NTNU
Norwegian University of
Science and Technology

Development and Testing of a Sensor Setup for a Ship Captain's Chair

Helge Soltvedt Garsmark

Mechanical Engineering
Submission date: June 2017
Supervisor: Martin Steinert, MTP

Norwegian University of Science and Technology
Department of Mechanical and Industrial Engineering

Acknowledgements

This thesis is the result of a challenge given by PhD candidate Andreas Wulvik, to look in to the possibilities that lies in a ship captain's chair. Thanks to the research community TrollLabs NTNU, and especially Andreas Wulvik and Jørgen A. B. Erichsen for guidance and couching regarding especially experiment setup. Thanks to Martin Steinert for guidance and counseling. Finally I would like to give a special thanks to my partner in crime Erik Borge for sharing the frustration of setting up an experiment.

This page is be intentionally left blank.

Abstract

Human-Computer Interaction systems are rapidly changing. In such interactions, human performance might be essential for the system to function in the best possible way. Thus the need to optimize for interaction design that take human emotion in to account are advancing. Great challenges are involved in capturing emotions, for instance that human emotion are complex, and difficult to quantify. In this thesis theory about body language and emotions are adapted in an attempt to introduce a new tool for Affective Engineering. This is done through developing a pilot chair sensor setup and test its capabilities towards capturing emotional states. A pilot experiment setup is used to test association between posture sensed by the chair and level of mental activation or arousal. Statistical analyzes on the data recorded is done to compare established measures of arousal with posture data from the chair. The analyzes show some tendencies toward association between movement and arousal, but more comprehensive analyzes are needed to introduce the setup as a new tool for affective research. The main take away from this thesis would be the experimental framework that was developed, which with small modification can be used for testing tools to be used in interaction studies.

Keywords: Affective Engineering, Arousal, Body Language, Emotions, Interaction Studies, Unembedded Sensing

This page is be intentionally left blank.

Sammendrag

Human-Computer Interaction systemer endrer seg raskt. I slike interaksjoner kan menneskelig ytelse være alfa omega for at systemet skal fungere på best mulig måte. Derfor har behovet for å optimalisere for interasksjonsdesign som tar hensyn til menneskers mentale tilstand økt betraktelig. Store utfordringer er involvert hva gjelder måling av mental tilstand, for eksempel er mental tilstand kompleks og vanskelig å kvantifisere. I denne oppgaven er teori om kroppsspråk og mental tilstand brukt for å introdusere et nytt verktøy for Affective Engineering. Dette er gjort gjennom å utvikling av et pilotoppsett av sensorer i en stol og testing av oppsettets evne til å måle brukerens mentale tilstand. Et piloteksperiment ble satt opp for å teste assosiasjoner mellom sittemønster og mentalt aktivitetsnivå. Statistiske analyser på innsamlet data ble gjort for å sammeligne etablerte målemetoder for arousal med data fra stolen. Analysene viser noen tendenser hva gjelder assosiasjon mellom bevegelse og arousal, men mer omfattende analyser er nødvendig for å kunne introdusere oppsettet som et verktøy for affektiv forskning. Nøkkelpunktet fra denne oppgaven vil nok være utviklingen av rammeverket for eksperimentet, dette vil med små forrandringer kunne brukes i testing av nye verktøy for interaksjonsstudier.

This page is be intentionally left blank.

Table of Contents

This page is be intentionally left blank.

List of Tables

This page is be intentionally left blank.

List of Figures

Abbreviations

Symbol	=	definition
HCI	=	Human Computer Interaction
ANS	=	Autonomic Nervous System
PNS	=	Parasympathetic Nervous System
SNS	=	Sympathetic Nervous System
HR	=	Heart Rate
HRV	=	Heart Rate Variability
ECG	=	Electrocardiogram
NN	=	Normal-to-Normal
RR	=	Normal-to-Normal
RMSSD	=	Root Mean Square of Sucsessive Differences
NN50	=	Number of Sucsessive NN Intervals With a Difference Greater Than 50 ms
pNN50	=	NN50 Devided by Total Number of NN Intervals
TINN	=	Triangular Interpolation of the NN intervals
FFT	=	Fast Fourier Transform
AR	=	Autoregressive
VLF	=	Very Low Frequency Component
LF	=	Low Frequency Component
HF	=	High Frequency Component
LF/HF	=	Low Frequency Component Devided by High Frequency Component
AD ACL	=	Activation-Deactivation Adjective Check List
A1	=	Energetic Sub-Dimension of Activation-Deactivation Adjective Check List
A2	=	Tiredness Sub-Dimension of Activation-Deactivation Adjective Check List
B1	=	Tension Sub-Dimension of Activation-Deactivation Adjective Check List
B2	=	Calmness Sub-Dimension of Activation-Deactivation Adjective Check List
API	=	Application Programming Interface
EMFi	=	Electromechanical film
FSR	=	Force Sensitive Resistor
NTNU	=	Norwegian University of Science and Technology
PANAS	=	Positive and Negative Affect Schedule
PA	=	Positive Affect
NA	=	Negative Affect
EMG	=	Electromyography
GUI	=	Graphical User Interface
S1	=	Scenario 1
S2	=	Scenario 2
S3	=	Scenario 3

This page is be intentionally left blank.

Chapter 1

Introduction

In an ever growing world of autonomy and automation, the Human-Computer Interaction HCI are constantly evolving, and some years age Picard and Picard (1997) coined the term ”Affective Computing”, for computing that is related to human emotions. In short it is computing affected by human emotions, or computing that can affect human emotions. If you combine the concept of Affective Computing with the concept of Affective Engineering (Balters and Steinert, 2014), a need for interaction design for new unexplored situations will emerge. The ability to measure the emotional responses in an interaction, allows the design of the interaction system and its informational stimuli to facilitate for better user experience and performance. In the design of a new interaction system, interaction studies can be used to measure what kind of stimuli and communication evokes different types of emotions and corresponding performances in different tasks for the system.

