reaches its maximum in spectral power density (PSD) for the second harmonic of the stimulation frequency. In order to avoid the confusion between the frequency of the SSVEP and the one of the stimulation pattern, the latter is characterized by the number of reversals per second.

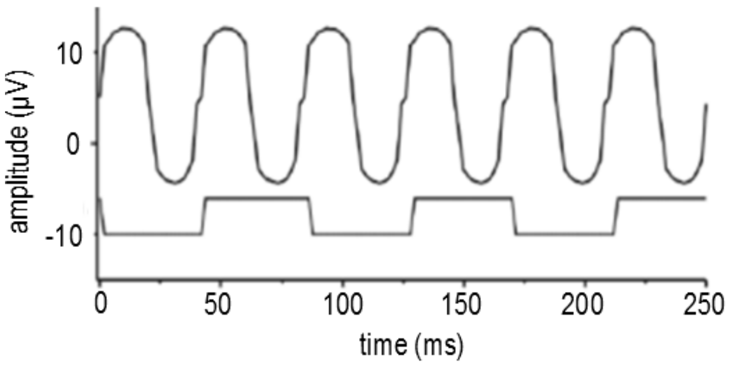

Figure 5: An SSVEP evoked by 12 Hz stimulation. Modified from (Russo et al., 2002).

(Regan, 1966) investigated the influence of stimulation color to the SSVEP amplitude. He found that the maximum PSD in the spectrum of an SSVEP depends on the stimulation color (Figure 6). The highest amplitude is achieved using red light and 10 – 12 Hz for stimulation. For white flickering the highest amplitude is reached at 15 Hz (Pastor et al., 2003).

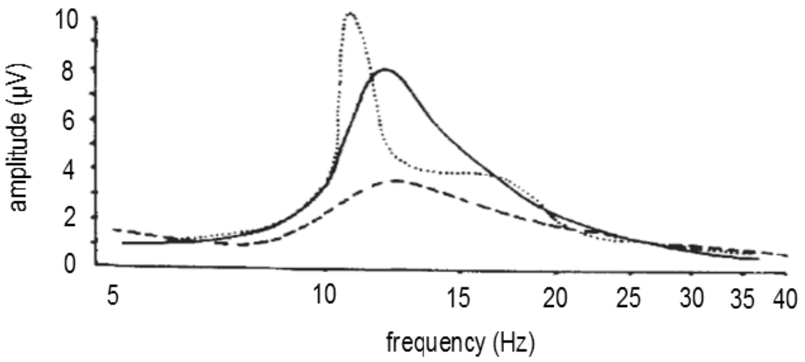

Figure 6: Visual stimulation performed with three different colors: red (dotted line), yellow (dashed line) and blue (solid line). The amplitude of an SSVEP depends on the selected frequency and on the color of the stimulus. Modified from (Regan, 1966).

1.3.5. Photic-Induced Seizures

After high excitation of pyramidal cells in the cerebral cortex the inhibitory mechanisms may be insufficient, to prevent a simultaneous activation of a large number of neurons. Such a behavior may lead to seizures, similar to those observed from epileptic patients. Abnormalities in the EEG, caused by visual stimuli, are called photoparoxysmal response. In their anthology about photic- and pattern-induced seizures, (Fisher et al., 2005) describe some characteristic parameters of visual stimuli, which can increase the risk to provoke seizures, such as flash rates between 15 and 25 cycles. Also the stimulation color influences the sensitivity, where red light is more likely to provoke seizures than sources emitting white light. Stimulation patterns with spatial change rates of 0.3° of visual angle provoke seizures more often than other patterns. All these parameters show values that are close to the VEP maximizing ones. As a consequence, a BCI based on VEPs should only be used by people without any known photo sensitivity. Around 0.3 – 3 % of the people show abnormalities in the EEG related to visual stimulation. Nevertheless, the number of people that suffer from light induced seizures is much lower (1 per 10000) (Fisher et al., 2005).

1.4. Brain-Computer Interfaces

A Brain-Computer Interface (BCI) is defined as a system or device that provides the user a communication channel, without using the normal neuromuscular output pathways of the body (Wolpaw et al., 2002). It allows people to interact with their environment, even if they have limited or no possibilities to control their muscles. There exist two distinct types of BCIs. On one hand, there are independent systems, which do not require any normal output pathway of the brain, to generate the task specific signals in the EEG and to extract the intention of the user. On the other hand, there are dependent systems that require activity in the normal output pathways of the brain, to generate the signals in the EEG (Wolpaw et al., 2002). An example for a dependent system is a VEP based BCI, where the user has to gaze at a specific target for visual stimulation. Various data acquisition techniques like electroencephalography (EEG), functional magnetic resonance imaging (fMRI), near infrared spectroscopy (NIRS), magnetoencephalography (MEG) and electrocorticography (ECoG) can be used to build a BCI system (Weiskopf et al., 2004, Coyle et al., 2004, Leuthardt et al., 2004, Wolpaw et al., 2002, Mellinger et al., 2007, Pfurtscheller et al., 1993). The EEG-based BCI is the most common one due to the low cost of the required components, the non-invasive signal derivation and the short time constants. After amplifying and pre-processing the brain waves that are in the range of about +/-100 µV, the data can be analyzed with task specific algorithms. However, EEG has only a limited spatial resolution, as a channel is influenced by millions of neurons. Higher spatial resolution can be achieved with ECoG based BCIs. The usage of ECoG based BCIs is restricted, because of the invasive signal derivation.

For signal processing several algorithms are used to extract BCI specific features like the signal amplitudes, latencies or the power density of parts of a limited frequency band. On base of the extracted features, a translation algorithm generates a device command that can be used to control a wheelchair, robots, prostheses, neuroprostheses or virtual avatars (Wolpaw et al., 2002, Müller-Putz and Pfurtscheller, 2008, Millan et al., 2004).

The performance of a BCI depends on the time, the system needs to identify the users' intention, based on the features. The faster the extracted features expose a specific behavior, the faster the reaction time of the BCI can be. In addition the performance also depends on the maximum number of commands that can be used. Systems are more flexible, if the user can choose from more commands within one identification cycle. (McFarland et al., 2003) investigated the dependence of the ITR on the trial duration and the number of targets. The ITR provides a measure to compare different BCI systems. The calculation of the bits/trial of a BCI is explained in (1) that is only valid, if the classification accuracy is the same for each provided class. Also the number of trials per class has to be the same during the tests.

$$B_{\boldsymbol{trial}} = log_2(N) + P \cdot log_2(P) + (1-P) \cdot log_2\left(\frac{1-P}{N-1}\right) \tag{1}$$

It is based on the bits per trial B_{trial}, the number of targets N and the probability P for correct classification of a target. Using (1), the bits/min (B_{min}) can be calculated via (2), in which t_{trial} is the duration of a trial in minutes.

$$B_{min} = \frac{B_{trial}}{t_{trial}} \tag{2}$$

1.4.1. Visual Evoked Potentials based BCIs

There exist several different paradigms, which can be used to construct a VEP based BCI. (Vidal, 1977) built a BCI stimulating different areas on the retina, allowing discrimination of the resultant VEPs. More common are BCIs that exploit the formation of SSVEPs in the visual cortex (Middendorf et al., 2000, McMillan et al., 1995). Two main methods are used, to code the information relevant on a single control element: phase coding and frequency coding (Wang et al., 2008). A phase coded system uses one common frequency for all target stimuli. This leads to the same response in the EEG, but phase shifted. Frequency coded systems use targets with different stimulation frequencies. The resultant EEG shows an SSVEP at the corresponding target frequency.

In their survey, the group of (Zhu et al., 2010) investigated 58 different BCI systems based on SSVEP, which used stimulation frequencies in the range of 4 to 50 Hz. They compared 24 BCIs using light stimulation with LED, 14 BCIs using on-screen stimulation with checkerboards and 18 systems using on-screen stimulation with solid rectangles. All other BCIs presented, used different stimulation methods that are not relevant for this work. BCIs based on LED stimulation showed the highest median ITR (42 bits/min), followed by rectangular stimulation (35 bits/min) and pattern reversal (26 bits/min). The high ITR of the LED stimulation may be caused by the possibility to produce arbitrary frequencies and therefore more targets are possible. As it can be seen in (1), the number of targets highly affects the ITR. Frequently used stimulation colors were red, green and white.

Usually, SSVEP based BCIs are systems depending on eye movements of the user. This reduces the usability of such a system to people, who are able to control the muscles of their eyes. Locked-in patients suffering from amyotrophic lateral sclerosis (ALS) are not able to use such a system. However, locked-in patients would benefit most from a communication channel offering high bit like SSVEP based BCIs. As the amplitude of an SSVEP also depends on attention (Russo et al., 2002), it is possible to design an BCIs using SSVEPs that does not require any muscular activity, as it was presented by (Allison et al., 2008). The authors built a system using two overlapping images, which are oscillating with different frequencies. While the images are flickering, the user has to concentrate at one of the images. The study showed that about half of the subjects were able to generate differences in SSVEPs, which are sufficiently large to control a BCI.

The features of an SSVEP based BCI are often extracted through spectral analysis. A first approach with Fast Fourier Transformation (FFT) and threshold detection was presented by (Cheng and Gao, 1999). Another approach for feature extraction is the lock-in amplifier system (LAS) that provides information about the amplitude of the target signal (Middendorf et al., 2000). Also the incorporation of higher level harmonics increases the classification accuracy (Müller-Putz et al., 2005).

In the last years many improvements were introduced with respect to signal processing algorithms. The minimum energy (ME) combination algorithm allows the setup of a multichannel SSVEP based BCI system (Friman et al., 2007). This leads to feature channels, which have an improved signal-to-noise ratio between the target signals compared to the ongoing EEG. The ME is also implemented within the Bremen BCI (Volosyak et al., 2009b) that was used to evaluate the usability of SSVEP based BCIs (Allison et al., 2010). Five controls (13, 14, 15, 16 and 16.5 Hz) were used, to navigate within a speller matrix and to select a character. The study showed that people can use the BCI with a mean accuracy of 95.78 % and an ITR of 13 bits/min. Thereby young and female subjects seemed to perform best. A later study showed that the performance decreases for frequencies above 30 Hz, compared to a medium frequency range of 12 - 30 Hz (Volosyak et al., 2011).

In a later publication the Bremen BCI reached a mean ITR of 61.70 bits/min and a mean accuracy of 96.79 % (Volosyak, 2011).

Another multichannel approach is based on canonical correlation analysis (CCA), that is used as a spatial filter to maximize the correlation between the sinusoid template signals and the target signal in the EEG (Lin et al., 2006, Bin et al., 2009a). Later, a combined frequency and phase coding BCI used the CCA for feature extraction (Jia et al., 2011). This allows stimulation through sinusoidal signals of the same frequency. For discrimination, every stimulation target starts flickering with a unique phase. Therefore, the system provided 15 targets and reached a mean ITR of 66.5 bits/min.

Of special interest for this work are on-screen solutions, embedded in a virtual environment. Unfortunately, the presence of the higher level harmonics and the given refresh rate of the screen may decrease the number of valid frequencies for on-screen stimulation. The optimal frequencies for a typical 60 Hz monitor are: 6.67, 7.5, 8.57, 10 and 12 Hz (Volosyak et al., 2009a). A BCI using these frequencies can include the first two harmonics into feature extraction. Several groups combined different frequencies within one target to increase the overall number of possible targets (Mukesh et al., 2006, Wang et al., 2010, Shyu et al., 2010). With pattern reversal stimulation the maximum number of possible frequencies can be increased too. In this case, the second harmonic is the most dominant frequency, caused by the visual stimulation. Therefore, the lowest stimulation frequency can be chosen in the range of 3Hz or 6 reversals per second.
(Lalor et al., 2005) implemented a 3D gaming environment, where the user has to balance along a rope. Two inverting checkerboards allow movement to the left and to the right. Visual stimulation is performed with 17 and 20 reversals per second. An EEG buffer of 2 s length is used to process signals recorded from O1 and O2. For feature extraction FFT is used, followed by a linear discriminant analysis (LDA) to calculate a classifier. The system achieved a mean ITR of 10.3 bits/min, based on a mean accuracy of 89 %.

Another SSVEP based BCI system allowed to control a car within a computer game (Martinez et al., 2007). Four checkerboard targets move together with the car along a virtual track, providing an improved visual feedback. In their paper, the authors compared the performance related to low- and medium-frequency range. Best results were reached with pattern reversals between 12 and 17 Hz. A 120 Hz CRT monitor was used for stimulation. The mean accuracy was 96.5 % and the ITR was 30 bits/min.

An approach for avatar movement in a virtual reality was presented by (Faller et al., 2010). The authors implemented an SSVEP based BCI, using red rectangles on a computer screen for visual stimulation. The avatar can be controlled with three targets (12, 15 and 20 Hz). The resultant positive predictive value (PPV) achieved for movement through a virtual apartment was 94.7%. Instead of the ITR, 10.9 activations per minute were presented as a result.
The "brain response interface" was presented by (Sutter, 1992). He used a CRT monitor to visualize a speller matrix with 64 keys. The refresh rate of the CRT was adjustable, so that the user could change the stimulation frequency. For stimulation, M-sequences were used instead of constant frequencies. M-sequences are pseudorandom binary sequences, which are nearly orthogonal compared to shifted versions of the same sequence. Every target flashed with a phase shifted version of the same M-sequence. Target identification was performed through a cross-correlation between the raw EEG and a normalized template. One sequence cycle took about 1.5 s, including a time lag of approximately 20 ms between the target sequences. The evaluation of the system performance showed that subjects were able to write about 10 – 12 words per minute. This is a very high rate compared to the ITRs of modern BCIs. However, the author used transcutaneous electrodes, which provide larger signal amplitudes compared to electrodes placed on the skin surface.

Another BCI that uses M-sequences for stimulation was presented by (Bin et al., 2009b). They introduced the code-based VEP (c-VEP) BCI, which is a mixture of time and frequency modulation, similar to the Code Division Multiple Access (CDMA) method in mobile communication. Later, the same group implemented a multichannel BCI based on c-VEP (Bin et al., 2011). For target identification the correlation coefficients between trained templates and the raw EEG were used. Therefore the canonical vector calculated with a CCA was used as spatial filter to maximize the correlation coefficients. They set up a 32 target system with a sequence length of 1.05 s. The resultant mean online accuracy was 85 %, which led to an ITR of 108 bits/min.

1.5. Motivation

A Brain-Computer Interface (BCI) allows a person to communicate and interact with his/her environment without using natural pathways. The most common way, to measure signals from the brain, is the non-invasive EEG, which records signals from the intact scalp. Persons with disabilities in muscle control or disorders of the neural pathways (e.g. after spinal cord injury, amyotrophic lateral sclerosis, muscular dystrophies, multiple sclerosis,…) are able to use such a system as a new communication channel (Wolpaw et al., 2002). Due to the high information transfer rate (ITR), BCIs using SSVEPs are suitable to control continuous processes, like moving an avatar (Zhu et al., 2010). In the last few years virtual reality (VR) systems and also games were set up with a BCI as input channel (Lalor et al., 2005, Martinez et al., 2007, Faller et al., 2010). All these systems use on-screen stimulation that is either directly implemented within the application or depending on a specific framework (e.g. Open Inventor based frameworks in the case of (Faller et al., 2010)). A new on-screen stimulation module that is coupled with a BCI and also usable for nearly all graphics application based on OpenGL would increase the usability of the whole BCI system. Also the consideration of alternative stimulation techniques like in (Sutter, 1992) and (Bin et al., 2011) may increase the BCI performance with respect to continuous controlled devices or avatars.

1.6. Goals

Usually, SSVEP based BCIs use stimulation, in which a light source flashes with a constant frequency. Such a light source could be a LED or a computer screen. At the presence of SSVEPs, the brain waves derived from the scalp contain sinusoidal signals with the frequency of the visual stimuli. An alternative type of stimulation based on code sequences instead of constant periods was presented by (Bin et al., 2011, Bin et al., 2009b) and (Sutter, 2001, Sutter, 1992). Both types of stimulation should be implemented and compared concerning their performance. In a former work at g.tec an SSVEP based BCI using Fast Fourier Transformation (FFT) for PSD analysis and Minimum Energy (ME) combination was implemented (Prueckl and Guger, 2010). A linear discriminant analysis (LDA) was used to calculate a classifier for target identification. As the ME and LDA algorithms are already implemented for online and offline analysis, a reimplementation of these tools is not necessary for this work.

The aim is to build a BCI using SSVEPs and VEPs based on stimulation with pseudo-random sequences that can be used to control continuous processes with visual feedback. For that reason a system is required that allows the user to change the current task after arbitrary time of execution. The system should provide the user the possibility to switch the task every 200 ms. Also the latency of the system should not exceed 3 s. The latency depends on the length of the EEG buffer used for signal processing and the applied temporal filters. This is important, as the buffer length may affect the classification accuracy. A low accuracy of the system would decrease the maximum ITR that may be achieved.

For switching between the different states, the system must be able to provide a sufficient number of controls. In this work at least four commands are necessary (forward, backward, turn left and turn right), to steer an e-puck robot (Mondada et al., 2009). The user should be able to move a robot along a given route only by using the BCI and a visual feedback channel. The system should offer the possibility to increase the number of possible commands, if necessary.

The BCI requires a visual stimulation unit to evoke the corresponding EEG signals. In this work an on-screen module should be realized that allows stimulation within any OpenGL based graphics application. Here, the host application should provide visual feedback of the moving robot. For this, a unified interface has to be defined, which allows visualization of BCI controls and communication with the BCI. The design of the stimulation module should also give the possibility to load and unload BCI controls during runtime, so that the user can switch on and off the module, without restarting the host application. For portability reasons, the module should be realized as a dynamic linked library (DLL) in MS Windows (Microsoft Corporation, Redmond, USA) or as a shared object (SO) in LINUX.

2. **Methods**

2.1. System Overview

The complete experimental setup is shown in Figure 7. The user is sitting in a comfortable chair in front of the computer screen and the BCI controls and has to steer a robot along a given route. A computer screen provides video feedback of the robot's movement via a video camera above the route. This video system was implemented by the Technical University of Munich (TUM) and contains a software package to visualize the video stream (Video-Client and Video-Server) coming from a camera for image recording with 60 frames (section 2.3.5). The same monitor also includes either eight LEDs (affixed to the monitor) or four on-screen BCI controls (depending on the configuration). The user has to gaze at either the LEDs or the on-screen targets to perform actions. To manage on-screen stimulation in combination with visual feedback, the BCI-Overlay module was implemented (section 2.4). It is a dynamic library that other graphics applications can include at runtime to embed BCI controls within the displayed image. The module uses an UDP connection, to communicate with the BCI model. This allows a separation of the stimulation module and the BCI model.

The recorded EEG data from the user is sent to the online BCI based on MATLAB/Simulink (g.BCIsys, g.tec medical engineering GmbH, Austria). All signal processing, feature extraction and classification is performed with the BCI system g.BCIsys (section 2.6 and 0). It is based on the strategy of rapid prototyping, which is presented in (Guger et al., 2001).