Imagine you are a ship captain in the future, in fact your title is not ship captain, but ship monitor. You are responsible for monitoring several autonomous ships, and if needed take over the control. You feel tired and unresponsive and you are counting the seconds before your shift ends. Suddenly you have to take control over two of the ships at the same time, there are alarms going off all around you. You fail your task, and one of the ships crashes in to the docks.

What if in the scenario above, the interaction was designed to take your emotions and affective state in to account. First of all if the system was able to recognize that you were to tired for the task at hand, and your shift should have ended earlier. Second that depending on your affective state, all the alarms are not necessarily helping the situation. What if the system was able to know your capabilities based on your mental state?

But what is human emotion? Simply put can emotions be derived in to two main components, the rational and the affective feelings. The affective feelings are harder to explain compared to the rational feelings, but this thesis will however do an attempt to at least try to explain one dimension of the affective emotions, the level of activation, also known as arousal.

1.1 Scope

In interaction studies the use of physiological responses to measure emotions are growing fast and the need for alternative ways of capturing these responses are growing at the same pace. The use of biometric measurements are establishes and widely used, but most of the sensors used for this are either intrusive or embedded on the bodies of the subjects. This thesis is therefore a call for a non-intrusive and unembedded sensor setup for measuring emotion. Body language can be sensed without the need for embedding sensors on subjects or intrusive sensors. For instance are facial expressions widely used to measure emotional reactions. In this thesis another apprauch are taken, a chair setup, with sensors embedded in the chair rather then on the participants bodies, are piloted and tested as a measurement of emotions. This setup of this chair are explained in detail in **chap. 3**

Arousal or activation has previously been connected to the level of performance (Duffy, 1957). The ability to find a way to measure arousal in interaction studies can hence be helpful in the design of the next study or even the final interaction system. This thesis will focus on developing and piloting a chair sensor setup that can sense various activity patterns, and explore how much information about the user that can be read out of this activity. It will more specifically explore what features regarding the posture measured by the chair setup can give information about the level of arousal.

1.2 Outline

The structure of this thesis is set up in the following way: Theoretical background, selection and development of chair setup, experimental setup, results, evaluations, conclusion.

The theoretical chapter includes relevant background information and previous research used for the work done in this thesis, it also includes a section of similar work done earlier.

In the chapter about the chair setup, evaluations around sensor types and features needed for the setup are taken to consideration. It is important to state that the setup is more a selection based on information rather than a development process were different aspects are tested before they are chosen.

The experimental setup or method chapter is done in collaboration with another master student. It includes some hypotheses and information that is not further discussed in this thesis. However the setup in general is used in the way it is presented.

The result chapter includes descriptive statistics as well as statistical tests and evaluations of the hypotheses.

Lastly reflections and conclusions are discussed based upon the work done.

Chapter 2

Theoretical Background

2.1 Affective Engineering

Affective Engineering or Kansei Engineering was introduced by Nagamachi (1995) as a response to changing consumer desire. It is as a product development method that includes the customers emotions in product design. In short the definition of Kansei Engineering is:

> *"Translating technology of a consumer's feelings (Kansei) and image for a product into design elements"*
>
> -Nagamachi (1995)

Kansei is a japaneese word which means a consumer's psychological feeling and image regarding a new product Nagamachi (1995).

Dahlgaard et al. (2008) suggest "new Kansei Engineering" or Affective Engineering as a way to broaden the scope of Kansei Engineering. They stress the importance of stepping back, and try to understand the user's human needs before trying to design to affect the emotions of the user.

Balters and Steinert (2015) states that most of the existing Kansei tools main focus is deriving insights from triangulation of external factors, like supporting the consumer and the designers decisions (Matsubara and Nagamachi, 1997). Balters and Steinert (2015) believe that to be able to understand the response and the effect on the situational experience of the user, lies in measuring the emotional reactivity, determined by the physiological response from the human body. They describe the keys to understand the behavioral dimensions of the human are the emotions effect on behavior.

Dahlgaard et al. (2008) statement that users are getting more and more concerned about the fact that products or interactions are matching their feelings, might be quite accurate. The need to include the user's or customer's emotions or affect towards the product are hence becoming increasingly important in product development and design. This thesis is stressing and empathizing Balters and Steinert (2014) call for affective engineering.

2.2 Classifying emotions

Regarding the classification of emotion, two schools of thought exist. Were the first school describes the emotions to be discrete and fixed biologically (Ekman and Friesen, 1971). The other school describes the emotions in dimensions of arousal and valence (Wundt and Pintner, 1912). Russell (1978) described a circumplex model of affect with eight words presented in a two-dimensional model **fig. 2.1**, where the y axis are described as the sleep-arousal dimension, and the x-axis represents the pleasure-displeasure or valence dimension. In this model the remaining words do not form independent dimensions, but can be described by the two already established dimensions of arousal and valence.

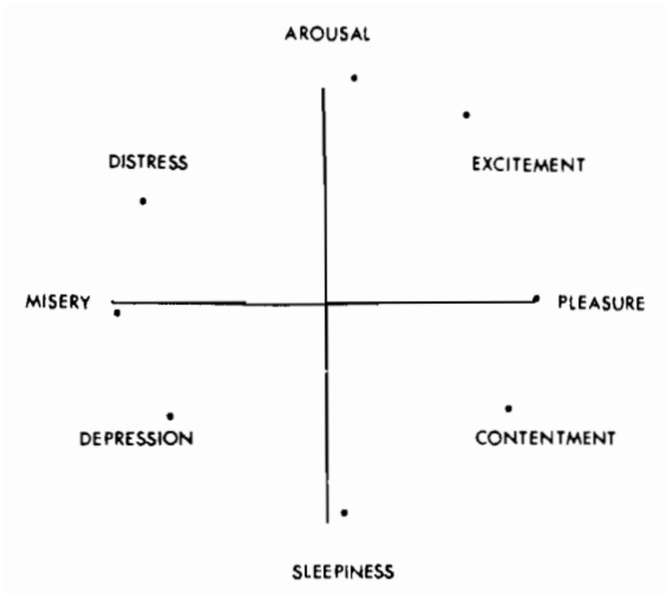


Figure 2.1: Eight addect concepts in a circular order (Russell, 1978)

Russell et al. (1989) empathizes that these two dimensions are not all there is to the concept of affect, but that there are good reasons to evaluate the state of affect using these two dimensions. The concept of arousal refers to a self reported subjective feeling (Russell et al., 1989).