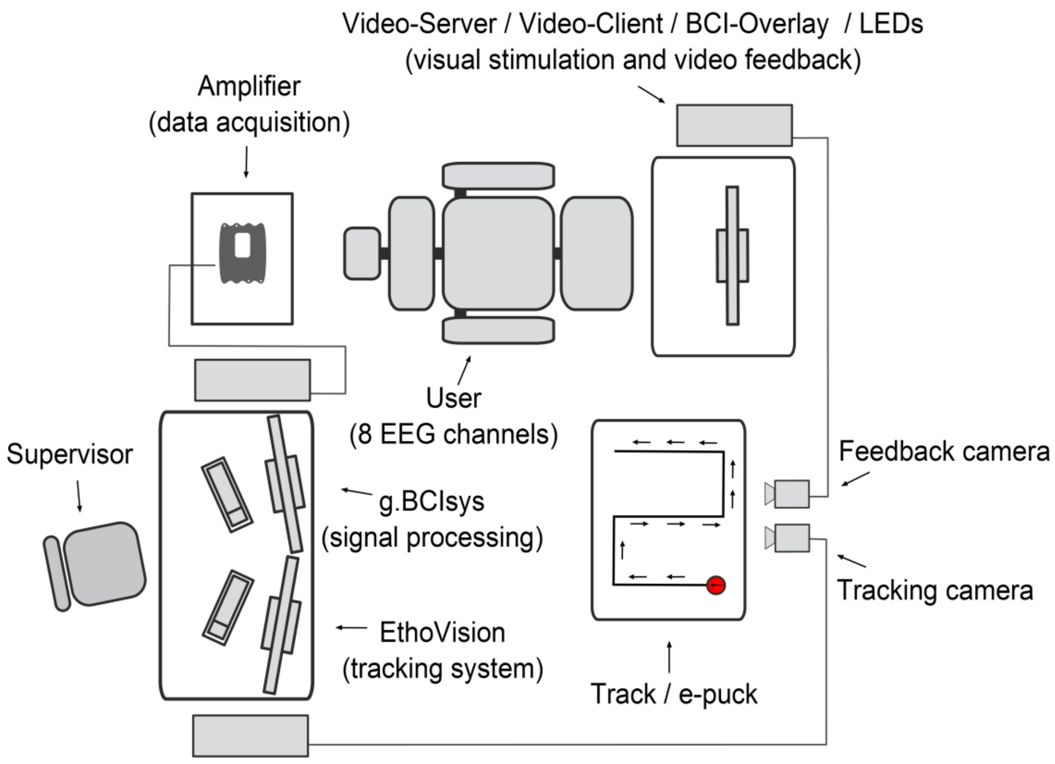

Figure 7: Complete experimental setup for the robotic control experiments. The user has to gaze at the monitor, which presents the video feedback from the Video-Client. To elicit relevant brain patterns, either LEDs on the border of the monitor or on-screen targets are used. The Video-Client includes the BCI-Overlay for visual stimulation and presentation along with the visual feedback received from the video server. The g.BCIsys represents the BCI model analyzing the EEG signals online and it sends the classification result to the robotic device (e-puck). The EthoVision tracking system is used to analyze the movement of the robot and evaluate the performance.

Within this work, three BCI configurations according to Table 1 were implemented and compared to find a suitable setup for continuous control of a robot. The system based on frequency coded stimulation with LEDs is called f-VEP LED, the system using on-screen frequency coded stimulation is called f-VEP on-screen and the system using code modulated stimulation is called c-VEP on-screen (Bin et al., 2009b). The f-VEP LED configuration uses eight LEDs (4 flickering LEDs for the directions and 4 small green LEDs to instruct the user which LED to designate as the target).

All f-VEP configuration use a ME combination for feature extraction and an LDA based classifier. The c-VEP configuration uses features based on correlation coefficients extracted through a CCA algorithm, followed by an LDA based classifier.

Table 1: This table presents the three BCI configurations used in the three conditions in this work. The c-VEP differs from the others in terms of the visual stimulation and the signal processing. The two f-VEP setups allow us to evaluate and compare the performance of the on-screen and LED stimulation methods.

Name	c-VEP on-screen	f-VEP on-screen	f-VEP LED
Visual stimulator	60 Hz Display	60 Hz Display	LED
Stimulation type	code-based	frequency-coded	frequency-coded
Stimulation sequence	63 bit M-sequence	constant periods	constant periods

In all configurations, the user can select a command by attending to one stimulus out of four (forward, backward, turn left, turn right). If the classification gives no confident result, the BCI remains in idle state, which is represented by the so called pseudo zero class. Otherwise, classification results are sent to the e-puck robot via the Bluetooth connection (section 2.2.6). Finally, the system uses the EthoVision (Noldus, Wageningen, Netherlands) tracking system and a camera to record the movement of the robot for accuracy calculation and to measure the time to complete the task.

2.2. Used Hardware

2.2.1. Biosignal Acquisition Device

The g.USBamp (g.tec medical engineering GmbH, Schiedlberg, Austria) is a biosignal amplifier that used during this thesis. The amplifier allows the acquisition of biosignals like EEG, EOG, EMG and ECG. A 24 bit analog-digital converter (ADC) is built in for each channel and acquires data with a maximum sampling rate of 38400 Hz. In total 16 channels are available on one device. They are arranged in groups of 4 channels, where each group provides one ground and one reference channel. Each of the A/D converters is 64 times oversampled compared to the 38400 Hz sampling rate. The range for incoming signals is +/-250 mV, with a resolution below 30 nV with respect to the 24 bit A/D conversion. The input impedance of the amplifier is higher than 10^{10} Ω. An application programming interface (API) in C provides access to the g.USBamp via MS Windows and LINUX operating systems. A MATLAB/Simulink interface is available that allows to configure the channels and to define filters (e.g. high-pass, low-pass or notch-filter) (g.tec, 2011d).

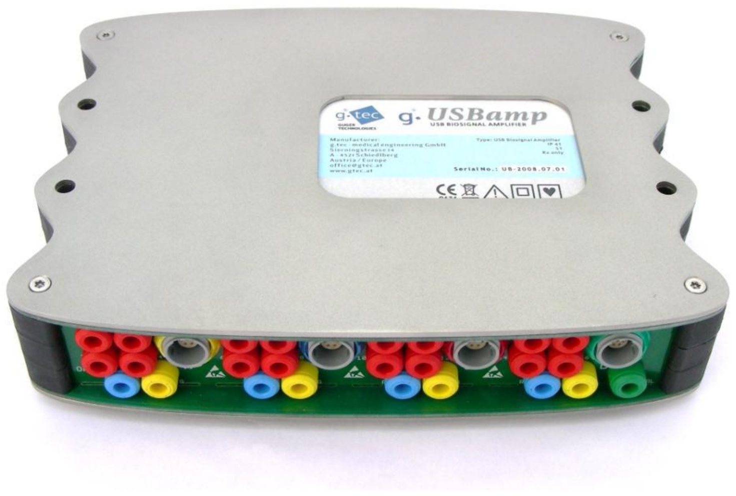

Figure 8: The g.USBamp from g.tec offers 16 DC-coupled input channels in 4 independent groups, 8 digital trigger inputs and 4 digital outputs.

2.2.2. EEG Cap and Electrodes

For EEG recordings the electrodes have to be placed on the scalp. A cap, like the g.GAMMAcap2, provides a comfortable way to keep the electrodes in place (Figure 9). In total, 74 positions are labeled on the cap, with respect to the international 10-20 system (g.tec, 2011a).

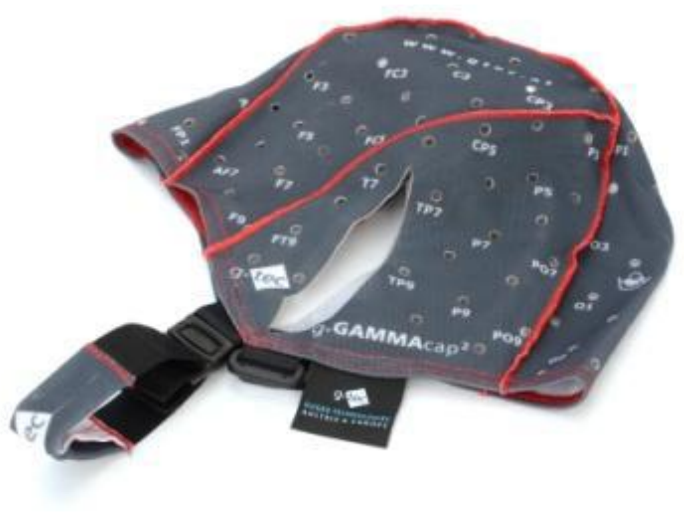

Figure 9: The g.GAMMAcap2 from g.tec. A chin-belt is used, to fix the cap to the users' head.

In this work, the cap is used in combination with active g.LADYbird electrodes consisting of a sintered Ag/AgCl crown. The ground electrode is a passive g.LADYbirdGND. As reference electrode an active ear-clip electrode is used (Figure 10) (g.tec, 2011e). Because of the usage of active electrodes, no abrasive gel is required. Nevertheless, conductive gel is necessary to connect the electrodes with the skin surface. To connect the electrodes to the amplifier a driver box, the g.GAMMAbox, is necessary. This driver box provides up to 16 signal channels, 1 reference channel and 1 ground channel (g.tec, 2011b).

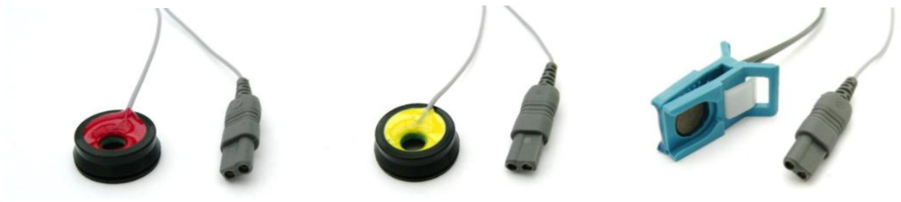

Figure 10: g.LadyBIRD electrodes. From left to right: active electrode, passive ground electrode and active ear-clip electrode.

2.2.3. Driver Box for Visual Stimulation

For visual stimulation with LEDs a stimulation device is needed, which controls the light sources. Therefore, the g.STIMbox, a digital I/O box that provides 14 inputs and 16 outputs is used. It is connected to a PC via USB and is accessed through a MATLAB/Simulink interface. The corresponding Simulink block allows defining the stimulation frequencies, which are available on specific output ports of the g.STIMbox. Connected to USB, the g.STIMbox provides 5 V with the maximum of 200 mA overall output current (g.tec, 2011c).

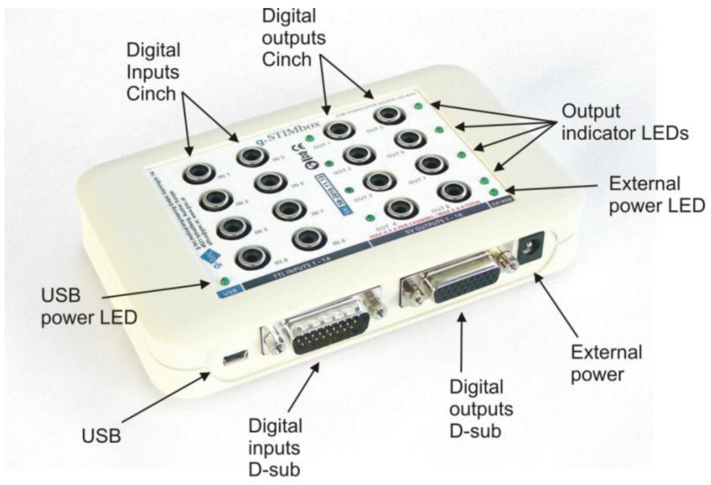

Figure 11: The g.STIMbox (left) and a single LED for visual stimulation (right).

2.2.4. Photodetector

The g.TRIGbox is a pulse generator that allows triggering light flashes in combination with a photo-sensor of the type: Silicon NPN Phototransistor, SFH 309 P (OSRAM, Munich, Germany). Each trigger pulse has an output Voltage of 200 mV and duration of 20 ms. An output port of the g.TRIGbox is compatible with a digital input port of the g.USBamp. The minimum duration of 20 ms of the trigger pulse is caused by a sample and hold circuit. Therefore, the slew rate of the voltage change is much shorter than the whole trigger pulse. Input voltage from the sensor: minimum 0.5 mV (low level inputs) and maximum 5 V (high level inputs) (g.tec, 2011e).

The g.TRIGbox is used to measure the flickering of the light sources. This allows an evaluation of the flash accuracy of the visual stimulation.

2.2.5. Noldus Tracking System

EthoVision® (Noldus, Wageningen, Netherlands) is a tracking system, which is specialized on movement tracking of animals. During the experiments, the system was used to track the movement of an e-puck. Therefore, a video camera is required, which is connected to the computer via a Picolo Diligent (Euresys, Angleur, Belgium) video-grabber board. A calibration has to be performed, before the tracking system is ready to use. During the calibration, the user has to define an arena and a scale. Only within this arena, the tracking is valid of the moving object. The scale allows the system to store the tracking data in real-world units. After calibration, the user has to define a point on the object that represents the tracking point, for which EthoVision stores the coordinates with a corresponding time stamp.

2.2.6. Robot (e-puck)

The education robot or e-puck is a small robot – its diameter is 75 mm – developed at the Ecole Polytechnique Fédérale de Lausanne (EPFL) (Figure 12). The device consists of a microcontroller including a 16 bit processor running at 64 MHz and a digital signal processor. Two step motors allow the e-puck to move. To communicate with an external device, a Bluetooth radio link and a RS232 serial interface can be used. Moreover, the robot consists of several sensors and extensions, which are described in detail in (Mondada et al., 2009).

In this thesis, the e-puck was used to move according to commands from a BCI. A Simulink S-function was used to establish a Bluetooth communication channel to the robot, which was implemented in a former project at g.tec. The resultant Simulink block generates the commands STOP (0), FORWARD (1), RIGHT (2), LEFT (3) and BACK (4), dependent on the selection of the BCI.

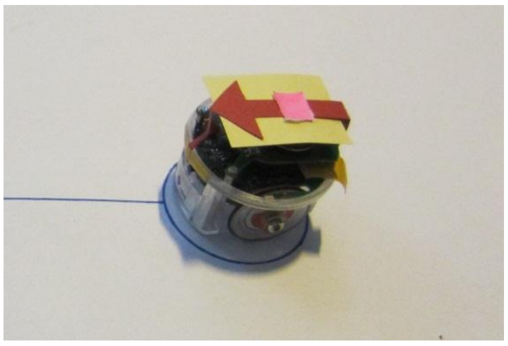

Figure 12: The e-puck is a small robot, which can move through two independently controlled wheels. The figure shows the robot that was used in the presented experiments. The red arrow informs the user about the current direction of the robot. For tracking purposes, the pink square was placed on the center of the e-puck.

2.3. Used Software and Libraries

2.3.1. MATLAB/Simulink

MATLAB (The MathWorks Inc., Natick, United States) is a high-level programming language that provides an environment for algorithm development, signal processing and numeric computation. It is developed to perform matrix operations. Toolboxes allow expanding the functionality, for a quick implementation of specific algorithms or programs. The programmer can solve computing problems in less time for development than with C or other programming languages (MathWorks, 2011b). MATLAB interprets scripts and therefore its usage is limited for tasks that need high computing performance. This is why MATLAB is able to execute C or C++ programs within so called MEX-files (MATLAB-executable) (MathWorks, 2011a).

Simulink (The MathWorks Inc., Natick, United States) is a tool for model based system design, simulation and analysis. It provides a graphical user interface (GUI), which allows the programmer to “draw” the model as a block diagram. Both discrete and continuous can be performed, as well as linear and non-linear behavior of the system. Simulink requires MATLAB to run and is able to use variables as input signals or to call MATLAB functions. Various models can be used together or as sub-systems. This provides a highly flexible environment, in which the programmer can define and change models in a comfortable way (MathWorks, 2011d).

A simulation in Simulink takes place in several stages. In the initialization phase the required libraries are loaded, data types, block parameters are defined and initial values are assigned. After initialization, the engine calls the update, derivative and output function of each block and for each stimulation step (MathWorks, 2011c).

With the help of so called S-functions, parts of the model can be defined outside of the Simulink environment. An S-function may be written in MATLAB, C, C++ or FORTRAN and is integrated into the simulation, using a specific calling protocol. This requires that an S-function provides embedded function blocks. An S-function that is written in C, has to define the following functions, which will be called by Simulink: *mdlInitializeSizes*, *mdlInitialize SampleTimes*, *mdlStart*, *mdlOutputs*, *mdlUpdate* and *mdlTerminate*. In addition, optional functions can be defined. The functions *mdlOutputs* and *mdlUpdate* will be called for every time step, within the simulation loop (MathWorks, 2011c).

2.3.2. Signal Processing Tools

As mentioned in section 2.2.1, g.tec provides software to access the g.USBamp. The g.HIsys software package contains a Simulink block set for tuning the amplifier settings. This allows online processing of the acquired data, within a Simulink model. Therefore, g.tec provides the a Simulink libraries called g.RTanalyze and g.Highspeed (g.tec, 2011e).

In this work, the offline analysis of pre-recorded data is mainly performed with the help of g.BSanalyze. This tool provides an environment for biosignal processing and contains spatial and temporal filters, spectral analysis, correlation, classification and visualization tools. g.BSanalyze can either be called from MATLAB or used as a standalone program (g.tec, 2011e).

2.3.3. UDP Interface

The User Datagram Protocol (UDP) is a minimized network protocol that does not require prior arrangement to send packages from one network endpoint to another one. It was introduced in 1980 and allows fast communication within a network. A UDP package contains sender and destination address with the corresponding port, the package length, a checksum and the data (Postel, 1980). The protocol does neither perform an initiating handshake nor can it guarantee that the package ever is received. Therefore the programmer has to be careful using UDP.

The g.UDPinterface allows the setup of a UDP connection and to transmit or receive data. It can either be used as a C based DLL or in MATLAB/Simulink. In all versions four functions are necessary for remote communication: *gUDPinit*, *gUDPrecv*, *gUDPsend* and *gUDPclose*.

2.3.4. OpenGL

This section provides an overview of the Open Graphics Library (OpenGL), which is useful for a better understanding of the technical details of the BCI-Overlay. OpenGL is a standardized software interface, which provides access to graphics hardware. It allows the user to set up a model out of geometric primitives like points, lines and polygons. OpenGL is a state machine with various state variables that define the behavior of the currently drawn primitives. A change of the state does not affect previously drawn objects.

The state information is stored in the so called OpenGL context, which is managed by a window-system like MS Windows or Ubuntu based on a Debian LINUX distribution. In addition, the contexts contain objects like textures, display lists, etc. New primitives are always drawn within the current active context.

OpenGL allows displaying a 3D scene on a 2D screen. This is achieved by viewing-, modeling-, projection-and viewport-transformations. The viewing-transformation sets the view-point to the scene its orientation. The modeling-transformation gives the programmer the possibility to translate, rotate and scale the model (e.g. an asymmetric scaled cube leads to a rectangular box). To specify the field of view and the way of how to look at the scene, the projection-transformations are used.

Similar to the color representation of a computer screen, OpenGL uses the Red-Green-Blue-Alpha (RGBA) model. The color values represent the saturation of each color. Together, all three colors lead to an additive color impression. The alpha value can be used for transparency. All color values are stored in the color buffer, for every single pixel.

OpenGL supports the combination of source and destination color values, which is known as blending. Here, the alpha value can be used to weight the source and destination values. This leads to a transparency effect of the combined pixel. Of course, there are many possibilities, on how blending functions can be used (e.g. weighted overlapping of images, implementation of color filters, inverting the color values…).

Display lists can be used, to increase performance. They store precompiled command sequences for later use. Multiple calls of the same commands can be rendered more efficient than repeated individual calls. Moreover, some graphics hardware stores the commands in a form, which is optimized for the used instructions. Display lists are also used to store Fonts. Fonts define the appearance of a set of characters on the screen (e.g. in ASCII format).

Texture mapping is another performance increasing feature of OpenGL. Instead of defining a surface structure with a vast number of primitives, a texture can be used as a single object. A texture underlies the same transformations as the textured primitive. Textures are stored as simple data arrays containing color and alpha values and can be used for all primitives. Therefore textures can have various dimensions (1D, 2D and 3D).

A detailed and well explained specification of OpenGL functions and techniques can be found in (Shreiner, 2009).

2.3.5. VideoClient

The VideoClient is an OpenGL based application provided by the Technical University of Munich (TUM). It is used to remotely display received video streams, recorded by up to two cameras (each with a resolution of 640x480 pixels). A VideoServer program is used to acquire video data. The VideoClient is implemented in C++ using the Qt framework and is split into a thread for each display, to ensure a 60 Hz refresh rate of the video stream. This allows the system together with the two cameras to visualize the recorded scene in 3D. For stereoscopic view, for example in a head-mounted-display (HMD), the VideoClient starts an OpenGL context for each view. Within this work only one camera was used to visualize visual feedback on a single Monitor in 2D.

2.4. BCI-Overlay

The BCI-Overlay is a runtime loadable module based on OpenGL. It is implemented as a dynamic linked library (DLL) for MS Windows and as a shared object for LINUX and can be used by OpenGL based host applications, to embed targets for visual stimulation within the displayed scene. The host applications could be virtual reality environments or real-world videos, acquired with a camera. The BCI-Overlay works as a visual stimulator and can be used in combination with a biosignal analyzing system like a BCI. Figure 13 shows an example for the usage of the BCI-Overlay, as it is used for the presented experiments.