2.3 Arousal

The concept of arousal has been part of psychology for a long time. Both Wundt (1896) and Freud and Strachey (1964) had some kind of model for arousal in their research. Duffy (1962) gave the concept of arousal or activation some more empirical context in her behavioral analyzes. Thayer (1989) refers to arousal as a basic element of mood and behavior, he further differentiate between two types of arousal in his approach, energetic arousal and tense arousal. The energetic arousal involves that with the exertion necessary for an activity the autonomic nervous system ANS switches from being dominated by the parasympathetic nervous system PNS to being dominated by the sympathetic nervous system SNS. The PNS are associated with body maintenance, while the SNS are associated with mobilization for action. The tense arousal are very similar, but the changes in the ANS are triggered by emergency, rather than the energy motivated planned increase in activation in the energetic arousal.

The concept of arousal has had its share of criticism, and according to Thayer (1989) the criticism has limited the concept from being used in a larger scale. Most of the criticism is based upon the fact that different models of arousal does not correlate well enough with each other. One instance of such a criticism is Lacey (1967) who concluded that there are three types of arousal: autonomic, behavioral, and cortical. Thayer (1989) argues that this criticism does not make sense evolutionary, because dissociation between the three types of arousal should not be favorable in natural selection.

Research has been done to find the optimal level of arousal for maximum performance. According to Duffy (1957) the arousal-performance relationship can be represented by an inverse U-curve **fig. 2.2**, where moderate arousal gives best performance. The ability to measure arousal in a non-intrusive non-interrupting way can possibly facilitate for maximum performance potential in interaction studies.

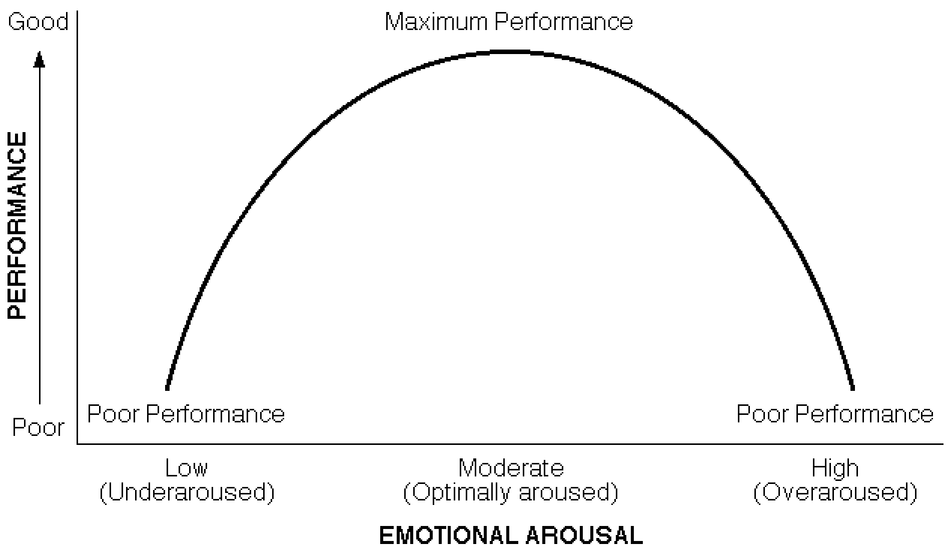


Figure 2.2: Relationship between arousal and performance (Williams, 1993)

2.3.1 Objective Measures Arousal

Even if the reason for stress is psychological there are some documented physiological effects of stress. According to Akselrod et al. (1981) if a person is exposed to stress the ANS is engaged. In this case the SNS is activated, while the PNS is suppressed (Taelman et al., 2009). The result of this effect is increased muscle tension, change in heart rate, change in heart rate variability (HRV), and increased blood pressure (Van Houdenhove, 2005). These physiological reactions are triggered by the secretion of the hormones epinephrine and norepinephrine into the bloodstream (Taelman et al., 2009).

Heart Rate Variability

Taelman et al. (2009) concluded that short term heart rate variability HRV was reduced with a mental task, and that the sympathovagal balance may have been increased. HRV and its features are explained in this section.

HRV can be evaluated with different methods, but the two most used methods are the Time Domain Method and the Frequency domain method. In a continuous electrocardiogram ECG signal, each of the QRS complexes are detected and marked, and the normal-to-normal NN- or RR-intervals are calculated (Saykrs, 1973). In the time domain method the features obtained are mean RR-intervals and their standard deviations, RMSSD the root mean square of successive differences, NN50 number of successive NN intervals with a difference greater than 50 ms, pNN50 the proportion of NN50 divided by total number of NN intervals (Carney et al., 1995). The time domain method also include a couple of geometrical features. The RR triangular index is the integral of the density distribution of NN intervals, and TINN the triangular interpolation of the NN intervals (Acharya et al., 2006). The frequency domain method also known as the spectral density method gives information of how the spectral power is distributed as a function of frequency (Kay and Marple, 1981). There are different methods to calculate this spectrum, but they are generally classified as either parametric or non-parametric (Malik, 1996). The non-parametric methods are generally preferred because of the simplicity of the algorithm and the high processing speed. Fast Fourier Transform FFT is the most common of these non-parametric methods (Malik, 1996), and employs a Hanning window for the spectrum (Welch, 1967). In short term measurements of HRV it is common to devide the spectrum in to three spectral components (Rottman et al., 1990), **tab. 2.1** shows the three components with their respectable bands. The sympathovagal balance is derived from dividing the low frequency component LF with the high frequency component HF (LF/HF). Both Taelman et al. (2009) and Thayer et al. (2012) has found HRV as an indicator of stress, in this thesis it will be treated as an indicator of arousal, even if stress is not a direct synonym for arousal.