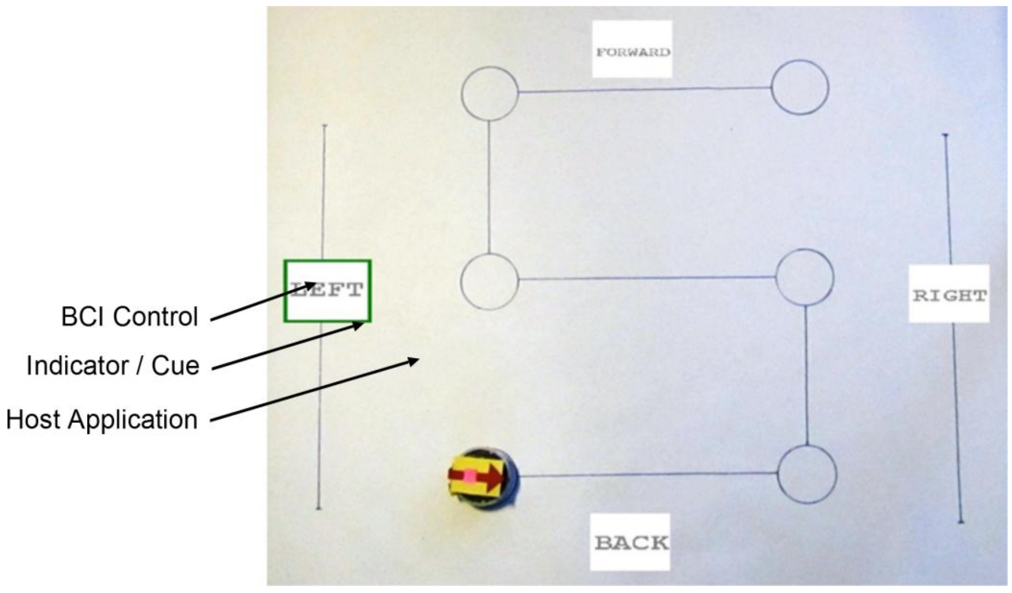

Figure 13: Example for the usage of the BCI-Overlay. Four BCI controls flash together within a running video application (Video-Client). The spatial stimulation type is set to SOLID flickering and the temporal type could either be f-VEP or c-VEP. In this example, the user has to control a robot only by using the BCI and a visual feedback on the screen. The green border is an indicator and shows the user the active task. A task is active as long as the user gazes at the BCI control and the classification result of the BCI model is valid.

During the initialization phase, the user can set several options of the BCI-Overlay via network commands. First, there is the choice between two temporal stimulation types: f-VEP and c-VEP. During f-VEP stimulation each control flashes with a specific frequency.

Therefore, the BCI-Overlay sends the number and values of possible frequencies. The c-VEP system uses a stimulation sequence containing high and low frequencies. As each control uses a phase shifted version of the same sequence the BCI-Overlay sends the length of the sequence to the BCI. Together with the number of displayed controls and the sequence length, the expected phase shift can be calculated. The second option is the spatial stimulation pattern: CHECKERBOARD, SOLID or TRANSPARENT. In CHECKERBOARD mode each BCI control is formed by an 8x8 matrix of alternating black and white quads. More details of stimulation with pattern reversals can be found in section 1.3.4. The SOLID type can be seen in the example in Figure 13. Here, controls switch between black and white above a homogenous area. If other shapes than quads or rectangles are required, the user has to choose TRANSPARENT mode. Only the area defined in a bitmap is flickering in solid mode, the remaining part of the quad or rectangle is transparent.

The BCI-Overlay provides an indicator, realized as a border around the BCI control. It is either used to provide the user with a feedback about selected command or to direct the user's gaze to a specific target (e.g. during training or feedback run).

In case no bitmap is available for a flickering target, a text can be defined within the rectangle (Figure 13). Controls containing a bitmap do not support checkerboard stimulation. The text and the bitmaps for all controls have to be defined in the mask-file. If has neither a text nor a shape defining bitmap, it will be a solid rectangle.

2.4.1. Technical description

The implementation of the BCI-Overlay was done in object-oriented C++. The BCI-Overlay is defined within a class named *BCIOverlay* and is instantiated once at the start of the program. This approach is based on the principle of singleton-pattern (Gamma et al., 2004). Two threads are running within *BCIOverlay*. The first one is the display thread of the host graphics application and handles all the initialization and drawing commands. Therefore, all communication between the graphics application and the BCI-Overlay takes place in the drawing thread. The second thread is the network thread, which manages the UDP based communication with the BCI (see section 2.5). Triggers are the only exception and can be sent out of the drawing thread. It should be mentioned that UDP packages may not reach the receiving side. Therefore, the standard commands have to be acknowledged. Concerning trigger commands, it is only allowed to use the command for synchronization of the stimulation with the BCI. In case of a lost package, the last synchronization event is longer ago than usual, but this has no effect on the performance of the system.

The communication between the two threads is based on state flags and a local loopback socket or a so called socket pair. The socket pair is used by the drawing thread, to inform the network thread that states have changed (Figure 14). This is necessary, as the network thread is only activated by incoming network commands.

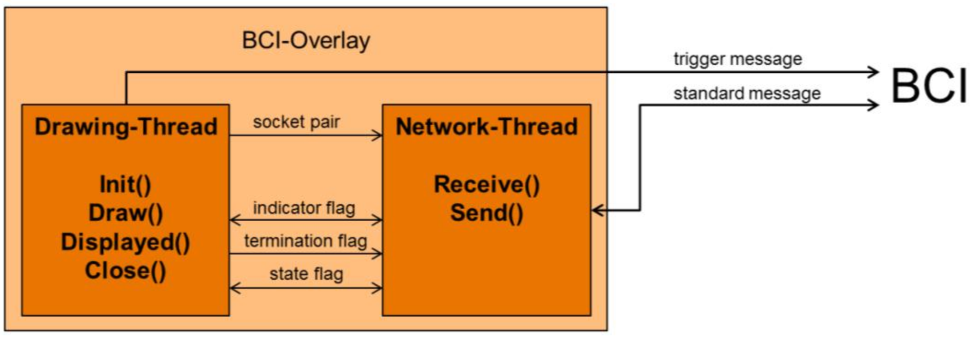

Figure 14: The BCI-Overlay works with two simultaneously running threads. One thread is the drawing thread of the host graphics application. The second thread manages the UDP based communication with the BCI. The threads communicate through state flags. In addition, a local loopback socket is used to awake the network thread.

The host graphics application has to initialize and control the BCI-Overlay. This is realized with the help of an abstract interface class that provides the following public functions: *Init*, *Draw*, *Displayed* and *Close*. All of them are implemented within *BCIOverlay* class.

2.4.2. Drawing Thread

The *Init* function registers the current active context and thereby ensures that the BCI controls appear only within registered contexts. The graphics application has to provide several parameters, such as the screen refresh-rate, the path of the mask-file and network parameters (local IP address, local port, remote IP address and remote port). Within the *Init* function, the socket for the UDP connection is established and the network thread is initiated. In case the BCI controls should be displayed on multiple screens Due to the possible usage of more than one drawing context, the *Init* function has to be called once for each context. It takes care that the network connection is established only once, even for multiple calls.

Changes in the state of the BCI are handled by the *Draw* function. If the BCI is active the OpenGL commands are executed and the BCI controls begin to flash, otherwise nothing will be displayed. All drawing operations are performed in the relative area from -1 to 1 in x and y direction. Therefore, the size of the BCI controls can be defined in relative values too. The depth (z-direction) is always set to zero. The host application has to setup the appropriate transformations of these coordinates, to define the position of the BCI controls in the scene. To increase the performance most of the drawing commands are stored in display lists, which are redefined after each change of the BCI controls. This update process is done in the *Draw* function, to avoid conflicts during drawing. All OpenGL state variables are stored at the beginning of the function and will be reset at the end of the function. The reset guarantees, that the OpenGL states stay unchanged in the host graphics application. As the BCI controls are flat areas, the depth test is disabled during drawing, to avoid artifacts caused by overlapping primitives. Because of disabled depth test, the visible data depends on the order of the drawing calls. The lighting state is set inactive, so that the BCI controls always flash with the same intensity and the same colors.

Textures are used to draw arbitrary shapes on base of bitmap images and to write text into the drawings. Each BCI-control consists of a rectangle and a texture. An alpha test of the textured primitive ensures that the BCI control is transparent for undefined regions.

After each call of the *Draw* function, the host application must call the *Displayed* function. This function sets the display list for the next frame. Therefore, all initialized contexts have to flash synchronously. The next display list is loaded after every context has called and finished the *Draw* function. Whenever a state change of the module has to be reported to the BCI, the local loopback socket is used to force the network thread sending the command. Only the trigger command is sent directly from the *Displayed* function, to minimize the delay between sending and receiving.

At termination of the host application, the *Close* function has to be called for every registered context. As long as more than one context is still active, the *Close* function will only unlink the context. If the last context is closed, the function releases the loopback socket and sets a flag, to terminate the network thread.

2.4.3. Network Thread

Except triggers, all network communication is handled by the network thread. A list of all valid commands is shown in Table 3 in section 2.5. This section focuses on the state changes in *BCIOverlay* and the definition of the BCI controls. A list that is describing the states used by *BCIOverlay* can be found in Table 2.

Table 2: States of *BCIOverlay*. The referred commands in the description are all network commands and should not be mistaken with any internal functions.

State	Description
WAITING	This is the initial state at the start of the program. Any incoming UDP command will be ignored exempt the Init and the End commands. This state is set, whenever the connection the BCI is closed.
CONNECTED	As soon as the BCI-Overlay receives an Init command, it changes its state to CONNECTED. Therefore, CONNECTED can be reached from any state. It is also used as a temporary state, during the extraction of new BCI controls from the mask-file.
CHANGED	If an UpdateMask command is received, the BCI-Overlay sets the CHANGED state. This state can only be reached from ACTIVE, READY and CONNECTED. The BCI-Overlay stops drawing and the BCI controls will be redefined.
READY	This state is reached automatically, after successful update of the BCI controls and the corresponding display lists and textures. The only possible previous state is CHANGED.

The network thread ignores any UDP command unless it originates the remote IP address and port, which were registered at startup. Once the network connection is established, *BCIOverlay* switches into the CONNECTED state and waits for an UpdateMask command, to define the BCI controls. An update command provides the path to a mask-file, which is used for parsing the BCI controls within the network thread. Afterwards *BCIOverlay* is in CHANGED state and begins to define the display lists and textures within the drawing thread. Finally, the state changes to READY, when each registered context has finished the update. Now, the *BCIOverlay* can switch to ACTIVE, if a Run command is received. A state diagram of *BCIOverlay* can be found in Figure 15.

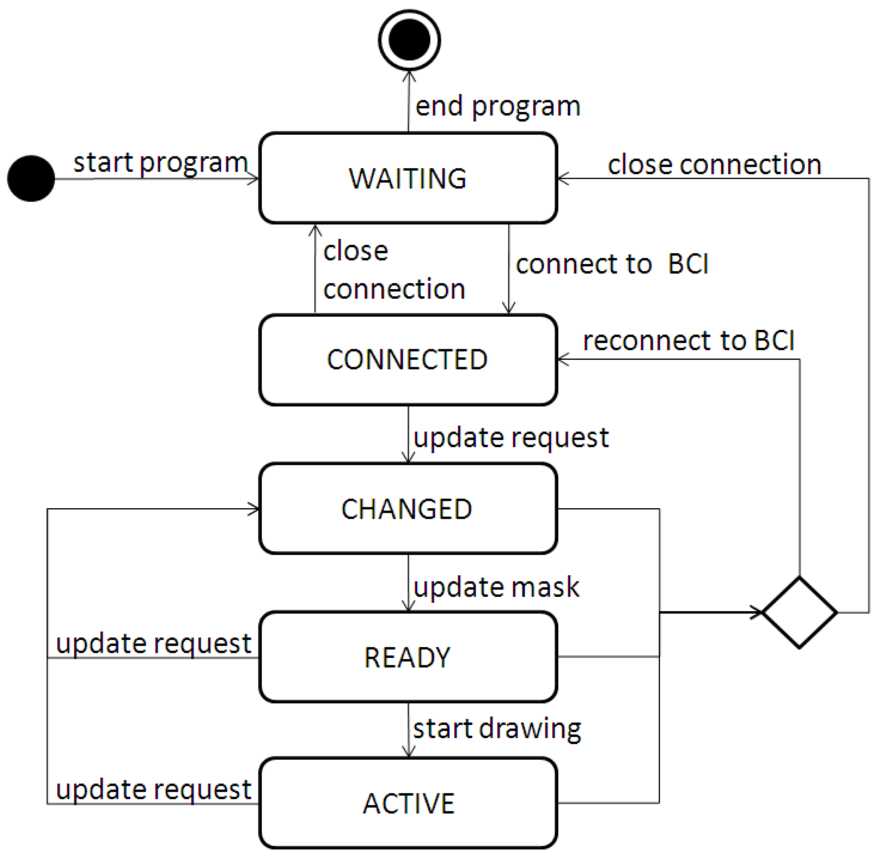

Figure 15: State diagram of *BCIOverlay*. The initial state after program start is WATING. After connection to the BCI, *BCIOverlay* changes to CONNECTED state. Out of this state *BCIOverlay* can define BCI controls and runs through the states CHANGED and READY. A run command from the BCI sets *BCIOverlay* to ACTIVE state.

The network thread remains sleeping and listens to the local port for incoming c. An incoming command awakes the network thread and the command is interpreted. After execution of the corresponding commands, all commands in the sending queue will be sent to the BCI. If no command is received, the thread leaves the receive function after two seconds. Also the loopback socket can awake the network thread, just by sending an arbitrary character. This is used for example during updating, when a “re;;” command is added to the command queue and the loopback socket awakes the network thread, to send the command. As long as there are commands available on the listening socket, the network thread will process and execute the corresponding commands before going to sleep.

2.5. Network Communication between the BCI and the BCI-Overlay

The BCI-Overlay and the MATLAB/Simulink BCI-model use UDP packages to communicate with each other. The packages are strings, which are split by the separator character “;”. Two consecutive semicolons (“;;”) indicate the end of the command. Commands are only valid, if the source IP and the source port is registered at the destination. The network protocol is used to initialize the BCI-model and the BCI-Overlay, as well as for sending trigger commands. Table 3 lists the valid commands and their possible parameters .Definition of a command: <command>;[<param_1>;<param_2>;…;<param_n>;];.

Table 3: Commands, which are used by the BCI model and the BCI-Overlay to communicate with each other.

Command	**Parameters**	**Direction** **BCI ↔ Overlay**	**Description**
in	-	↔	"Init": Initialize Connection.
gp	-	→	"Get Paradigm": Request the list of supported Paradigms.
lp	*1-END: List of Paradigms*	←	"List of Paradigm": The BCI-Overlay returns a list of supported Paradigms.
gf	*1: Name of Paradigm*	→	"Get Features": Request the settings for a specific Paradigm. *Name of Paradigm*: SSVEP
fl	*1: Name of Paradigm* *2: c-VEP length* *3: # of frequencies* *4-END: List of frequencies*	←	"Feature List": The BCI-Overlay returns a list of features for the chosen Paradigm.
um	*1a: Name of maskfile* *1b: "-"* *2: Indicator ID*	→	"Update Mask": Stop visual stimulation and update the controls specified by the *maskfile*. In case the indicator has to be set or unset, then the filename is has to be set to *"-"*, to prevent from stopping the visual stimulation
re	-	←	"Ready": The mask change is done; the Overlay is ready to flash.
ru	-	→	"Run": Start the visual stimulation.
rn	-	←	"Running": BCI-Overlay is stimulating.
en	-	↔	"End": Terminates the connection.
er	*1: Error command*	↔	"Error": *Error command* contains the description of the error.
tr	*1-END: List of IDs*	←	"Trigger": Contains a list of the IDs, which have flashed since the last display refresh cycle.
sf	*1: Name of Paradigm* *2:SSVEP-Mode* *3: BCI-Mode*	→	"Set Flash-Mode": Sets the flash mode for certain Paradigm. *SSVEP-Mode* (default mode: 2): 1…solid quad, 2…transparent bitmap, 3…checkerboard If no bitmap is available for a target, mode 1 is used for this target. *BCI-Mode* (default mode: 1): 1...f-VEP, 2...c-VEP
fs	*1: Name of Paradigm*	←	"Flash-Mode set": Acknowledge for the "Set Flash-Mode" command.
iu	-	←	"Indicator Updated": The Overlay signals that the Indicator has been updated, in response to a "um;-;<ID>;;" command.

To establish a connection between the BCI model and the BCI-Overlay, both modules have to be active. If one module is activated, it sends an “Init” command to the other one. Only in case the receiver answers with an “Init” command, in response to an incoming “Init” command, the modules are connected. The BCI model is controlled by the Simulink engine and starts within the initialization phase. On the other side the BCI-Overlay is driven by the Video-Client and thus, it is ready as soon as the Video-Client has been fully initialized. Once the network connection has been established, the BCI model and the BCI-Overlay exchange information concerning the paradigm and the corresponding stimulation parameters, such as frequencies or the temporal stimulation sequence. Afterwards, the BCI model quits the initialization phase and waits until the simulation loop of the Simulink engine starts.

Before the BCI-Overlay is able to start the visual stimulation, it has to be informed by the BCI model about the used controls. This is done by the “UpdateMask” command that contains a path of the mask definition. Moreover, the “UpdateMask” command can be used to stop the simulation and also to set the indicator control. The indicator shows the user the current control, which can be used during training to direct the users gaze and also in free running mode provide the user a feedback about the classification result. After the successful definition of the controls, the BCI-Overlay is ready to start the visual stimulation. The BCI model sends the “Run” command to initiate the visual stimulation by the BCI-Overlay. During stimulation a trigger command provides information about the phase of flickering.

If either the BCI model or the BCI-Overlay terminates, the terminating side sends an “End” command. A new start requires again an initialization phase. The sequence diagram in Figure 16 shows an example, for the communication between the BCI model and the visual stimulation module.

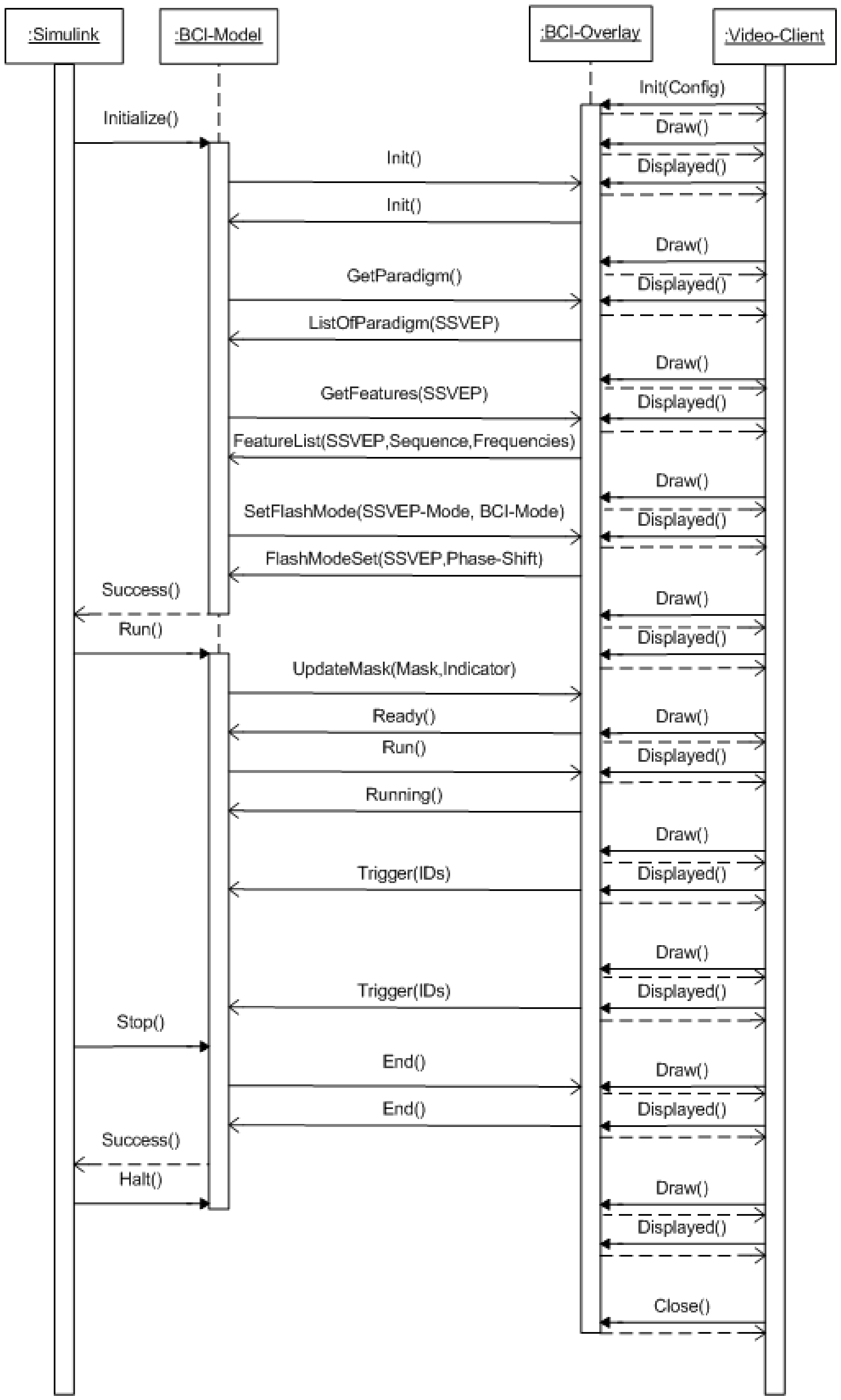

Figure 16: Sequence diagram of the interaction between BCI model, the BCI-Overlay and the Video-Client. The BCI model is based on Simulink and therefore controlled by the Simulink calls. The BCI-Overlay runs within the Video-Client. The BCI model and the BCI-Overlay communicate via UDP commands. A dashed arrow indicates the return of a function call.