Table 2.1: Spectral Components of the frequency domain

Frequency component	**Frequency Band**
Very Low Frequency (VLF)	0.00 Hz - 0.04 Hz
Low Frequency LF	0.04 Hz - 0.15 Hz
High Frequency	0.15 Hz - 0.40 Hz

2.3.2 Subjective Measures Arousal

There are at least 20 different methods of self report to measure affect (Ekkekakis, 2013), but depending on the structure of the study, not all methods fit the experiment. One of the most widely used are the Affect Grid (Russell et al., 1989), which evaluates both dimensions of affect. Another commonly used tool to asses the arousal dimension are the Activation-Deactivation Adjective Check List AD ACL (Thayer, 1967).

Affect Grid

Developed by Russell et al. (1989) the affect grid is a 9x9 matrix of squares **fig. 2.3**, where the middle square represents neutral feelings. The horizontal dimension represents the pleasant-unpleasant dimension or valence dimension. from unpleasant on the left to pleasant on the right. The horizontal dimension is the arousal dimension and represents the degree of arousal or how awake, alert or activated the subject is. The fact that it describes both arousal and valence, and that the inquiry of the emotional state is fast and simple lead to Steinert et al. (2012) and Balters and Steinert (2015) labeled it as the best subjective tool to capture affective emotions.

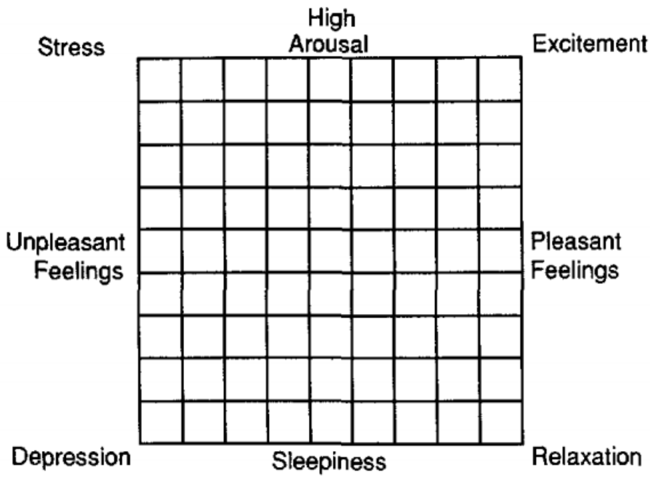


Figure 2.3: The affect grid, represented by a 9x9 matrix of squares. (Russell et al., 1989)

AD-ACL

The activation-deactivation adjective check list or AD ACL is a multidimensional questionnaire test of different aspacts of arousal (Thayer, 1990). According to Thayer (1986) the AD ACL format have showed evidence of being concistant with other models of arousal and mood (Watson and Tellegen, 1985).

AD ACL consists of four sub-dimensions; energetic A1, tiredness A2, tension B1 , and calmness B2. Each of these consists of five words, which are rated on a scale consisting of; 4:definitely feel, 3:feel slightly, 2:cannot decide , and 1:definitely not feel. The value of one to four are then assigned to the word, and the score for the representative sub-dimension are summed up, each with a minimum and maximum of 10 and 40. The short version of AD ACL and its associated words are presented in **tab. 2.2**

Table 2.2: Adjectives from AD ACL, in the same order as presented

Word	**Sub-dimension**
Active	Energetic
Placid	Calmness
Sleepy	Tiredness
Jittery	Tension
Energetic	Energetic
Intense	Tension
Calm	Calmness
Tired	Tiredness
Vigorous	Energetic
At-rest	Calmness
Drowsy	Tiredness
Fearful	Tension
Lively	Energetic
Still	Calmness
Wide-awake	Tiredness
Clutched-up	Tension
Quiet	Calmness
Full-of-pep	Energetic
Tense	Tension
Wakeful	Tiredness

2.4 Body Language

The concept of body language is not a simple concept. Body language is complex, and is derived from several different "channels" of information, for instance body movement, gestures and facial expressions (Wulvik et al., 2016). Wulvik et al. (2016) also separated the tools for measuring body language into manual and automatic tools. Were the manual includes observations and manual video coding. The automatic tools are more interesting, since humans does not have to interpret all the data. These automatic tools rely on sensors and data processing for clustering and classification. They further separated the tools into camera based tools and wearable based tools **fig. 2.4**.

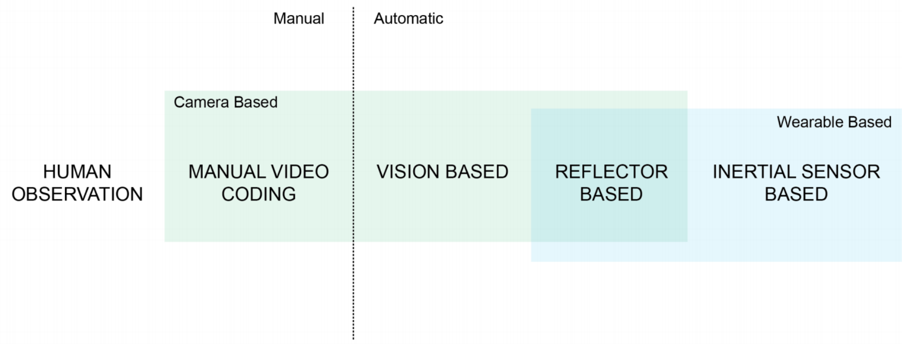


Figure 2.4: Grouping of technologies (Wulvik et al., 2016)

2.4.1 Body Language and Emotions

> *"From everyday experience we know that an angry face is more menacing when accompanied by a fist."*
>
> -De Gelder (2006)

There have been done a lot of research on how human emotion effects facial expressions, however there is little exploration done on how the emotions affects the body movement. This is in spite of (Darwin and Ekman, 1872) having a postural descriptions included in The expression of the emotion in man and animals. Emotion recognition from body movement is still unresolved. Coulson (2004) generated computer figures from descriptions of postural expressions and presented results on six emotions(anger, disgust, fear, happiness, sadness, and surprise). He concluded that recognition from static body postures could be compared to recognition from voice, and some postures are as recognizable as facial expressions. Gunes and Piccardi (2007) did a bi-modal emotion recognition study with the two modalities being facial expressions and upper-body gestures. They found that the classification using both modalities outperformed both the mono-modal recognition classifications. Since bodies are large, compared to for example faces, and have the possibility to show a waste amount of different static postures, if the static postures is combined with a similar amount of movements, the channel of information gathered from body language would be ideal for affective communication (Coulson, 2004). The affective state of a person can be decoded more precisely over distances with body postures compared to facial expressions (Walk and Walkers 1988). This also implies that it might be possible to decode some affective states with a sensor setup that classify the body postures and possible the change in postures(movement). This sensor setup is what is going to be explored in this thesis.