2.6. Visual Stimulation

The visual stimulation is either accomplished through LEDs (Figure 17b) or by using the BCI-Overlay in combination with a computer screen (Figure 17a). For comparative reasons, white stimulation color and an identical spatial pattern is chosen for all configurations. Because of the homogenous on/off stimulation of the LEDs, the on-screen solutions use white rectangles without any spatial patterns.

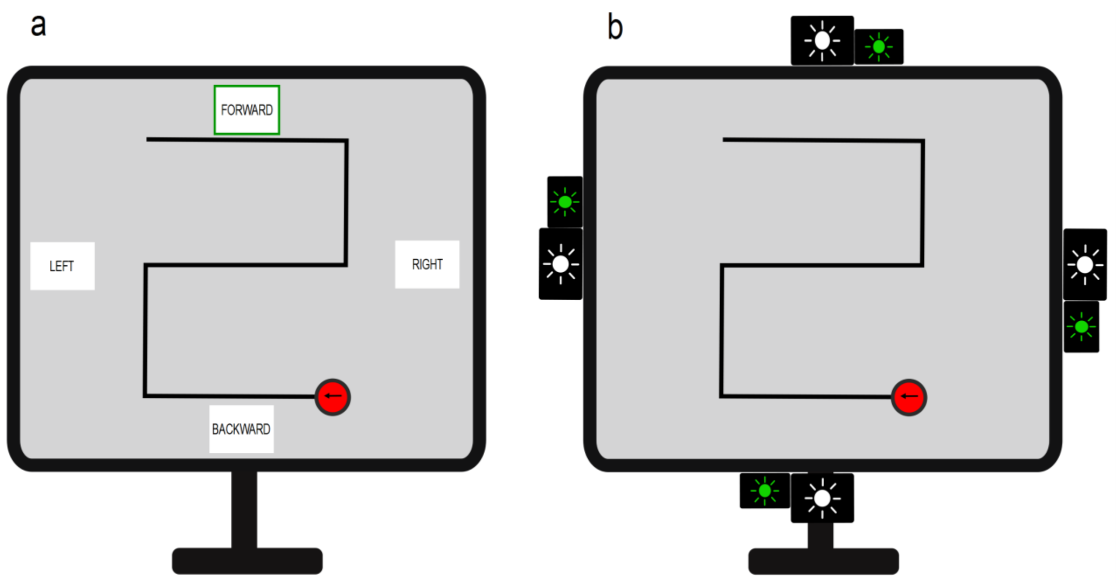

Figure 17: User interface for controlling the robot with the SSVEP based BCI and the video images of the track and the moving robot. (a): Screen stimulation with 4 targets to move forward, backward, left and right, which was used for f-VEP on-screen and c-VEP on-screen. The camera picks up the robot (red circle) and the track that the subject should follow with the robotic device. The green frame shows either the current classification result or (in training mode) the current target. (b): The f-VEP LED configuration for the experiment with four LEDs mounted on the screen frame. The four green LEDs indicate the current target LED during training.

Each on-screen BCI control fills an area of 4.2 cm x 3.2 cm corresponding to a visual angle of 3.0° x 2.3°, with respect to the distance of 80 cm to the screen. This relationship results from equation (3) (Kaiser, 1996):

$$\alpha = 2 \cdot arctan\left(\frac{S}{2 \cdot D}\right) \tag{3}$$

Whereas α represents the visual angle, S the size of the visual target and D the distance between the eye and the target. The display, which was used for the experiments, is a Belinea 1930 S2 with 300 cd/m² luminance and a reaction time of 5 ms. Due to the 13.44 cm² of a single flickering target, the luminous intensity is 0.4 cd.

For LED stimulation a white diode of the type WU-2-104WDwas used. Luminous intensity is 1.5 cd and the area covered by the LED is 0.57° x 0.57° visual angle (0.8 cm x 0.8 cm at a distance of 80cm). Therefore, the resulting luminance is 23437.5 cd/m².

The frequencies used for the f-VEP based BCIs are similar for LED and for on-screen stimulation. This allows a better evaluation of the BCI-Overlay performance. Both systems work with 8.57, 10, 12 and 15 Hz. As the frequencies used in (Volosyak et al., 2009a), also these frequencies do not overlap within the first two harmonics.

Stimulation of the c-VEP system is performed on-screen only. Therefore the BCI-Overlay is used to visualize stimulation sequences. The stimulation sequences are so called M-sequences. These binary sequences are used for non-linear signal analysis and also in multi-input systems (Sutter, 2001). Moreover they have an autocorrelation function, which is an approximation of the unit impulse function. This is very important, because of the used features of the c-VEP system are based on correlation coefficients. The used M-sequence in the implemented c-VEP based BCI consists of 63 bits, which leads to a duration of 1.05 s for a whole flash cycle. Every stimulating target uses a phase shifted version of the reference sequence. The phase shift depends on the number of targets and results in 0.25 s or 15 samples for four targets.

2.7. Electrode Setup

The EEG data is recorded using eight active electrodes from the scalp over the visual cortex. Figure 18 shows the used positions according to the international 10-20 system. The corresponding ground electrode is positioned on Fpz. The right earlobe is used for a reference electrode. Within the used feature extraction algorithms (ME combination and CCA), spatial filtering of the EEG channels leads to improved feature channels for the further processing. Therefore, multiple channels over a sufficiently large area above the visual cortex are required, to achieve the best results. The configuration used in this work is based on the setup of (Prueckl and Guger, 2010). This configuration is used for the f-VEP configurations as well as for the c-VEP configuration.

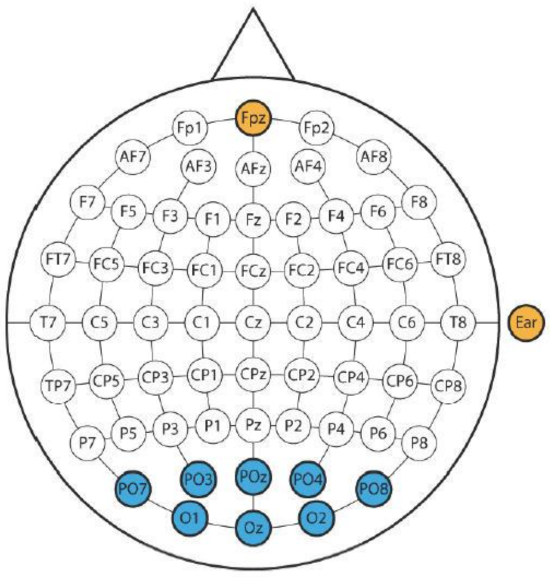

Figure 18: Electrode setup for VEP based BCI systems. The positions PO7, PO3, POz, PO4, PO6, O1, Oz, and O2 (blue channels) are the signal channels, derived with active g.LADYbird electrodes. A passive ground electrode (g.LADYbirdGND) is placed On Fpz (orange). An active earclip electrode is used on the right earlobe (orange) as a reference.

2.8. Signal Processing

2.8.1. Pre-Processing

The acquired EEG data is recorded with a sampling frequency of 256 Hz and then bandpass (f-VEP LED and on-screen: 0.5 – 60 Hz, c-VEP on-screen: 0.5 – 30 Hz) filtered and Notch (48 – 52 Hz) filtered to remove noise, baseline drifts and power line interference. Both f-VEP configurations use a higher low-pass to analyze also harmonic components of the stimulation frequency. The lower cutoff frequency is 0.5 Hz for every BCI system used in this work. A high value for the lower cut-off frequency may decrease the amplitude of the VEPs (Bach et al., 2005). In literature, common filter settings for VEP analysis are e.g. 1 – 250 Hz (Paulus, 2005) and 0.3 – 100 Hz (Bach et al., 2005).

2.8.2. Minimum Energy (ME)

The f-VEP configurations use features based on spectral analysis of the EEG. The extracted feature is the SNR between a target signal (power density of a specific frequency range) compared to the base EEG (power density of the estimated noise). Thereby it is assumed that the EEG is comparable to white or pink noise. The analyzed spectrum is not directly calculated from the derived EEG channels, but from combined channels that result from the ME combination. These channels are the result of a spatial filter operation and contain an improved SNR of the target signals. The weight vector w results from the following minimization problem in (4):

$$\min_{w} \left\| \tilde{Y} w \right\|^2 = \min_{w} \left(w^T \tilde{Y}^T \tilde{Y} w \right) \tag{4}$$

Thereby, $\tilde{Y}$ represents the EEG without the target signals. The cancellation of the target signals in the EEG takes place via equation (5):

$$\tilde{Y} = Y - X(X^T X)^{-1} X^T Y \tag{5}$$

Y is a matrix containing the pre-processed EEG derivations. X represents a model matrix, which consists of the sinusoidal signals of the target frequency and its higher level harmonics. A detailed description of this algorithm can be found in (Friman et al., 2007).

2.8.3. Canonical Correlation Analysis (CCA)

The CCA is used for multivariate variables, to compute the correlation coefficients between the two data sets. Compared to the ordinary correlation, the canonical correlation analysis (CCA) is independent from the used coordinate system. Therefore, it provides the maximum correlation of the variables, which is also called the canonical correlation. The algorithm and implementation is based on the tutorial of (Borga, 2001). Equation (6) shows the corresponding maximization problem. The correlation coefficient ρ should be maximized, with respect to$\widehat{\mathrm{w}}_{\mathrm{x}}$ and $\widehat{\mathrm{w}}_{y}$. The two vectors $\widehat{\mathrm{w}}_{x}$ and $\widehat{\mathrm{w}}_{y}$ are the normalized base vectors for canonical correlation. *X* and *Y* are the analysed multidimensional variables.

$$\max_{\widehat{\mathrm{w}}_x\widehat{\mathrm{w}}_y} \rho = \max_{\widehat{\mathrm{w}}_x\widehat{\mathrm{w}}_y}\left(\frac{E[{\widehat{\mathrm{w}}_x}^T X Y^T \widehat{\mathrm{w}}_y]}{E[{\widehat{\mathrm{w}}_x}^T X X^T \widehat{\mathrm{w}}_x] E[{\widehat{\mathrm{w}}_y}^T Y Y^T \widehat{\mathrm{w}}_y]}\right) \tag{6}$$

The calculation of the maximum correlation is based on the eigenvalue equations (7) and (8). Thereby, C is the covariance matrix of the multidimensional variables. In these equations more non-zero eigenvalues may be possible. The eigenvalues represent the squared correlation coefficients. This is why the highest eigenvalue leads to the canonical correlation. The corresponding eigenvectors are the base vectors from (6).

$$C_{xx}^{-1} C_{xy} C_{yy}^{-1} C_{yx} \widehat{\mathrm{w}}_x = \rho^2 \widehat{\mathrm{w}}_x \tag{7}$$

$$C_{yy}^{-1} C_{yx} C_{xx}^{-1} C_{xy} \widehat{\mathrm{w}}_y = \rho^2 \widehat{\mathrm{w}}_y \tag{8}$$

Further details on the CCA can be found in (Borga, 2001). When used for the c-VEP based BCI the multidimensional variables X and Y can be replaced by the multichannel EEG signals and a set of templates. Therefore, the used features are not based on spectral analysis, but on the relationship between the derived EEG and a template sequence. For online processing the canonical base vectors act as spatial filters, like they are used in (Bin et al., 2011). An online and an offline version of the CCA were implemented. An online Simulink block loads the template sequences, which are extracted from a training session and the correlation maximizing weight vectors (through spatial filtering). The corresponding S-function computes the correlation coefficient between the weighted template and the weighted EEG online. A MATLAB function, allows offline calculation of the canonical correlation coefficients, which are used as features signals for the calculation of the classifier.

2.8.4. Classification and Pseudo Zero Class

For target identification an LDA based classifier is used. The LDA is a method that is used for discrimination and classification based on feature signals. During a training session a linear discriminator is calculated, based on data, which can be matched to a previously defined class. This discrimination is based on feature vectors containing all necessary attributes of the individual classes. The resultant classifier is used online to predict the actual class based on the corresponding features As all configurations in this work use four targets, an LDA implementation is used that compares one class with the rest and repeats the comparison for each class. Each comparison provides score with respect to the discrimination. The final classification result is the class providing the highest score (Bishop, 1995).

A pseudo zero class provides an idle state, which is used, when no target is selected. Based on the classification scores only, it is not possible to detect, if the user has selected any target. This is achieved by rejecting any classification result, for which the residual error probability is larger than a predefined limit. (9) shows a Softmax function, which transforms the output of the discrimination function into a value *p*, which lies between 0 and 1. In the equation, q_i is the distance to class i and τ is the so called temperature, which is used to adjust the gap between the resultant probabilities (Sutton and Barto, 1998). *N* is the total number of possible classes.

$$p_i = \frac{e^{q_i/\tau}}{\sum_{b=1}^{N} e^{q_b/\tau}} \tag{9}$$

2.9. Online BCI model

2.9.1. f-VEP

In the configuration used in this work, BCIs based on f-VEP require a training session, before the system can be used online. Therefore, the user has to focus on the indicated targets. The extracted features are saved together with an indicator flag, which is used for trial identification. To calculate a valid LDA classifier, the user has to perform a training run. Figure 19 shows a schematic flow diagram. The LDA classifier is calculated offline based on the trained features. Offline calculation is performed with the g.BSanalyze functions. Then, this classifier is used online in the Simulink BCI-model.

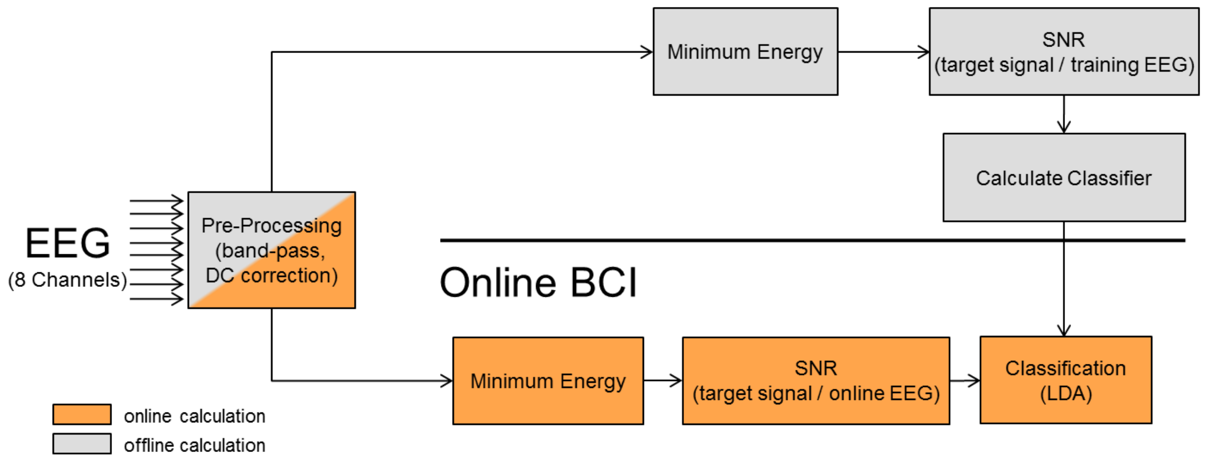

Figure 19: Flow diagram of the f-VEP based BCIs. The ME algorithm is used to improve the SNR between the target signal and the base EEG, which is then used as a feature for classification. The offline calculated LDA classifier is used for online target identification.

The Simulink model for the BCI using LED stimulation can be found in Figure 20. The *g.USBamp* block provides the EEG data with a sampling rate of 256 Hz for online processing. Within the *Minimum Energy* block several parameters can be configured like the EEG buffer length and the re-estimation time of the features. For the presented experiments the EEG buffer is set to 1, 2 and 3 s. The feature re-estimation interval is 200 ms for all configurations. A temporal median filter of 2 s is used to smooth the feature signals. Together with the EEG buffer length, the median filter influences the reaction time of the BCI. The inter-feature median is subtracted from each feature channel to improve the discrimination of the features. The *Apply Classifier* block uses the offline calculated classifier to identify the selected target. If the classification result does not lie within a 97 % confidence interval, it will be rejected and the output is assigned to the pseudo zero class.

An e-puck robot is connected through the *epuck Control* block and receives a translated command that is based on the classification result. The *Paradigm* block is used to select the operation mode, where training, feedback or free run can be used. For the visual stimulation based on LEDs, the *g.STIMbox* block controls the flash cycles of the light sources. For every session, the corresponding data is stored in a .mat file. The file contains the EEG data, the flash indicator flag and the classification result, which are needed for offline analysis after the training run. All blocks, used in the model of the f-VEP LED configuration were provided by g.tec and was already used in the work of (Prueckl and Guger, 2010). Therefore, this model is the reference model to test the performance of the on-screen stimulation.

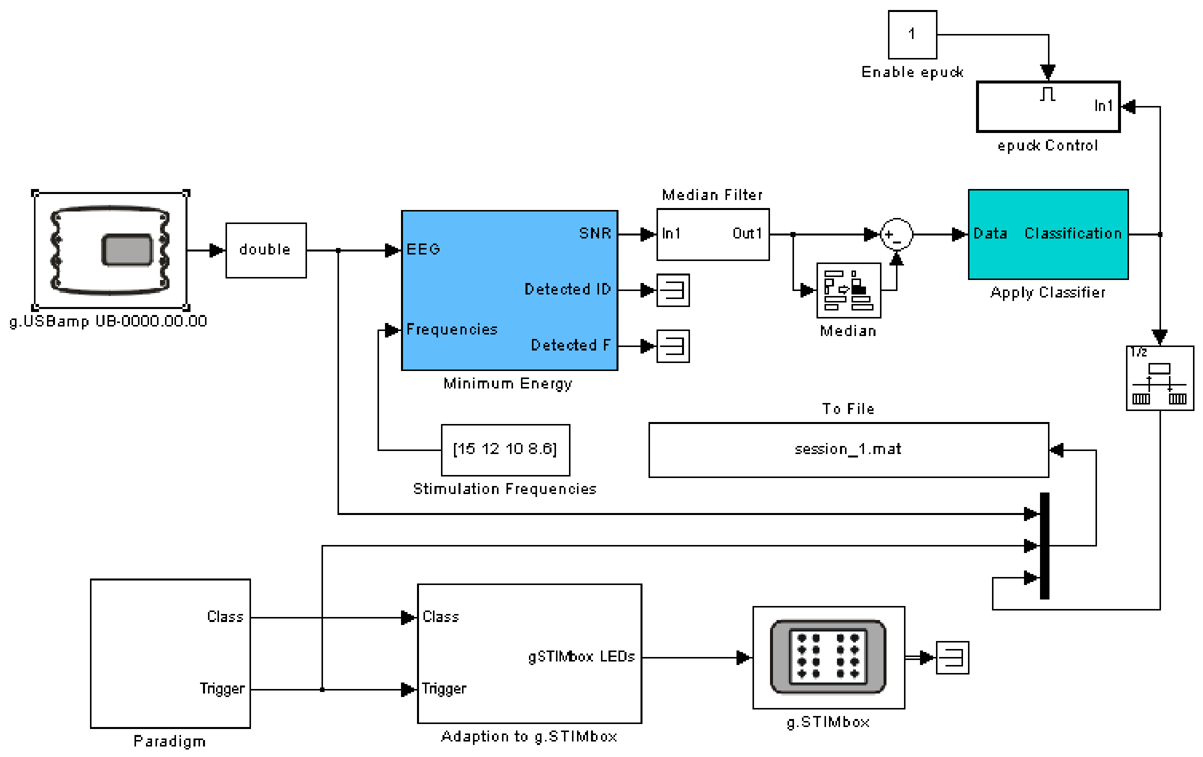

Figure 20: Simulink model of the f-VEP based BCI with LED stimulation. The *g.USBamp* block provides the acquired EEG data. The *Minimum Energy* block computes the features every 200 ms. Inter-feature and temporal median filters are used to smooth the feature signals. Classification is performed within the *Apply Classifier* block. The *Paradigm* block can switch between training, feedback and free run mode. Visual stimulation is controlled by the *g.STIMbox* block. The classification result is sent to the robot via the *epuck Control* block.