2.5 Similar work

Woolf et al. (2010) used a combination of a pressure sensing chair, a pressure sensing mouse, a camera, and a skin conductance bracelet to measure physiological responses that are linked to affective states. They used this in combination with self report and human observation to classify the affective states of students, and was in some degree able to do so, but in their step-wise regression model, they found that the pressure sensing chair features was not among the most important features.

Kapoor et al. (2004) developed a multimodal framework for recognition of affective states. In their experiments used among other modalities postures and activity on chair, they used a neural network to classify postures and levels of activity from a sensor sheet with 42 by 48 sensing units placed on a chair. They found that from the modalities they tested these were the ones that had the highest recognition rate when performance from individual modalities was tested. Which means that the individual contribution to the final classifier was higher from the posture and activity features, compared with facial data and game stimuli data.

Chapter 3

Sensor Chair

As a part of a process where research are to be done in a ship simulator and possibly even on a real ship bridge, the chair described in this chapter is the first version of a sensor based chair which are going to be further developed in the future. It will later be used for testing of several aspects of the in situation interactions and stimuli. In this part of the development process the chair is going to be tested in a more neutral environment compared to the in-situ situation that is represented by a ship simulator or a real ship. In this more neutral environment the idea is to explore and evaluate what is possible to get out of a sensor embedded chair.

In this chapter the challenges of embedding a chair with sensors are presented and given some context. It is important to make it clear that the development of the chair is by intention done in small increments, where this is the first increment. The development is done in increments because it is easier to evaluate different aspects of the solution this way, rather than developing a complete solution before any testing is done. Although the chair is only in its first stage, there are several aspects that can be tested, however in this thesis the evaluation is focused on measurements of affect, more on this is explained in the next chapter. Hopefully these evaluations and results will ultimately indicate what to explore next, and how to further develop the chair.

3.1 Development challenge

The idea is to implement a chair with some kind of sensor setup to be able to recognize the user's behavior. In this case the interesting features are sitting position, level of forward leaning, utilization of arm rest, position changes and heart rate. The thought is to use these features to examine if they have some correlation with emotional states. As mentioned earlier regarding heart rate, more specifically HRV, there have been proven correlation to arousal and stress. The main focus is hence to try to find some correlation concerning the posture or movement based features.

3.2 Key Features

Before developing a sensor based chair it is important to make it clear which features the chair needs to be able to sense. These features will be presented in this section.

3.2.1 Posture/ Sitting Position

The chair have to be able to differentiate between different postures or sitting position in this case. This is important because with this feature it is possible to explore the correlation between postures and emotions. It is also necessary to have this feature to be able to obtain the next key feature, which is position change.

3.2.2 Position Change

Based on the feature of sitting position, the setup have to be able to record the number of position changes for a given period of time, to be able to do this the sampling has to be continuous and the threshold must be given by the before mentioned feature of sitting positions. Again this feature is important because correlation between number of changes and emotions are to be evaluated.

3.2.3 Heart rate and HRV

Although not taken in to consideration at this stage of the chair development, a future version of the chair should possibly include the ability to sense at least heart rate, and possibly also be able to extract HRV features from this data.

3.3 Sensor Exploration

Based on the above mentioned scope, it was challenging to find a suitable technology for the sensor setup. Following are some of the different technologies and setups that was considered.

3.3.1 Optical Fiber

One of the most promising setups was the optical fiber sensor based setup. Derived from the (darmachair), it is supposed to be able to sense all the above mentioned features. Also there is some in-house expertise on this kind of sensor (Rayleigh backscatter). However the ready-to-use darmachair does not include an application programming interface API and would be useless in this scenario. Also the available in-house sensor was not chosen, mainly because of it being to comprehensive for the scope of this thesis.

3.3.2 EMFi

The next technology considered was EMFi or electromechanical film. This is a polypropylene film based sensor that is sensitive to forces normal to the materials surface (Kärki et al., 2009). This material is commercially available from the Finnish company Emfit Ltd. Anttonen and Surakka (2005) actually used this kind of sensor to make a chair for sensing heart rate in an experiment were they used emotional provocative stimuli, they successfully measured heart rate responses that differed significantly between positive and negative emotions. The uncertainty and complexity regarding using this film material lead to it not being chosen for the chair at this time.

3.3.3 Eontex fabrics

These are fabrics that have been coated with Eonyx to create different types of fabric-based sensors. They are conductive and the resistance depends on both distance and strain applied pressure or force. It is possible to sense location, pressure, stretch and bend with these sensors, and they can even be combined to sense a combination of the before mentioned. However these material based sensors was discovered far to late for them to be applicable in the solution in this version, but they can possible be implemented in a later version of the setup.

3.3.4 FSR - Force sensitive resistors

The force sensitive resistors (FSRs) will have a resistance that is dependent on the pressure that is applied to its surface. More total force gives lower resistance and vice versa. The sensors are somewhat flexible and flat, and to implement them in a chair is highly possible and was actually done as a student project here at NTNU last year. The students put these sensors in an office chair **Fig. 3.1** which combined with a machine learning program allowed them to differentiate between different pre-trained sitting positions (Jensen et al., 2016).