Figure 21 shows the model for f-VEP system using the BCI-Overlay for on-screen stimulation. The parameters are similar for f-VEP LED and the f-VEP on-screen models, except of the adapted Paradigm and the Interface Unit block. The *Interface Unit* loads the specification of the BCI controls. During simulation the Paradigm block manages the UDP connection to the BCI-Overlay. Again, the *Paradigm* block can switch between training, feedback and free run mode. The Interface Unit was provided by g.tec. The *Paradigm* block had to be extended, to control the BCI-Overlay. All communication between the BCI-model and the BCI-Overlay is performed within a C based S-function that is used by the *Paradigm* block.

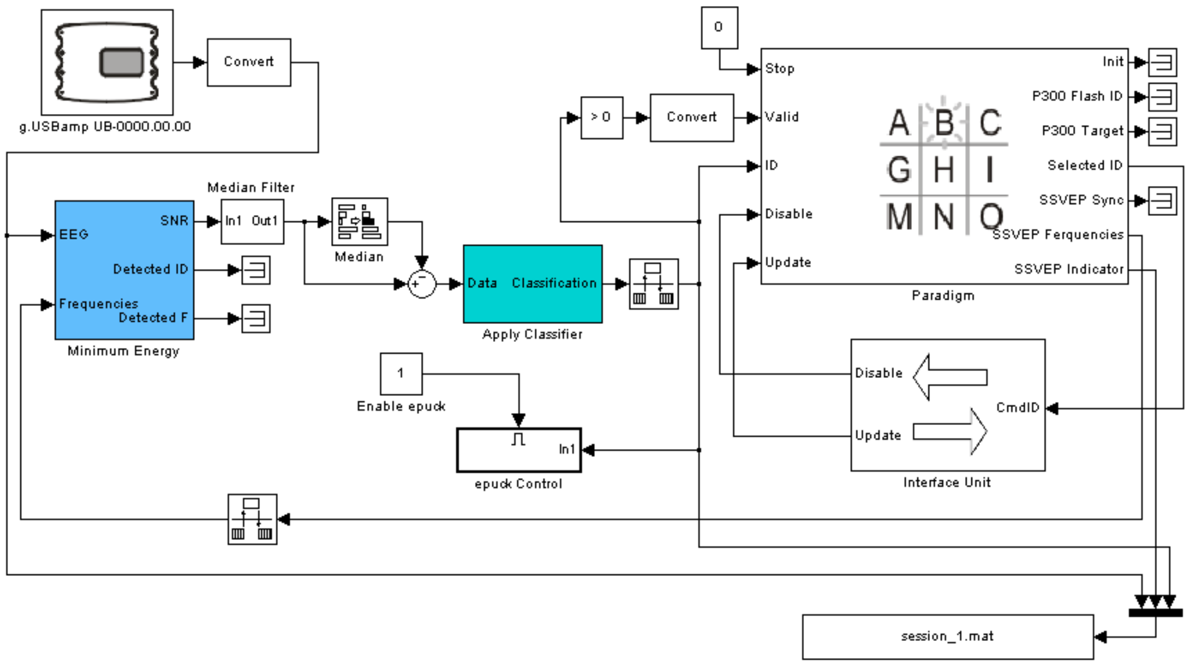

Figure 21: Simulink model of the f-VEP based BCI with on-screen stimulation. The *g.USBamp* block provides the acquired EEG data. The *Minimum Energy* block computes the features every 200 ms. Inter-feature and temporal median filters are used to smooth the feature signals. Classification is performed within the *Apply Classifier* block. The *Paradigm* block can switch between training, feedback and free run mode. In contrast to the LED based stimulation, the *Paradigm* block directly controls the BCI-Overlay. The BCI controls are defined through the *Interface Unit*. The classification result is sent to the robot via the *epuck Control* block.

2.9.2. c-VEP

Like the previously defined systems, the c-VEP based BCI requires a training session. In contrast, this BCI configuration uses only one reference target for training that is stimulating with an M-sequence during the whole training run. These stimulation cycles are averaged during offline analysis, to calculate a template of at least two cycles length. The resulting template is 1.05 s times the number of repetitions, caused by the 63 bit M-sequence (section 2.6). Phase shifted indicator flags are applied to the trained data, to extract the feature signals for the classifier. As well as for the f-VEP configurations, an LDA is used to calculate the classifier. The CCA algorithm is used to calculate the spatial filter for the EEG data and the template data (see section 2.8.3). The features are the correlation coefficients between the combined EEG data and the templates. For each target there exists an individual set of templates, which is a phase shifted version of the reference template set. Figure 22 depicts the flow diagram of the c-VEP based BCI. After the training session, the classifier is used for online classification.

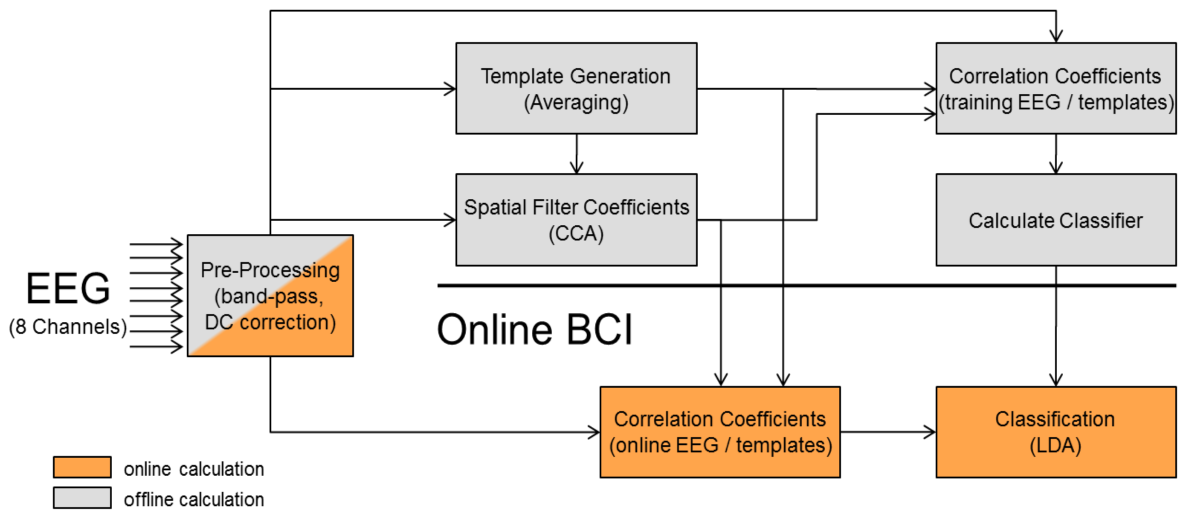
Template Generation (Averaging)
Correlation Coefficients (training EEG / templates)
Spatial Filter Coefficients (CCA)
Calculate Classifier
EEG (8 Channels)
Pre-Processing (band-pass, DC correction)
Online BCI
Correlation Coefficients (online EEG / templates)
Classification (LDA)
online calculation
offline calculation

Figure 22: Flow diagram of the c-VEP based BCI. A CCA algorithm is used to calculate a spatial filter that maximizes the correlation between the trained data and the averaged template. The resulting correlation coefficients are used for LDA based classification and identification of the selected target. The offline calculated templates and the spatial filters are used for feature extraction during online processing.

The corresponding implementation in Simulink is shown in Figure 23. The *g.USBamp* provides data with a sampling rate of 256 Hz. Feature extraction is performed in the *CCA* block every 200 ms. The EEG buffer length has to be multiple of the template length, where the minimum length is the duration of one M-sequence cycle (here: 1.05 s). Also 2.10 s and 3.15 s buffers are possible. Longer buffer lengths are possible, but not be considered in this work, because of the decreasing reaction time of the BCI. The rest of the model works similar to the f-VEP based BCI with on-screen stimulation, including the confidence interval for classification, which is set to 3 %.

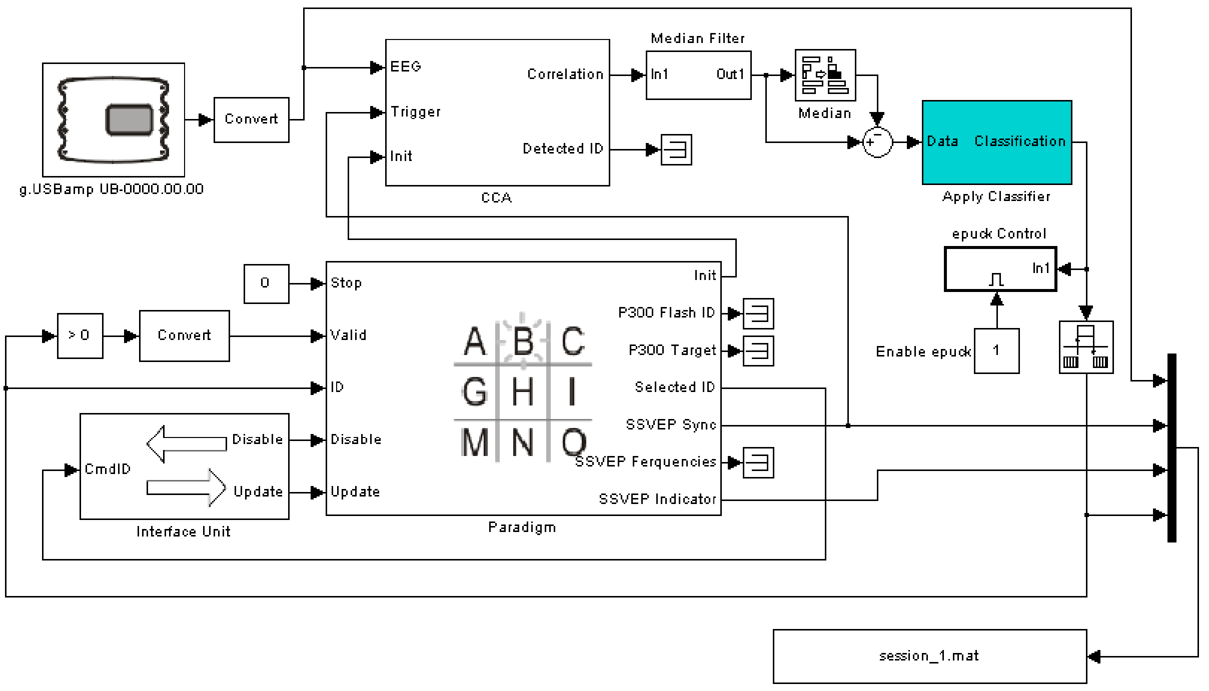

Figure 23: Simulink model of the c-VEP based BCI with on-screen stimulation. The *g.USBamp* block provides the acquired EEG data. The *CCA* block computes the features every 200 ms. Inter-feature and temporal median filters are used to smooth the feature signals. Classification is performed within the *Apply Classifier* block. The *Paradigm* block can switch between training, feedback and free run mode. Also the *Paradigm* block controls the BCI-Overlay. The BCI controls are defined through the *Interface Unit*. The classification result is sent to the robot using the *epuck Control* block.

2.10. Experimental Procedure

Eleven subjects aged 27.36 +/-5.84 years participated in all experiments (ten male and one female). All subjects were in good health, with normal or corrected to normal vision, who volunteered to participate. Each subject participated in one session that lasted about two hours. The whole session contained three experiments with each subject, one for each configuration: f-VEP LED, f-VEP on-screen and c-VEP on-screen. The order of these conditions was counterbalanced across subjects to avoid any possible training or fatigue effects.

Each subject first performed a BCI training run, depending on the used configuration, to set up a subject specific classifier. In the experiments using the f-VEP LED and the f-VEP on-screen configuration, the training contained 20 trials per class (80 trials in total). A single trial was formed by 3s dark time and 7s flash time. During the flash time all light sources flickered at their predefined frequency. A green indicator showed the user the current target. This cue started together with the visual stimulation. The f-VEP training is split into four individual runs, to avoid tiring effects.

The training of the c-VEP configuration contained one run with a single reference target. Therefore, the duration of the training session depended on the sequence length and the number of cycles, but not on the number of targets. The training contained 200 stimulation cycles of the selected M-sequence. From the 200 trained sequences, 190 trials were available to calculate the classifier, which leads to 760 trials in total for four classes, as each set of trials was phase shifted version of the others. As there was only one flickering target during c-VEP training, no cue was necessary do direct the gaze of the user.

After the training run each subject performed a feedback run containing 5 trials per class (20 classification in total), to test the performance of the classifier. The feedback run was equal for each configuration (c-VEP on-screen, f-VEP on-screen and f-VEP LED). During the feedback run the system presented four targets to the user either stimulating with constant frequencies or with pseudo-random sequences. A cue showed the user the current target s/he had to gaze at. The cue was realized either as a green LED for the f-VEP LED configuration or as a green border around the on-screen target in the c-VEP on-screen and the f-VEP on-screen configuration. One trial consisted of 3 s dark time and 7 s flash time. The cue and the stimulation targets were active during flash time only.

Next, the subject had to steer the robot along a given track as fast as possible. In this run no cue was used and the user had the possibility to choose between four commands (forward, backward, left and right). Depending on the configuration, either a green LED or a green border around the target showed the user the current active selection of the BCI. This gives the user additional feedback, if the system is in idle state or not. The entire track was 170 cm long and contained four 90° turns - two to the left and two to the right (Figure 24). Each subject was told to move as accurate as possible along the track. The pseudo zero class was enabled during the experiment. Therefore, the robot stopped when no control was detected, which was the case for bad feature signals or when the user did not focus on any target. This led to a delay decreasing the performance of the system.

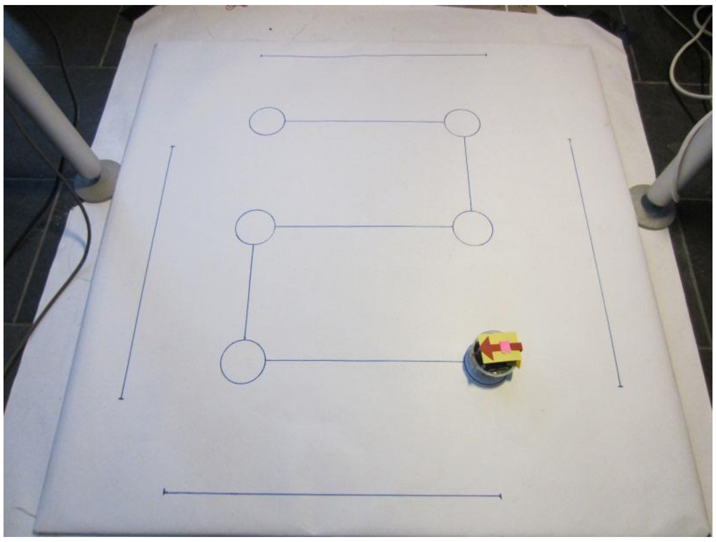

Figure 24: Setup of the track with the e-puck robot.

During all experiments, the used hardware setup for the VideoClient and the BCI-Overlay consisted of a PC with an AMD Phenom II X4 955 (4 x 3.2 GHz) processor, an Nvidia GeForce 8600GT graphics card and 4 GB RAM. The operation system was an Ubuntu using a LINUX 2.6.33-29-realtime kernel.

2.11. Quade-Test

The Quade-test is a non-parametric statistical test and an extension of the Wilcoxon-signed rank test for two or more groups. The null-hypothesis for this test is that the groups have the same median (Quade, 1979). In this work the test is used to compare the classification accuracies and the time of movement of the three configurations.

3. **Results**

3.1. Technical Performance Tests

This section describes some technical experiments, which ensure a valid setup of the tested systems. Critical aspects are the flash accuracy of the display for on-screen stimulation, the trigger accuracy for synchronizing the BCI-Overlay with the c-VEP BCI model and the tracking accuracy of the robotic movement within the test environment.

3.1.1. Trigger Accuracy Test

Two test runs were performed to evaluate the delay caused by the transmission of UDP commands. In the first test run a package with one byte was sent 10000 times. The initial sender was based on MATLAB and used the g.UDPinterface to send and receive commands. After receiving a command, the BCI-Overlay sent back the same command, so that the round-trip-time (RTT) could be measured. A second test was similar to the first one, but with bigger packages containing 72 bytes (Table 4).

Table 4: The trigger accuracy test contained two runs. The round trip time was measured by the MATLAB test program.

Parameter	Test run 1	Test run 2
Number of packages	10000	10000
Data size (byte)	1	72
Mean Round-Trip-Time (RTT) (ms)	0.367	0.414
Standard deviation of RTT (ms)	+/- 0.147	+/- 0.138
Min RTT (ms)	0.089	0.243
Max RTT (ms)	1.006	2.074

The maximum delay observed in both tests was about 2 ms. It is important to keep in mind that the RTT measures double the time of a trigger command from the BCI-Overlay. This is why the maximum delay observed for a trigger command is about 1ms. Figure 25 shows that the outliers were very rare, compared to the amount of packages sent. The mean RTT of a package with 72 bytes reaches about 400 µs. The test conditions were similar to the performed experiments later.

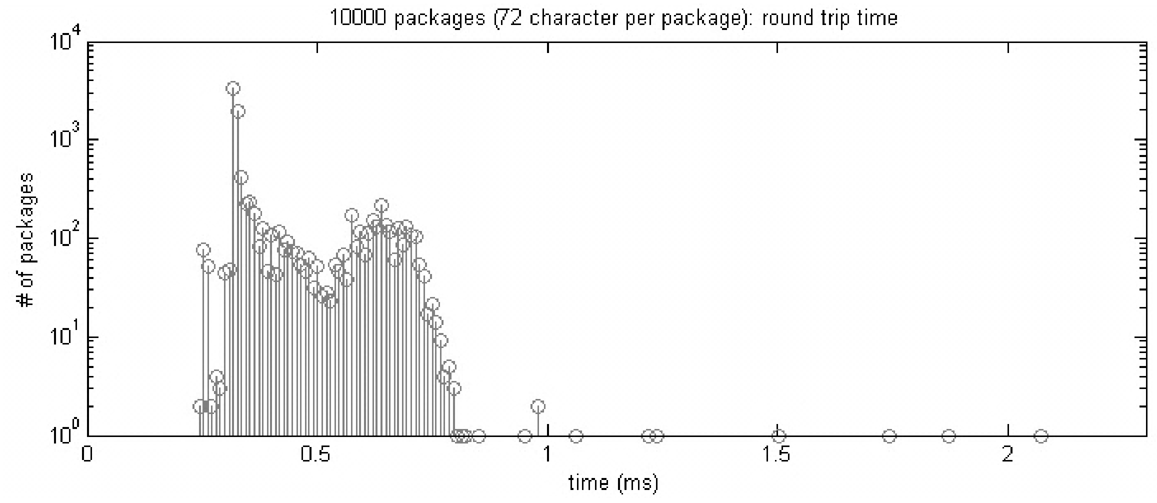

Figure 25: Round-Trip-Time of the 72 bytes packages. The figure is plotted with semi logarithmic scale. Most of the packages have RTTs less than 500 µs.

3.1.2. Flash Accuracy Test

Measurements of 10 minutes were used to evaluate the flash accuracy of the on-screen stimulation. The measurement setup included a photo detector connected to the g.TRIGbox. For signal acquisition a g.USBamp with a sampling rate of 38.4 kHz and a corresponding Simulink model was used. Target frequency was 15 Hz, resulting in more than 8000 measured periods. The host application, which included the BCI-Overlay, was the Video-Client (further details see 2.3.5). Table 5 summarizes the results of this test. The maximum error was 33.33 ms, which resulted from a delayed call of swap-buffer by the host application. 99.7 % of the flashes occurred within the expected number of frames. The signal jitters with a standard deviation of +/- 1.18 ms around the given period.

Table 5: Flash accuracy of more than 8000 stimulation cycles.

Parameter	**Frequency**	**Time**
	Hz	ms
Ideal	15.00	66.67
Measured mean	14.99	66.72
Standard deviation	+/- 0.21	+/- 1.18
Max error	5.00	33.33
Mean error	-0.01	0.05

Figure 26 shows the acquired durations. The effective jitter, excluding outliers, is less than the overall standard deviation. The standard deviation of the regular signal is +/- 84 µs, which is at least two orders of magnitude smaller than the sampling rate of the amplifier (256 Hz) and the refresh rate of the screen (60 Hz).

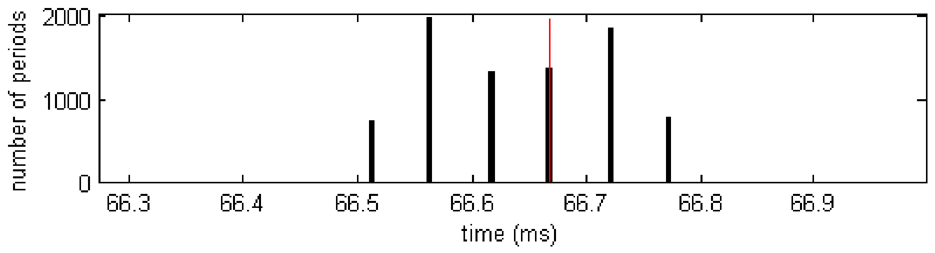

Figure 26: Display jitter of the Belinea 1930 S2 monitor and the BCI-Overlay running within the Video-Client. The red bar shows the ideal value of 66.67ms, due to the 15Hz stimulation of the target.

The error generated by an outlier is in any case a multiple of the screen refresh cycle, as it is caused by a delayed swapping of OpenGL frame-buffers. One reason for these artifacts may by background processes, which are executed by the operating system of random time points (Figure 27).

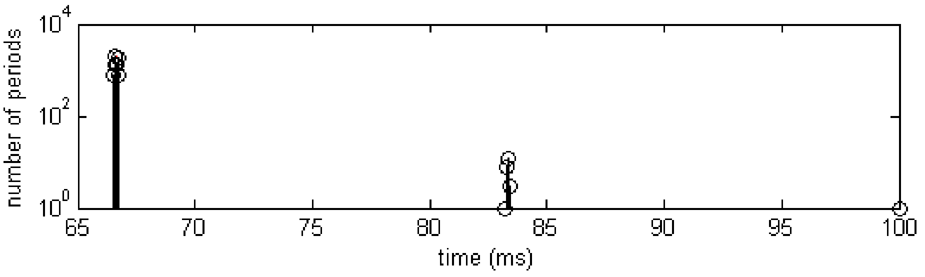

Figure 27: Outliers during visual stimulation are caused by a delayed call of the swap-buffer command. The errors are always multiples of 16.67 ms, because of the 60 Hz refresh rate of the display and the vertical synchronization of the graphics card. The plot shows a semi-logarithmic scale for a better view on the rate of the outliers.

3.1.3. Tracking Accuracy Test

A marker with the dimensions of 1 cm x 1 cm was placed on the robot. This size of the marker was necessary to ensure detection by the tracking system. The marker position was measured over 2 min without any movement of the robot. In ideal case, the tracking system detects the same point every time. However, the tracking system underlies inaccuracies during measurement. Figure 28 shows the distribution of the acquired data points. The median in x-direction and in y-direction was used, to estimate the zero point.

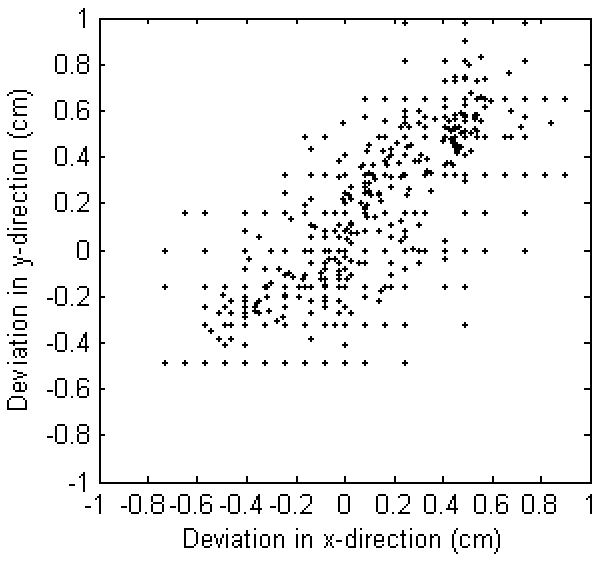

Figure 28: Distribution of 2440 measured positions of a steady marker with a size of 1 cm x 1 cm.

The histogram in Figure 29 shows the distribution of the data points around the estimated zero point. In total 2440 data points were acquired. The maximum Euclidian distance between the estimated zero point and the data points was 1.23 cm. The mean error distance was 0.45 cm, with a standard deviation of +/- 0.25 cm.

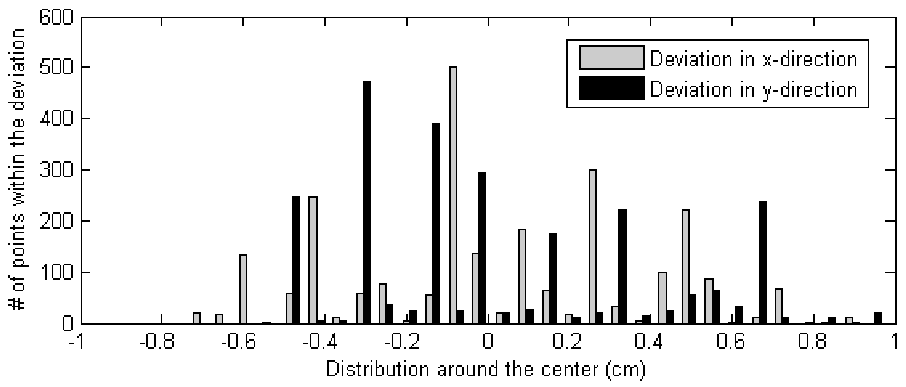

Figure 29: Histogram of the data points in x-direction and in y-direction. The data points are spread around the estimated zero point.

3.2. Online Classification Accuracy Test

3.2.1. Classification Result without Pseudo Zero Class

An overview of the classfication accuracies for the individual configurations is provided in Table 6, Table 7 and Table 8. Table 6 contains the classification accuracy for the f-VEP LED configuration. The f-VEP on-screen configuration is shown in Table 7 and the c-VEP on-screen configuration can be seen inTable 8. The mean accuracy is calculated using the last 3 s of data in all trials. Three different buffers were used for classification, to show the dependence of the classification accuracy on the EEG buffer size. An online classification test was performed in combination with the 2.0 s EEG buffer for f-VEP systems and the 2.1 s EEG buffer for the c-VEP system. All other results were calculated offline on base of the recorded EEG data. In all tests the c-VEP system provided better results on average over all subjects.

A Quade-test showed significant better accuracy for the c-VEP on-screen configuration compared to the others (p = 0.0168). An ANOVA was not possible in this case, because the test data does not come from normal distribution.

Table 6: Classification accuracy of the f-VEP LED configuration after 20 trials in total. Three EEG buffer sizes were used: 1.0 s, 2.0 s and 3.0 s. The 2.0 s configuration was tested online; all other accuracies were calculated offline on base of the EEG data, recorded during the online experiment.

Subject	**f-VEP LED**		
#	mean accuracy (1.0 s buffer) (%)	mean accuracy (2.0 s buffer) (%)	mean accuracy (3.0 s buffer) (%)
1	100.00	98.65	100.00
2	58.37	75.72	80.57
3	88.09	88.44	93.09
4	78.01	85.57	95.70
5	98.08	100.00	99.99
6	94.42	96.63	96.58
7	43.03	40.76	38.56
8	73.71	72.45	70.57
9	74.16	84.62	91.46
10	65.63	77.19	78.95
11	100.00	100.00	100.00
mean	**79.41**	**83.64**	**85.95**
std-dev	17.88	16.58	17.69

Table 7: Classification accuracy of the f-VEP on-screen configuration after 20 trials in total. Three EEG buffer sizes were used: 1.0 s, 2.0 s and 3.0 s. The 2.0 s configuration was tested online; all other accuracies were calculated offline on base of the EEG data, recorded during the online experiment.

Subject	**f-VEP on-screen**		
#	mean accuracy (1.0 s buffer) (%)	mean accuracy (2.0 s buffer) (%)	mean accuracy (3.0 s buffer) (%)
1	99.83	98.14	98.50
2	72.99	77.14	84.35
3	76.13	83.05	78.83
4	93.62	97.26	97.93
5	99.66	100.00	99.52
6	93.47	95.09	91.37
7	46.85	42.48	45.32
8	57.85	80.77	86.60
9	72.25	89.28	89.97
10	57.98	62.75	75.36
11	100.00	100.00	100.00
mean	**79.15**	**84.18**	**86.16**
std-dev	18.43	17.25	15.21

Table 8: Classification accuracy of the c-VEP on-screen configuration after 20 trials in total. Three EEG buffer sizes were used: 1.05 s, 2.1 s and 3.15 s. The 2.0 s configuration was tested online; all other accuracies were calculated offline on base of the EEG data, recorded during the online experiment.

Subject	**c-VEP on-screen**		
#	mean accuracy (1.05 s buffer) (%)	mean accuracy (2.1 s buffer) (%)	mean accuracy (3.15 s buffer) (%)
1	96.29	98.29	100.00
2	87.95	90.39	94.71
3	91.96	95.85	94.83
4	94.70	93.42	91.55
5	94.06	95.83	98.16
6	94.85	97.97	95.11
7	72.33	80.49	81.20
8	89.77	95.00	96.35
9	98.40	96.88	94.18
10	94.95	96.84	99.68
11	96.74	98.66	97.42
mean	**92.00**	**94.51**	**94.84**
std-dev	6.87	4.98	4.93

Another comparison of the three configurations is provided in Figure 30. Here, the accuracy of every system is plotted against the duration of one trial. Each curve contains the average accuracy over eleven subjects and 20 trials. The accuracy values were calculated every 200 ms. Apart from the small differences in the EEG buffer sizes (c-VEP: 2.1 s; f-VEP: 2.0 s), the data was smoothed with moving median window filter (c-VEP: 1 s, f-VEP: 2 s). Nevertheless, the c-VEP configuration reaches a higher accuracy level in shorter time, compared to the f-VEP based BCIs.

The dependence of the systems reaction time on the EEG buffer length is demonstrated in Figure 31. In this example only the c-VEP configuration is shown. Smaller buffer sizes improve the reaction time, while the loss in accuracy is very small. Table 8 shows a decrease of less than 3 % of the mean accuracy, when an EEG buffer of 1.05 s is used instead of 3.15 s. Compared to this, the accuracy the f-VEP systems decreases about 7 %, caused by the change from 3 s to 1 s (Table 6 and Table 7).

In Figure 30 and Figure 31, no visual stimulation takes place prior to the vertical bar. The expected accuracy would be 25 %, which is the probability of random classification, within a four-class system and a very large number of trials. However, this is not the case in the presented results. First, the small number of trials enlarges the interval for random classification (Müller-Putz et al., 2008a). Second, in the example shown, the EEG buffer contains data from the previous run. Therefore, the classification result is more influenced by the target signal of the previous and different trial, than by random classification.

Based on the accuracy curve in Figure 30, the trial duration for a valid classification can be assumed as 2.8 s for the c-VEP system and as 3.7 s for the f-VEP configurations. These time points represent the start of constant classification accuracy over time. For ITR calculation the same settings as in the experiments for robot movement are used. This leads to an ITR of 34.4 bits/min for the c-VEP based BCI, with respect to the 4 targets and the 94.51 % mean accuracy for the 2.1 s EEG buffer. The f-VEP on-screen solution has an ITR of 18.0 bits/min, based on the 2 s EEG buffer and the 84.18 % mean accuracy. The ITR for f-VEP with LED stimulation is 17.8 bits/min. Here, the settings were the same as for the f-VEP system with on-screen stimulation.

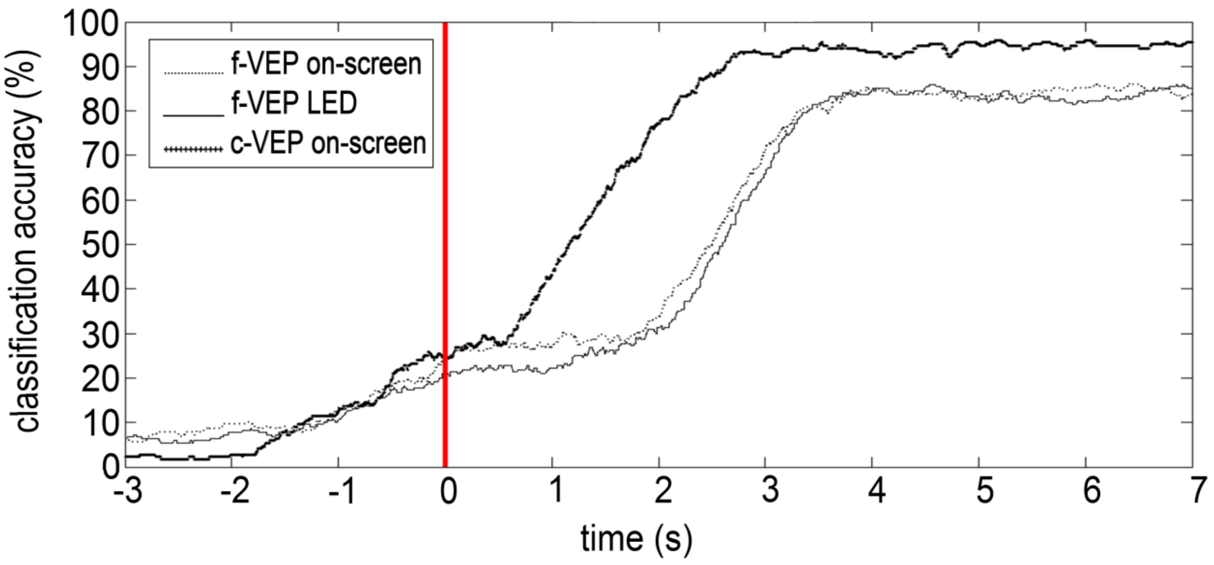

Figure 30: Online accuracy test for the three configurations: c-VEP on-screen, f-VEP on-screen, f-VEP LED. The vertical bar indicates the start of visual stimulation. The EEG buffer was 2.1 s for the c-VEP system and 2 s for the f-VEP systems. A moving median filter of 1 s for c-VEP and 2 s for f-VEP was used to smooth the data.

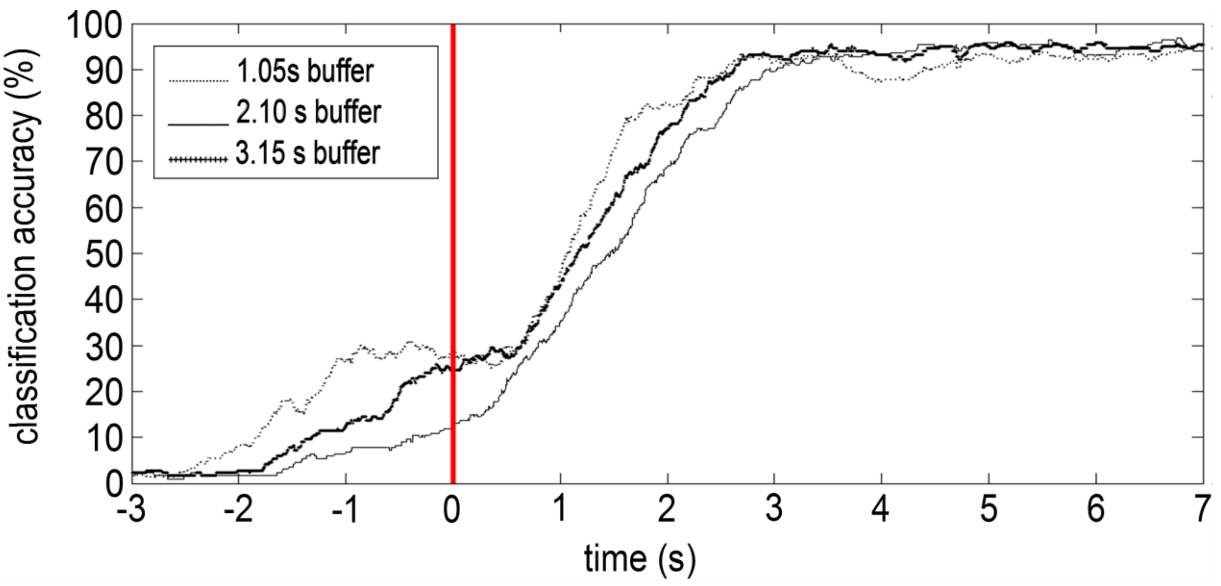

Figure 31: Comparison of the c-VEP accuracies achieved for 1.05 s, 2.10 s and 3.15 s EEG buffers. The vertical bar indicates the start of the visual stimulation.

3.2.2. Classification Result with Pseudo Zero Class

On base of the EEG data acquired during the classification accuracy test (section 3.2.1), another classification accuracy test was performed introducing an additional pseudo zero class. Table 9 provides the classification accuracies for the individual subjects after 20 trials in total. In this experiment a 2 s EEG buffer was used for f-VEP systems and a 2.1 s EEG buffer for the c-VEP system. The mean accuracy was calculated over all trials, using the last 3 s of each trial.

Table 9: Classification accuracy with pseudo zero class over 20 trials in total. The performance of all three configurations was evaluated: f-VEP LED, f-VEP on-screen and c-VEP on-screen.

Subject	**f-VEP LED**	**f-VEP on-screen**	**c-VEP on-screen**
#	mean accuracy (2.0 s buffer) (%)	mean accuracy (2.0 s buffer) (%)	mean accuracy (2.1 s buffer) (%)
1	97.63	93.46	88.63
2	8.28	34.24	64.45
3	57.02	38.33	83.24
4	31.93	77.88	86.32
5	90.48	86.73	80.98
6	90.12	81.41	91.21
7	0.00	0.68	20.47
8	39.71	8.43	65.64
9	53.18	47.75	92.64
10	40.19	18.96	53.50
11	100.00	100.00	91.34
mean	**55.32**	**53.44**	**74.40**
std-dev	33.76	34.23	21.08

Again, the c-VEP configuration provided the highest accuracies. The f-VEP configurations showed quite similar results compared to each other. A Quade-test was performed and showed significantly higher accuracy than the others ($p = 0.0055$).

The reaction time of the c-VEP system is slower compared to the setup without the pseudo zero class, it is still shorter than for f-VEP setups (Figure 32). The benefit of the pseudo zero class is that the BCI remains in idle state, when the user has not selected any target.