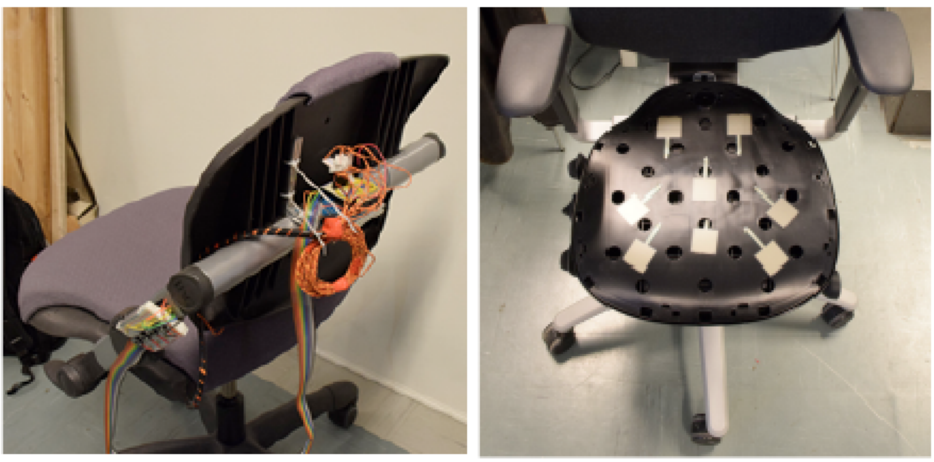


Figure 3.1: The student project chair, (Jensen et al., 2016)

This ended up being the chosen setup for the chair. The process of putting the chair sensor setup together will be described in detail in the next section.

3.4 The Sensing Chair

As in future research the chair are to be used in a in-situ environment, possibly in a ship simulator, the environment of the simulator was explored. **Fig. 3.2** show how the captain's chair look in Kongsberg Maritime's ship simulator.

Figure 3.2: Kongsberg Maritime chair in ship simulator

The chair used in this setup is a Captains Ship Bridge Chair provided from Kongsberg Maritime **Fig. 3.3**, which is similar to the one they have employed in the simulator.

Figure 3.3: Captian's Ship Bridge Chair from Kongsberg Maritime

The FSRs that are most suitable are the type that is square, they have a sensing area that is 4.445 cm x 4.445 cm **Fig. 3.4**. These are the ones with the biggest sensing area of the standard component FSRs that was obtainable within a reasonable time.

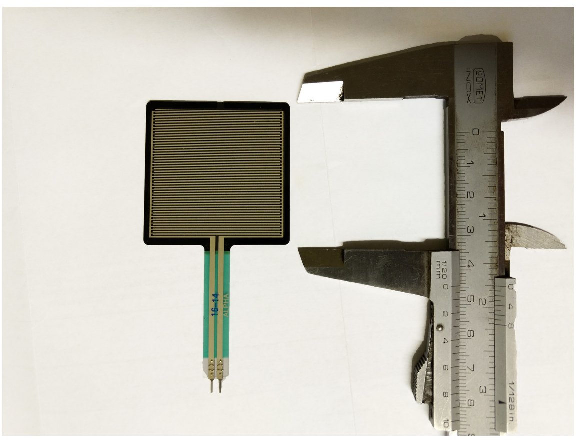


Figure 3.4: Captian's Ship Bridge Chair from Kongsberg Maritime

As mentioned the FSR sensors were chosen to sense the posture and movement in the chair. The placement of the sensors are shown in **Fig. 3.5**. In **fig. 3.6** the mounting of some of the FSRs on the seat can be seen.

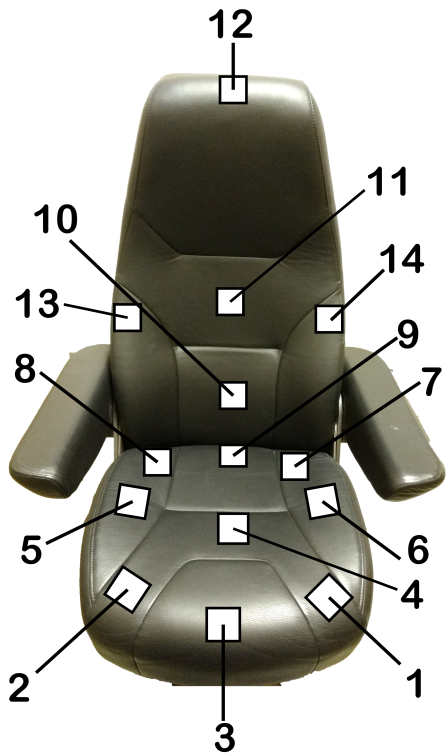


Figure 3.5: The FSR placements on the chair

Figure 3.6: Mounting the sensors

Chapter 4

Experiment Setup

Erik: The following chapter presents the experimental approach and setup used by the author to study how the physiological reactions measured by the chair setup could vary with arousal. The author and another master student have been working on this setup together, as the intention of the two allowed a merge of two experiments. Hereby, this chapter is written by both students. As some tools and theory was applied by one student only, this causes some information from Chapter 2 to be duplicated. The abstract of this student's master thesis is presented to enable an overview of his research:

> *The much-debated topic of physiologically differentiating emotions has been of great interest to the research community in recent times. While many studies focus on facial reactions, the rest of the body is rather unexplored. This thesis takes upon the challenge of experimentally piloting and testing whether we can measure positive and negative emotional valence changes through physiological reactions in a user interaction setting. 34 adult subjects played variations of Tetris while collecting electromyography (EMG) data on the neck and forearm, and leaning distance in a seated position. Three scenarios were designed with different levels of valence and arousal (high valence-low arousal, low valence-high arousal, high valence-high arousal), upon which physiological data was analyzed. Valence was used as a within-subjects factor via self-report measurements by the Russell Affect Grid (Valence-Arousal grid) and Positive Affect Negative Affect Schedule (PANAS). Results showed that there was no statistically significant difference in physiological data between the scenarios (*$p = .761$*,* $p = .191$ *and* $p = .093$ *for EMG neck, EMG arm and leaning distance, respectively). However, the data showed tendencies in increased leaning distance from negative to positive valence. Furthermore, the thesis focuses on how one without much knowledge within the field of human experimentation can prototype and iterate one's way to such an experimental design, applying product development models to achieve this.*
>
> -Erik Aas Borge (2017)

The combined intentions of the experiment was hereby to study how physiological reactions vary with changes in affect, including both valence and arousal. The goal was to design three scenarios to achieve three different levels of affect within the two dimensions, and then evaluate how the physiological data relates with the two. Furthermore, the setup of this experiment was not only result based. It was also a pre-study and an exploration on how one can study physiological reactions and emotional states in a user interaction setting. As such, this experiment is also provided as a pilot and a framework for how one can achieve this. This chapter introduces hypotheses, the complete setup for the experiment as well as the chosen measurement methods and input variables.