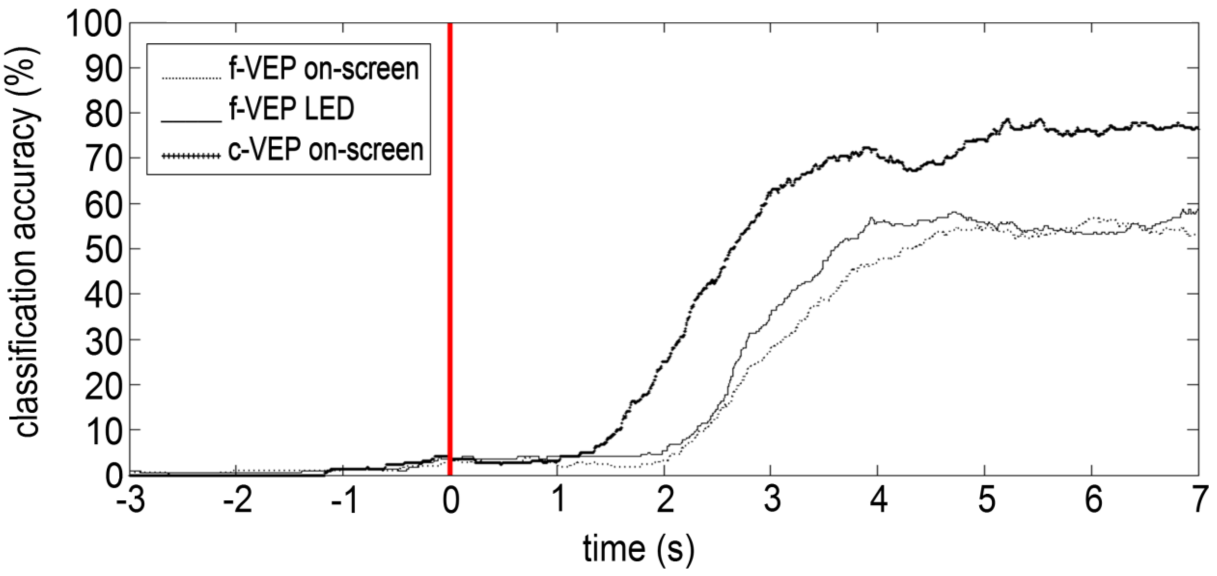

Figure 32: Online accuracy test for the three configurations: c-VEP on-screen, f-VEP on-screen, f-VEP LED. The vertical bar indicates the start of the visual stimulation. The EEG buffer was 2.1 s for the c-VEP system and 2 s for the f-VEP systems. Moving median filter was set to 1 s for c-VEP and 2 s for f-VEP.

Figure 33 shows the average false positive classifications of all sample points in one trial over all subjects. In the first 3 s, no stimulation takes place. Therefore, the positive classifications are either resulting from delayed classification of samples from the previous trial or by insufficient rejection of the classification result.

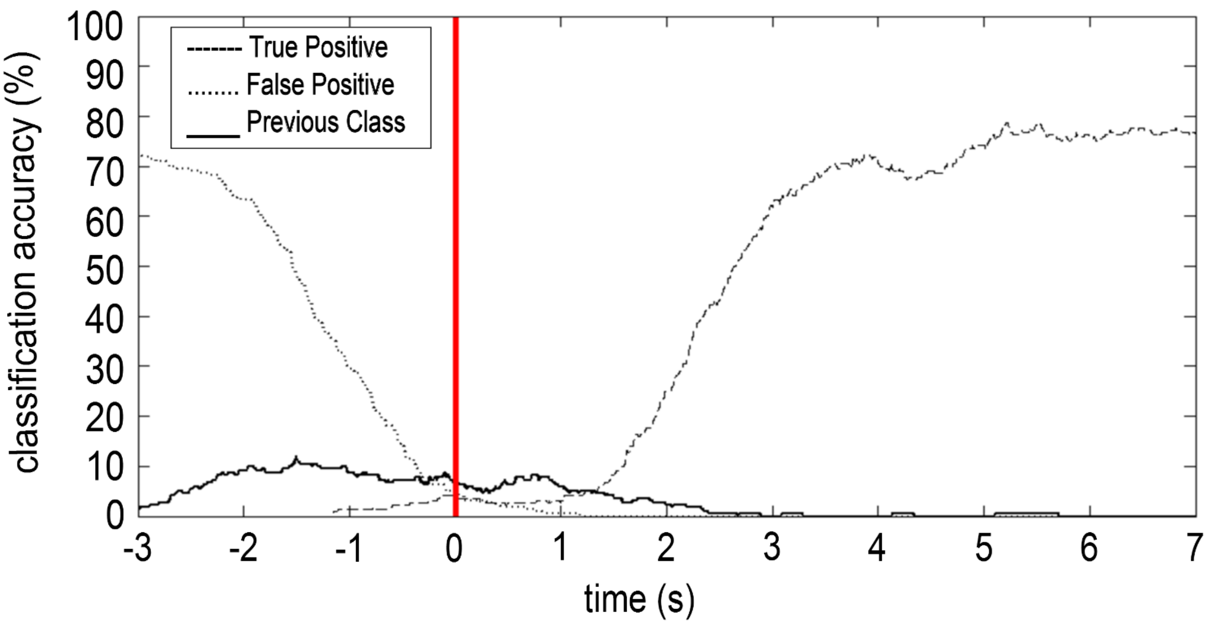

Figure 33: Classification accuracy and false positive classifications. In this test c-VEP on-screen system was used with a 2.1 s EEG buffer. The doted curve at the start of the trial represents the classifications caused by the remaining samples of the previous trial. The solid curve indicates the false positive classifications, when the system is idle and during stimulation of any other class. The vertical bar indicates the start of the visual stimulation.

This leads to false positive assignments of a sample to any class that is not the target class. In a system without pseudo zero class, the false positive rate would be 100 %. The pseudo zero class provides a decrease of false positive classifications down to 11.82 %, which is the maximum number of false positive classifications during the ten seconds of the trials.

For ITR calculation of the BCIs using the pseudo zero class, the new trial times are 3.4 s for the c-VEP system and 3.8 s for f-VEP with LED stimulation and 4.2 s for f-VEP with on-screen stimulation (Figure 32). The calculation is based on the mean accuracies of the 2 s and 2.1 s test runs. Because of the introduced pseudo zero class, the number of classes has increased to five. Therefore, the ITR is 17.5 bits/min for the c-VEP based BCI, 6.9 bits/min for the f-VEP based BCI with LED stimulation and 5.6 bits/min for the f-VEP based BCI with on-screen stimulation.

3.3. Controlling a Robot

The estimated maximum error of the Noldus tracking system was 1.23 cm. The data was filtered with a 15 samples median filter to eliminate outliers and the minimum step size between two points was set to 1.5 cm. The median filter length is based on the 25 Hz sampling rate of the tracking system and the 2.5 cm/s maximum speed of the robot. Figure 34 shows an example for a detected track. The data was taken from subject 8 including runs with four different input types: Keyboard, c-VEP on-screen, f-VEP on-screen and f-VEP LED.

Not all subjects were able to handle the task as well as the subject in Figure 34. Subjects with deviations of more than +/- 2*σ (standard deviation), either from the ideal track or from the path length, were excluded from the test. The intention of the outlier correction was avoiding comparison of inadequate data, caused by shortening the track. A list of the deviations that were achieved by the subjects can be seen in Table 10. The subjects 3, 6 and 11 had to be excluded due to their inaccuracy. As subject 7 was not able to move the robot along the track using any of the f-VEP configurations, it had to be excluded too.

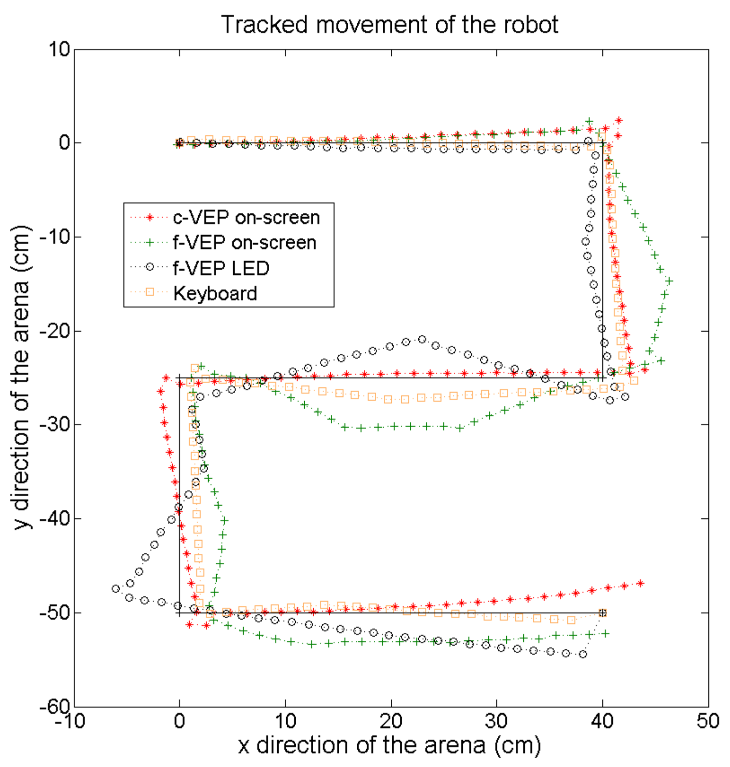

Figure 34: Tracked movement of the robot. Data was taken from subject 8. The individual curves show the way, the robot took for each of the four selected control paradigms.

Table 10: Deviation in length and the mean deviation from the ideal line of the track. The length of the track was 170 cm. Four subjects had to be excluded from the experiment because of their high deviations. The outliers are highlighted with grey background.

Subject	Keyboard		f-VEP LED		f-VEP on-screen		c-VEP on-screen	
#	deviation in length cm	mean deviation from track cm	deviation in length cm	mean deviation from track cm	deviation in length cm	mean deviation from track cm	deviation in length cm	mean deviation from track cm
1	3.70	0.81	18.60	1.47	21.60	1.33	15.40	1.29
2	0.60	0.70	-9.30	1.21	27.30	1.78	-12.60	1.82
3	6.00	0.99	32.60	2.93	-11.70	2.93	15.30	2.15
4	-6.40	1.16	19.80	2.18	9.90	1.65	18.20	1.16
5	6.00	1.22	-4.60	0.97	35.00	1.15	10.40	1.50
6	0.70	1.30	18.10	3.07	26.50	2.57	39.10	3.29
7	-4.40	1.17	-	-	-	-	-4.80	1.29
8	3.50	0.84	12.90	1.38	9.60	2.25	15.90	0.71
9	-0.40	0.55	32.40	2.60	49.70	3.51	24.90	1.15
10	1.00	0.72	17.40	1.28	24.70	1.25	1.60	1.15
11	-1.20	1.49	65.10	1.68	28.70	1.45	-3.00	1.34
mean	**0.83**	**1.00**	**20.30**	**1.88**	**22.13**	**1.99**	**10.95**	**1.53**
std-dev	3.74	0.28	19.68	0.72	15.72	0.76	14.10	0.66

Table 11 shows the time, the subjects needed to move the robot from start to end of the track. For comparison only the durations achieved on valid data sets were used. A Quade-test showed that the c-VEP configuration is significantly faster than the f-VEP LED and f-VEP on-screen configurations (p = 0.0187).

The keyboard run was excluded from outlier detection and every comparative test. It is just an additional measurement and should give the reader a feeling about the achievable values.

Table 11: Duration of movement for every subject and configuration. Grey highlighted subjects were excluded from the tests. The mean values are calculated for all subjects.The corrected mean is based on data without outliers.

Subject	**Keyboard**	**f-VEP LED**	**f-VEP on-screen**	**c-VEP on-screen**
#	time	time	time	time
	s	s	s	s
1	93.00	176.00	170.00	149.00
2	92.00	451.00	187.00	163.00
3	97.00	351.00	426.00	194.00
4	91.00	469.00	252.00	272.00
5	100.00	335.00	312.00	233.00
6	96.00	243.00	183.00	209.00
7	99.00	-	-	507.00
8	91.00	422.00	1158.00	298.00
9	89.00	844.00	679.00	145.00
10	97.00	365.00	1256.00	298.00
11	91.00	198.00	150.00	177.00
mean	**94.18**	**385.40**	**477.30**	**240.45**
std-dev	3.56	180.65	395.42	99.71
corrected mean	**93.29**	**437.43**	**573.43**	**222.57**
corrected std-dev	3.57	189.29	431.36	64.26

4. **Discussion**

The chosen white stimulation with solid rectangles provided a good comparative configuration. This allowed an evaluation of the BCI-Overlay and its performance. Nevertheless, it is quite interesting to see the behavior of the individual configurations, when spatial stimulation patterns are used for stimulation. In further experiments, such a test could also compare c-VEP and f-VEP with more than four targets.

Also test runs with targets of individual shape are very interesting. In this case, the training for f-VEP systems may become time consuming, as each control has to be trained individually. Nevertheless, this problem could be solved with the introduction of alternative classification techniques or with decreasing the trial duration. The c-VEP system is trained with only one reference target. This may lead to a stronger dependence on the shape and size of the controls. Of course, the templates could be trained individually for each target, but this would take more time for training. In f-VEP systems the SNR between the target signal and the base EEG depends on size and shape. Nevertheless, a change of the training paradigm will not improve the feature signals.

In all experiments performed with the f-VEP based BCIs, the BCI-Overlay showed nearly similar results compared to LED stimulation. Therefore, the on-screen solution works and can be used for further experiments. The design of the module allows usage in real world video application, as it was used in the presented experiments, but also in virtual reality applications, where the user could interact with the virtual environment.

In the c-VEP based BCI, a cutoff frequency of 30 Hz was chosen that is also the highest stimulation frequency (caused by the 60 Hz refresh rate of the screen). Compared to spectral analyzes of the EEG, the CCA uses the whole spectrum for feature extraction, which includes also nuisance signals and artifacts. This is why the filter contained a low cutoff frequency.

Classification accuracy tests with the c-VEP system showed very good results, also with short EEG buffers. The used stimulation sequence took 1.05 s, which is the minimum EEG buffer size too. This is why faster systems require shorter sequences. The maximum number of controls depends on the sequence length, such that shorter sequences allow fewer controls than longer sequences.

The results for the classification accuracy are based on 5 trials per class only (20 trials in total), so that the results have to be interpreted very carefully. The validity of classification results with respect to the number of trials is shown in (Müller-Putz et al., 2008a). The lower the number of trials, the higher is the threshold for confident accuracies that are better than random classification. For example, results in a four class system with 10 trials per class are random until 40 % classification accuracy. Nevertheless, the presented results are in the range of 80 % - 100 %. Only subject 7 showed accuracies in the range of 40 % - 50 %, which may be caused by random classification.

Subject 7 was not able to use the f-VEP configurations. The usage of the c-VEP was possible, but the accuracy was bad compared to the other subjects. Therefore it was excluded from the evaluation of the robot experiment. Nevertheless, this subject may be a hint, that the feature extraction of the c-VEP system works with people, which are not able to use f-VEP systems.

The reaction time of the used monitor was 5 ms, which produces a delay and therefore a phase shift. Nevertheless, the flash accuracy test showed that the jitter of the visual stimulation was very low compared to the used sampling rates. So, the delay seems to be constant and the resulting phase shift is the same in all signals. This is why the reaction time had no influence on the performance of the system.

The latency of the f-VEP configurations was much higher than the latency of the c-VEP configuration. One reason for the higher latency is the temporal moving median filter that was used. The filter length was bigger for the f-VEP configuration (2 s) compared to the c-VEP configuration (1 s). This was necessary, because of the less robust features of the ME combination compared to the features based on the CCA.

Compared to the c-VEP based BCI presented by (Bin et al., 2011), the ITR is very low with 34.4 bits/min, which was reached without pseudo zero class. It is important to notice that the ITRs presented in this work are not really comparable. The implemented configurations work nearly continuously and allow the user to change the state within 200 ms. The trial durations for the ITR are based on the latency of the system. Nevertheless, the high ITR of 108 bits/min in (Bin et al., 2011) results from the 32 targets (the BCI presented in this work uses only four targets). The c-VEP configuration can also be used with more controls, than the four targets used in the performed experiments.

The pseudo zero class allows the user to remain in an idle state. However, it is no real zero class and not trained by the system. The user can set a threshold for the confidence interval of valid classification. Therefore, the system rejects false positive classification, with the tradeoff of additional false negative classifications. This is also the reason for the decreased performance, when the pseudo zero class is enabled. The value for the maximum false positive classifications (11.82 %) is just estimation and should be replaced by a false positive test run that lasts several minutes, to see the false positive classification, when no target is selected.

5. Conclusion

In all tests the c-VEP configuration showed significant improvements compared to f-VEP LED configuration. It performed better than the f-VEP configurations in all tests. Also the c-VEP on-screen configuration is faster than the other configurations. Therefore this system is the recommended BCI for further experiments. The BCI-Overlay is working as good as the LED stimulation and provides the possibility to include BCI controls in any OpenGL based graphics application.

6. Acknowledgement

I want to thank Dr. Christoph Guger, who supported me all the time and gave me the chance to participate in this fascinating field of research. Also I want to thank Dr. Christoph Hintermüller for supervising me, for his great introductions into new topics and our interesting discussions. Not forgetting Clemens, Robert, Rupert and all the other likeable colleagues from g.tec, who helped me, whenever I had a problem.

I want to thank Assoc.Prof. Dr. Gernot Müller-Putz from the University of Technology in Graz for his supervision and his helpful suggestions during our meetings.

Related to the VERE project, I thank Angelika Peer, Robert Jenke and Mohammad Abu-Alqumsan from the Technical University of Munich, Franko Teccia from Scuola Superiore Sant'Anna and Kristopher Blom and Mar González-Franco from the Universtat de Barcelona for their support during this thesis.

Of course, I want to thank my parents and my friends for their love and support over all the years.

Especially, I want to thank my girlfriend Ines for her love, also in stressful and hard times, when everything seemed to go wrong.

This research is supported by the European Union FP7 Integrated Project VERE (No. 257695).