4.1 Hypotheses

In the following section a number of hypotheses are presented. All of the hypotheses are listed with their corresponding null-hypotheses. The hypotheses relates to relative change in valence and arousal.

4.1.1 Valence Hypotheses

These valence hypotheses relates to the established subjective measurements of valence and the goal was to explore the relationship between the objective measurements of physiological data, EMG and leaning distance, and the established subjective measurements, the Affect Grid and PANAS. The first hypothesis is regarding valence and its effect on muscle tension:

H1: "Changes in valence induce changes in muscle tension"
H1null: "Changes in valence does not induce changes in muscle tension"

The second hypothesis is regarding valence and its effect on the level of forward leaning, i.e. leaning distance:

H2: "Changes in valence affects the level of forward leaning"
H2null: "Changes in valence does not affect the level of forward leaning"

Hypothesis H1 was tested with EMG data which was compared with the data from the PANAS and the Affect Grid **section 4.7**. Hypothesis H2 was tested with data from a Lidar distance sensor, which was then compared with the same control measurements.

4.1.2 Arousal Hypotheses

The following arousal hypotheses relates to established measurements of arousal. The goal was to test if another objective measurement gathered from features from the chair setup were associated with the established subjective and objective measurements. The first hypothesis regarding arousal is connected to the effects arousal has on number of posture changes in a chair.

H3: "Changes in arousal affect the number of position changes in a chair in a given time period "
H3null: "Changes in arousal does not affect the number of position changes in a chair in a given time period"

The second hypothesis is regarding arousal and its effect on the level of forward leaning:

H4: "Changes in arousal affect the level of forward leaning"
H4null: "Changes in arousal does not affect the level of forward leaning"

Hypotheses H3 and H4 were tested with data from the chair that was processed through machine learning to classify positions and level of forward leaning. This data was then compared with both the subjective self report AD-ACL-questionnaire and objective HRV measurements. H4 was also tested with the Lidar distance sensor data.

4.2 Independent Variables

Since there was no direct way to vary the affect of the subjects, stimuli was used to achieve the desired emotional state. The stimuli represents the independent variable, where this was tuned to guide subjects in the wanted directions of affect. Following are the different means of stimuli used, and a short explanation of how they were adjustable. How they were implemented in this experiment is described in **section 4.9**.

4.2.1 Difficulty

The task, in this case TETRIS (described in 4.6), varied in difficulty. The manipulated version of the game allowed for adjustable speed, reversing the controls and to decide the order of the pieces, including which pieces to be included at all.

4.2.2 Lights

This was a visual stimuli that was in the form of a LED-strip around the LCD-screen. It could vary in intensity and color, be turned on/off and the speed of all these variations could be controlled.

4.2.3 Music

This was an audible stimuli. There are infinite aspects of music that could be controlled. The aspects that were given most thought in this experiment were tempo, pitch and familiarity.

4.2.4 Feedback

This was both a visual and audible stimuli and was controlled to match the performance and desired affect level. The feedback consisted of several elements, including a continuously changing feedback bar on the screen, score of the game and score relative to other people. It could differentiate from positive to negative loaded with both color and sound, and could be enabled and disabled.

4.3 Dependent Variables

For this experiment several dependent variables were used, both subjective and objective. Heart rate variability (HRV) and all of the subjective variables are established as measurements of the affect dimensions. EMG and the posture measurements (both chair and leaning distance) are the proposed new measurements for valence and arousal. The intention was to compare the EMG and leaning data with the established valence variables, and all the posture data with the established arousal variables. The variables are presented in **table 4.1**.

Variable	Measurement	Subjective or objective
Valence	Panas	Subjective
Valence and arousal	Affect Grid	Subjective
Arousal	AD-ACL	Subjective
Arousal	ECG - HRV	Objective
Proposed valence	EMG - Muscle tension	Objective
Proposed valence	Chair posture data	Objective
Proposed arousal	Chair posture data	Objective

Table 4.1: Dependent variables

4.4 Physical Setup

In the physical aspects of the experiment setup it was important to have a minimum of external ”noise” to bias the data. In this experiment the physical setup was constructed by a mock-up room room **Fig. 4.1** where the test subjects participate without any human interaction after the initial briefing and sensor connection **Fig. 4.11**. That being said, the room was in a busy hallway close to a hallway door that was often being opened and slammed, that lead to some interrupting sounds and vibrations. Also the room requires to have a window open and a lot of outside sounds could be heard by the subjects. The intention behind this experiment does not require the setup to be ”white-room” and totally isolated from external factors, which also would have been difficult in the scope of the project. But it is was supposed to be a lot more ”neutral” compared to an in-situ setup which is a setup that is supposed to represent a situation that is as close to a real life situation as possible.