7. Appendix

7.1. XML Mask-File

This section contains the specification of the mask-file, which is used to define the BCI-controls. Important for the usage of SSVEP controls is the *Image* type. To distinguish SSVEP controls from any other icon, the *Image* type contains a *Format* attribute. If this attribute is set to “SSVEP”, the control will be recognized by the BCI-Overlay. The data of the Image can be either the string “solid” for a rectangular control or a hex-code of the desired bitmap. Within the *MaskConfig* type the screen is divided into a matrix, in which the controls can be aligned. Also the control size is defined in the *MaskConfig* type. In the mask, a sequence of commands based on the *SingleCommandType* specifies all BCI controls, which are all flickering at the same time. The mask-file contains additional attributes due to its usage in other BCI types. However, they are not needed by the SSVEP based BCI and therefore not further explained (Putz et al., 2011).

```
<?xmlversion="1.0"encoding="iso-8859-1"?>
<xs:schemaxmlns:xs="http://www.w3.org/2001/XMLSchema">
<!--
  Definition of different types:
  DispSymbolType: Sympbols will be able to be displayed only by a text or
by an ICON which
                  for which only the path where the icon is stored has to
be defined

  For the control interface there are three different possibilites of
commands:

  SingleCommandType: This type states a single command. This type requires,
the instruction, the command position
                     who the command should be displayed at the interface
(either text or icon) and the command type (see below)
                     and optionally a parameter. The type requires the
command name as an attribute and the optional the GroupTag
                     or the command ID optionally
-->

<!-- Definition of the complex type image-->
<xs:complexTypename="Image"mixed="true">
<xs:attributename="Format"use="required">
<xs:simpleType>
<xs:restrictionbase="xs:string">
<xs:enumerationvalue="SSVEP"/>
<xs:enumerationvalue="hex"/>
<xs:enumerationvalue="hexRGB"/>
<xs:enumerationvalue="binary"/>
</xs:restriction>
</xs:simpleType>
</xs:attribute>
```

```
</xs:complexType>

<!-- Definition of the MaskType (lines and cols are restricted to a number
>= 2 and <= 10-->
<xs:complexTypename="MaskConfig">
<xs:sequence>
<xs:elementname="NoLines"minOccurs="1"maxOccurs="1">
<xs:simpleType>
<xs:restrictionbase="xs:integer">
<xs:minInclusivevalue="1"/>
<xs:maxInclusivevalue="10"/>
</xs:restriction>
</xs:simpleType>
</xs:element>
<xs:elementname="NoCols"minOccurs="1"maxOccurs="1">
<xs:simpleType>
<xs:restrictionbase="xs:integer">
<xs:minInclusivevalue="1"/>
<xs:maxInclusivevalue="10"/>
</xs:restriction>
</xs:simpleType>
</xs:element>
<xs:elementname="BtnSize"minOccurs="1"maxOccurs="1">
<xs:simpleType>
<xs:restrictionbase="xs:double">
<xs:minExclusivevalue="0"/>
<xs:maxInclusivevalue="1"/>
</xs:restriction>
</xs:simpleType>
</xs:element>
<xs:elementname="SelectedBox"type="xs:string"minOccurs="0"maxOccurs="1"/>
<xs:elementname="EditBox"minOccurs="0"maxOccurs="1">
<xs:complexTypemixed="true">
<xs:attributename="Rows"use="required"type="xs:integer"/>
</xs:complexType>
</xs:element>
<xs:elementname="Logo"minOccurs="0"maxOccurs="unbounded">
<xs:complexType>
<xs:sequence>
<xs:elementname="Path"type="xs:string"/>
<xs:elementname="Position"type="xs:string"/>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>

<!-- Definition of the DispSymbolType: Display of a command (text or icon)-
->
<xs:complexTypename="DispSymbolType">
<xs:sequence>
<xs:elementname="Text"type="xs:string"minOccurs="1"maxOccurs="1"/>
<xs:elementname="Icon"type="Image"minOccurs="0"maxOccurs="1"/>
</xs:sequence>
</xs:complexType>

<!-- Definition of the SingleCommandType: This type states a single
command. This type requires, the instruction, the command position
                     who the command should be displayed at the interface
(either text or icon) and the command type (see below)
```

```
                     and optionally a parameter. The type requires the
command name as an attribute and the optional the GroupTag
                     or the command ID optionally-->
<xs:complexTypename="SingleCommandType">
<xs:sequence>
<xs:elementname="ICONPosition"type="xs:string"/>
<xs:elementname="DispSymbol"type="DispSymbolType"/>
<xs:elementname="CommType">
<!--Different command types are:
            single:
            doubleselect: for secrete operation, when a choice have to be
done twice before execution
            closemask:    this command type will close a submask
-->
<xs:simpleType>
<xs:restrictionbase="xs:string">
<xs:enumerationvalue="single"/>
<xs:enumerationvalue="standby"/>
<xs:enumerationvalue="doubleselect"/>
</xs:restriction>
</xs:simpleType>
</xs:element>
</xs:sequence>
<xs:attributename="CmdID"type="xs:string"use="required"/>
</xs:complexType>

<!--    Start definition of the xml file: The "Mask" includes a -->

<xs:elementname="Mask">
<xs:complexType>
<xs:sequence>
<xs:elementname="MaskConfig"type="MaskConfig" />
<xs:elementname="ControlTyp"minOccurs="1"maxOccurs="1">
<xs:simpleType>
<xs:restrictionbase="xs:string">
<xs:enumerationvalue="continuous"/>
<xs:enumerationvalue="singleFlash"/>
<xs:enumerationvalue="RCFlash"/>
<xs:enumerationvalue="patternFlash"/>
<xs:enumerationvalue="cursor2D"/>
<xs:enumerationvalue="cursor1D"/>
<xs:enumerationvalue="cursor1Dfeedback"/>
</xs:restriction>
</xs:simpleType>
</xs:element>
<xs:elementname="Symbols"maxOccurs="unbounded">
<xs:complexType>
<xs:sequence>
<xs:elementname="SingleCommand"type="SingleCommandType"maxOccurs="unbounded
"minOccurs="0"/>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:sequence>
<xs:attributename="MaskName"type="xs:string"use="required"/>
</xs:complexType>
</xs:element>
</xs:schema>
```

8. **References**

ALLISON, B., LUTH, T., VALBUENA, D., TEYMOURIAN, A., VOLOSYAK, I. & GRASER, A. 2010. BCI Demographics: How Many (and What Kinds of) People Can Use an SSVEP BCI? *Ieee Transactions on Neural Systems and Rehabilitation Engineering,* 18**,** 107-116.

ALLISON, B. Z., MCFARLAND, D. J., SCHALK, G., ZHENG, S. D., JACKSON, M. M. & WOLPAW, J. R. 2008. Towards an independent brain-computer interface using steady state visual evoked potentials. *Clin Neurophysiol,* 119**,** 399-408.

BACH, M., HAARMEIER, T. & DICHGANS, J. 2005. Visuell evozierte Potenziale und Elektroretinogramm. *In:* STÖHR, M., DICHGANS, J., BUETTNER, U. W. & HESS, C. W. (eds.) *Evozierte Potenziale.* 4th ed.: Springer Medizin Verlag.

BARTL, G., VAN LITH, G. H. & VAN MARLE, G. W. 1978. Cortical potentials evoked by a TV pattern reversal stimulus with varying check sizes and stimulus field. *Br J Ophthalmol,* 62**,** 216-9.

BIN, G., GAO, X., WANG, Y., LI, Y., HONG, B. & GAO, S. 2011. A high-speed BCI based on code modulation VEP. *J Neural Eng,* 8**,** 025015.

BIN, G., GAO, X., YAN, Z., HONG, B. & GAO, S. 2009a. An online multi-channel SSVEP-based brain-computer interface using a canonical correlation analysis method. *J Neural Eng,* 6**,** 046002.

BIN, G. Y., GAO, X. R., WANG, Y. J., HONG, B. & GAO, S. K. 2009b. VEP-Based Brain-Computer Interfaces: Time, Frequency, and Code Modulations. *Ieee Computational Intelligence Magazine,* 4**,** 22-26.

BISHOP, C. M. 1995. *Neural Networks for Pattern Recognition,* Oxford, Clarendon Press.

BORGA, M. 2001. Canonical Correlation: A Tutorial.

CHENG, M. & GAO, S. 1999. An EEG-based cursor control system. *[Engineering in Medicine and Biology, 1999. 21st Annual Conf. and the 1999 Annual Fall Meeting of the Biomedical Engineering Soc.] BMES/EMBS Conference, 1999. Proceedings of the First Joint.*

COYLE, S., WARD, T., MARKHAM, C. & MCDARBY, G. 2004. On the suitability of near-infrared (NIR) systems for next-generation brain-computer interfaces. *Physiol Meas,* 25**,** 815-822.

EPSTEIN, C. M. 2011. Analog Signal Recording Principles. *In:* SCHOMER, D. L. & SILVA, F. H. L. (eds.) *Niedermeyer's Electroencephalography: Basic Principles, Clinical Applications, and Related Fields.* 6th ed.

FALLER, A., SCHÜNKE, M. & SCHÜNKE, G. 2004. *The Human Body: An Introduction to Structure and Function*.

FALLER, J., MÜLLER-PUTZ, G., SCHMALSTIEG, D. & PFURTSCHELLER, G. 2010. An Application Framework for Controlling an Avatar in a Desktop-Based Virtual Environment via a Software SSVEP Brain-Computer Interface. *Presence-Teleoperators and Virtual Environments,* 19**,** 25-34.

FISHER, R. S., HARDING, G., ERBA, G., BARKLEY, G. L. & WILKINS, A. 2005. Photic- and pattern-induced seizures: a review for the Epilepsy Foundation of America Working Group. *Epilepsia,* 46**,** 1426-41.

FRIMAN, O., VOLOSYAK, I. & GRASER, A. 2007. Multiple channel detection of steady-state visual evoked potentials for brain-computer interfaces. *IEEE Trans Biomed Eng,* 54**,** 742-50.

G.TEC 2011a. g.GAMMAcap2 - Instructions for use. Schiedlberg: g.tec medical engineering GmbH.

G.TEC 2011b. g.GAMMAsys - Instructions for use. Schiedlberg: g.tec medical engineering GmbH.

G.TEC 2011c. g.STIMbox - Instructions for use. Schiedlberg: g.tec medical engineering GmbH.

G.TEC 2011d. g.USBamp - Instructions for use. Schiedlberg: g.tec medical engineering GmbH.

G.TEC 2011e. Product Catalogue. Schiedlberg: g.tec medical engineering GmbH.

GAMMA, E., HELM, R., JOHNSON, R. & VLISSIDES, J. 2004. Erzeuger Muster. *Entwurfsmuster.* Addison-Weslay.

GUGER, C., SCHLOGL, A., NEUPER, C., WALTERSPACHER, D., STREIN, T. & PFURTSCHELLER, G. 2001. Rapid prototyping of an EEG-based brain-computer interface (BCI). *Ieee Transactions on Neural Systems and Rehabilitation Engineering,* 9**,** 49-58.

HARTER, M. R. 1970. Evoked Cortical Responses to Checkerboard Patterns - Effect of Check-Size as a Function of Retinal Eccentricity. *Vision Research,* 10**,** 1365–1376.

HJORTH, B. 1975. An on-line transformation of EEG scalp potentials into orthogonal source derivations. *Electroencephalogr Clin Neurophysiol,* 39**,** 526-30.

JIA, C. A., GAO, X. R., HONG, B. & GAO, S. K. 2011. Frequency and Phase Mixed Coding in SSVEP-Based Brain-Computer Interface. *Ieee Transactions on Biomedical Engineering,* 58**,** 200-206.

KAISER, K. P. 1996. *Calculation of Visual Angle* [Online]. York University. Available: http://www.yorku.ca/eye/visangle.htm [Accessed Februray 22th 2012].

LALOR, E. C., KELLY, S. P., FINUCANE, C., BURKE, R., SMITH, R., REILLY, R. B. & MCDARBY, G. 2005. Steady-state VEP-based brain-computer interface control in an immersive 3D gaming environment. *Eurasip Journal on Applied Signal Processing,* 2005**,** 3156-3164.

LEUTHARDT, E. C., SCHALK, G., WOLPAW, J. R., OJEMANN, J. G. & MORAN, D. W. 2004. A brain-computer interface using electrocorticographic signals in humans. *J Neural Eng,* 1**,** 63-71.

LI, C. & JASPER, H. 1953. Microelectrode studies of the electrical activity of the cerebral cortex in the cat. *J Physiol,* 121**,** 117-40.

LIN, Z., ZHANG, C., WU, W. & GAO, X. 2006. Frequency Recognition Based on Canonical Correlation Analysis for SSVEP-Based BCIs. *Biomedical Engineering, IEEE Transactions on,* 53**,** 2610-2614.

MARTINEZ, P., BAKARDJIAN, H. & CICHOCKI, A. 2007. Fully Online Multicommand Brain-Computer Interface with Visual Neurofeedback Using SSVEP Paradigm. *Computational Intelligence and Neuroscience,* 2007**,** 9.

MATHWORKS. 2011a. *MATLAB® - External Interfaces* [Online]. Natick: The MathWorks, Inc. Available: http://www.mathworks.com/help/pdf_doc/matlab/apiext.pdf.

MATHWORKS. 2011b. *MATLAB® - Getting Started Guide* [Online]. Natick: The MathWorks, Inc. Available: http://www.mathworks.com/help/pdf_doc/matlab/getstart.pdf.

MATHWORKS. 2011c. *Simulink® - Developing S-Functions* [Online]. Natick: The MathWorks, Inc. Available: http://www.mathworks.com/help/pdf_doc/simulink/sfunctions.pdf.

MATHWORKS. 2011d. *Simulink® - Getting Started Guide* [Online]. Natick: The MathWorks, Inc. Available: http://www.mathworks.com/help/pdf_doc/simulink/sl_gs.pdf.

MCFARLAND, D. J., SARNACKI, W. A. & WOLPAW, J. R. 2003. Brain-computer interface (BCI) operation: optimizing information transfer rates. *Biol Psychol,* 63**,** 237-51.

MCMILLAN, G., CALHOUN, G., MIDDENDORF, M., SCHNURER, J., INGLE, D. & NASMAN, V. 1995. Direct Brain Interface Utilizing Self-Regulation of Steady-State Visual Evoked Response (SSVER). *RESNA 95 Annual Conference.*

MELLINGER, J., SCHALK, G., BRAUN, C., PREISSL, H., ROSENSTIEL, W., BIRBAUMER, N. & KUBLER, A. 2007. An MEG-based brain-computer interface (BCI). *Neuroimage,* 36**,** 581-93.

MIDDENDORF, M., MCMILLAN, G., CALHOUN, G. & JONES, K. S. 2000. Brain-computer interfaces based on the steady-state visual-evoked response. *IEEE Trans Rehabil Eng,* 8**,** 211-4.

MILLAN, J. R., RENKENS, F., MOURINO, J. & GERSTNER, W. 2004. Noninvasive brain-actuated control of a mobile robot by human EEG. *IEEE Trans Biomed Eng,* 51**,** 1026-33.

MONDADA, F., BONANI, M., RAEMY, X., PUGH, J., CIANCI, C., KLAPTOCZ, A., MAGNENAT, S., ZUFFEREY, J.-C., FLOREANO, D. & MARTINOLI, A. 2009. The e-puck, a Robot Designed for Education in Engineering. *9th Conference on Autonomous Robot Systems and Competition.*

MUKESH, T. M. S., JAGANATHAN, V. & REDDY, M. R. 2006. A novel multiple frequency stimulation method for steady state VEP based brain computer interfaces. *Physiol Meas,* 27**,** 61-71.

MÜLLER-PUTZ, G., SCHERER, R., BRUNNER, C., LEEB, R. & PFURTSCHELLER, G. 2008a. Better than random? A closer look on BCI results. *International Journal of Bioelectromagnetism,* 10**,** 52-55.

MÜLLER-PUTZ, G. R., EDER, E., WRIESSNEGGER, S. C. & PFURTSCHELLER, G. 2008b. Comparison of DFT and lock-in amplifier features and search for optimal electrode positions in SSVEP-based BCI. *J Neurosci Methods,* 168**,** 174-81.

MÜLLER-PUTZ, G. R. & PFURTSCHELLER, G. 2008. Control of an electrical prosthesis with an SSVEP-based BCI. *Ieee Transactions on Biomedical Engineering,* 55**,** 361-364.

MÜLLER-PUTZ, G. R., SCHERER, R., BRAUNEIS, C. & PFURTSCHELLER, G. 2005. Steady-state visual evoked potential (SSVEP)-based communication: impact of harmonic frequency components. *J Neural Eng,* 2**,** 123-30.

OLEJNICZAK, P. 2006. Neurophysiologic basis of EEG. *J Clin Neurophysiol,* 23**,** 186-9.

PASTOR, M. A., ARTIEDA, J., ARBIZU, J., VALENCIA, M. & MASDEU, J. C. 2003. Human cerebral activation during steady-state visual-evoked responses. *J Neurosci,* 23**,** 11621-7.

PAULUS, W. 2005. Elektroretinographie (ERG) und visuell evozierte Potentiale (VEP). *In:* BUCHNER, H. & NOTH, J. (eds.) *Evozierte Potenziale, Neurovegetative Diagnostik, Okulographie: Methodik und klinische Anwendung.* 1st ed. Stuttgart: Thieme.

PFURTSCHELLER, G., FLOTZINGER, D. & KALCHER, J. 1993. Brain-computer interface: a new communication device for handicapped persons. *Journal of Microcomputer Applications - Special issue on computer applications for handicapped persons*

16**,** 293-299.

POSTEL, J. 1980. *User Datagram Protocol* [Online]. Available: http://merlot.tools.ietf.org/html/rfc768 [Accessed 19.02.2012 2012].

PRUECKL, R. & GUGER, C. Controlling a robot with a brain-computer interface based on steady state visual evoked potentials. Neural Networks (IJCNN), The 2010 International Joint Conference on, 18-23 July 2010 2010. 1-5.

PURVES, D. 2004. *Neuroscience*.

PUTZ, V., GUGER, C., HOLZNER, C., TORRELLAS, S. & MIRALLES, F. 2011. A Unified XML Based Description of the Contents of Brain-Computer Interfaces. *5th International Brain-Computer Interface Confernece 2011*. Graz.

QUADE, D. 1979. Using Weighted Rankings in the Analysis of Complete Blocks with Additive Block Effects. *Journal of the American Statistical Association,* 74**,** 680-683.

REGAN, D. 1966. An effect of stimulus colour on average steady-state potentials evoked in man. *Nature,* 210**,** 1056-7.

REGAN, D. 1989. *Human Brain Electrophysiology: Evoked Potentials and Evoked Magnetic Fields in Science and Medicine*, Elsevier Science Ltd.

RUSSO, F. D., TEDER-SÄLEJÄRVI, W. A. & HILLYARD, S. A. 2002. Steady-State VEP and Attentional Visual Processing. *In:* ZANI, A., PROVERBIO, A. M. & PROVERBIO, A. (eds.) *The Cognitive Electrophysiology of Mind and Brain* Academic Pr Inc

SHREINER, D. 2009. *OpenGL Programming Guide*.

SHYU, K. K., LEE, P. L., LIU, Y. J. & SIE, J. J. 2010. Dual-frequency steady-state visual evoked potential for brain computer interface. *Neurosci Lett,* 483**,** 28-31.

SUTTER, E. E. 1992. The Brain Response Interface - Communication through Visually-Induced Electrical Brain Responses. *Journal of Microcomputer Applications,* 15**,** 31-45.

SUTTER, E. E. 2001. Imaging visual function with the multifocal m-sequence technique. *Vision Res,* 41**,** 1241-1255.

SUTTON, R. S. & BARTO, A. G. 1998. Softmax Action Selection. *In:* SUTTON, R. S. & BARTO, A. G. (eds.) *Reinforcement Learning: An Introduction.* Cambridge: The MIT Press.

TESSIER-LAVIGNE, M. 2000. *Visual Processing by the Retina: Pricnciples of Neural Science*.

VERE. 2010. *Research* [Online]. Available: http://www.vereproject.eu/.

VIDAL, J. J. 1977. Real-time detection of brain events in EEG. *Proceedings of the IEEE,* 65**,** 633-641.

VOLOSYAK, I. 2011. SSVEP-based Bremen-BCI interface--boosting information transfer rates. *J Neural Eng,* 8**,** 036020.

VOLOSYAK, I., CECOTTI, H. & GRASER, A. 2009a. Optimal visual stimuli on LCD screens for SSVEP based Brain-Computer Interfaces. *2009 4th International IEEE/EMBS Conference on Neural Engineering***,** 440-443.

VOLOSYAK, I., CECOTTI, H., VALBUENA, D. & GRASER, A. 2009b. Evaluation of the Bremen SSVEP based BCI in real world conditions. *2009 Ieee 11th International Conference on Rehabilitation Robotics, Vols 1 and 2***,** 374-383.

VOLOSYAK, I., VALBUENA, D., LUTH, T., MALECHKA, T. & GRASER, A. 2011. BCI Demographics II: How Many (and What Kinds of) People Can Use a High-Frequency SSVEP BCI? *Ieee Transactions on Neural Systems and Rehabilitation Engineering,* 19**,** 232-239.

WANG, Y., GAO, X., HONG, B., JIA, C. & GAO, S. 2008. Brain-computer interfaces based on visual evoked potentials. *IEEE Eng Med Biol Mag,* 27**,** 64-71.

WANG, Y., WANG, Y. T. & JUNG, T. P. 2010. Visual stimulus design for high-rate SSVEP BCI. *Electronics Letters,* 46**,** 1057-1058.

WEISKOPF, N., MATHIAK, K., BOCK, S. W., SCHARNOWSKI, F., VEIT, R., GRODD, W., GOEBEL, R. & BIRBAUMER, N. 2004. Principles of a brain-computer interface (BCI) based on real-time functional magnetic resonance imaging (fMRI). *Biomedical Engineering, IEEE Transactions on,* 51**,** 966-970.

WOLPAW, J. R., BIRBAUMER, N., MCFARLAND, D. J., PFURTSCHELLER, G. & VAUGHAN, T. M. 2002. Brain-computer interfaces for communication and control. *Clin Neurophysiol,* 113**,** 767-91.

ZHU, D., BIEGER, J., GARCIA MOLINA, G. & AARTS, R. M. 2010. A survey of stimulation methods used in SSVEP-based BCIs. *Comput Intell Neurosci***,** 12.

ZSCHOKE, S. & HANSEN, H. C. 2002. *Klinische Elektroenzephalographie*, Springer Berlin Heidelberg.

STATUTORY DECLARATION

I declare that I have authored this thesis independently, that I have not used other than the declared sources / resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

…………………………… ………………………………………………….

(Date) (Signature)