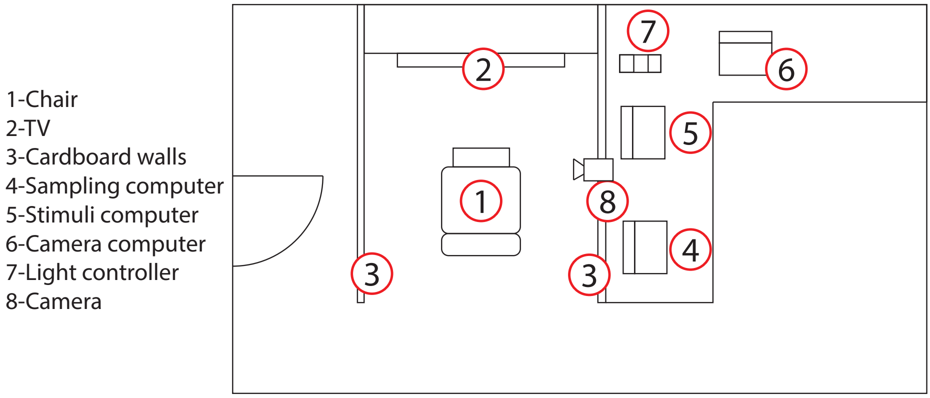
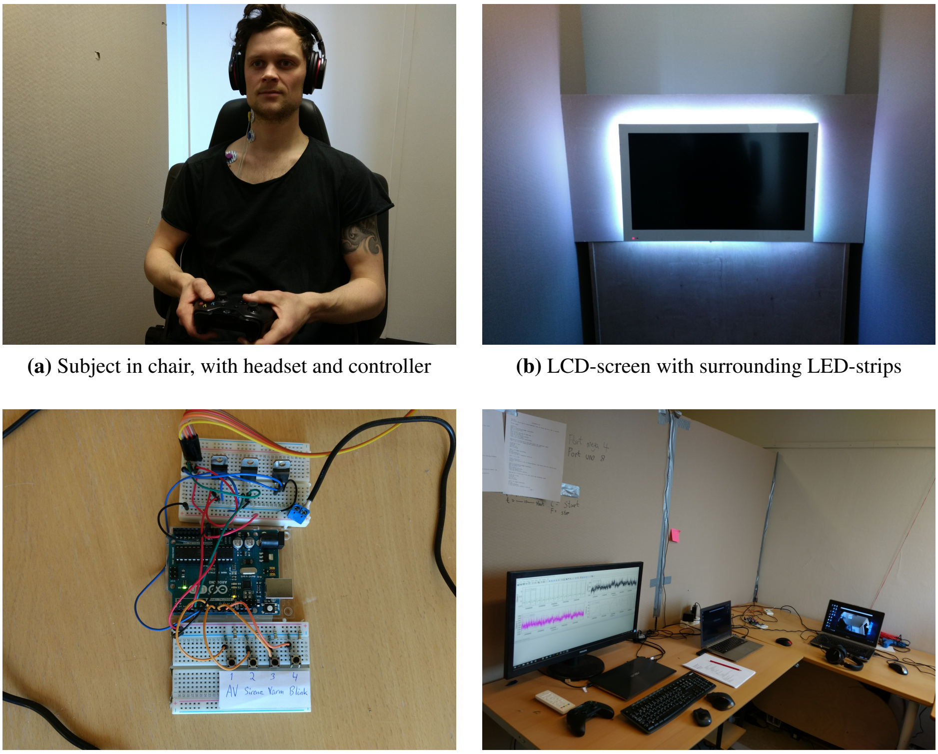


Figure 4.1: Room layout.

Figure 4.2: Overview of the physical setup

The subjects were placed in the chair, where they were initially briefed and connected to the physiological sensors **Fig. 4.11**. The subjects were given a wireless headset for sound stimuli and they interacted with the experiment interface with a wireless Xbox One-controller **Fig. 4.3a**. The interface was presented to the subjects on a 32" LCD screen with surrounding LED-strips for light stimuli **4.3b**. These LED-strips were controlled with an Arduino controlled button circuit **Fig. 4.3c**. Information sheets about how to use the Russel Affect Grid (**Appendix B**) were placed on the subjects left hand side. The subjects were isolated from the rest of the room with two cardboard walls to eliminate distractions from external effects in the room **Fig. 4.2**. Behind the cardboard wall on the chairs right hand side, the computer setup was placed **Fig. 4.3d**, this is where the experiment was controlled and monitored. This is explained in detail in the next section. A camera was placed on the top of the wall to the right of the subject for video recording **Fig. 4.1**.



(a) Subject in chair, with headset and controller

(b) LCD-screen with surrounding LED-strips

(c) Arduino controlled light circuit

(d) The computer setup

Figure 4.3: Other aspects in the physical setup

4.5 Computer Setup

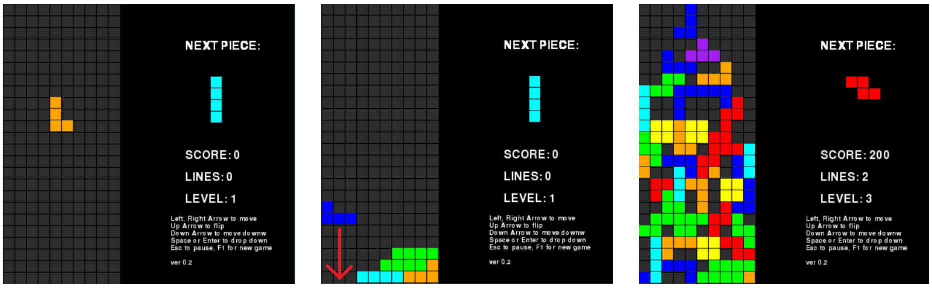
The vast amount of data collection and impulses required great control of the experimental environment. Three computers were used to ensure complete overview and control **(Fig. 4.4)**. The sampling computer was set up to gather all sensory data from the subjects, meaning ECG, EMG and posture both from the chair sensors and the Lidar distance sensor. The sensor platform, consisting of four Arduinos, provided the sensory data over a serial connection with software on the sampling computer capturing this and displaying it in real-time on screen. This allowed for substantial control of the input data, especially in the first phase of the experiment where the connection of the electrode pads for ECG and EMG sometimes didn't provide a good data stream. If some of the electrodes weren't connected well enough this could be easily seen and corrected. The sensor platform in itself will be fully explained and discussed in **section 4.7**. The stimuli computer functioned solely to provide the test subjects with the user interface part of the experiment. Here, the interface software, which will be explained in detail in (section 4.8), guided the user through the whole experiment on the TV screen. This made it possible to avoid interference with the subjects during testing, and at the same time ensured the exact same user experience for every subject. The third computer, the camera computer, was used to do video recordings as well as allowing visual observation of the subjects.

Figure 4.4: Overview of the computer setup

4.6 TETRIS

The traditional game TETRIS was applied as the main task of the experiment. TETRIS is a simple game where a grid defines the playing area. Pieces put together in different shapes fall down from the top one by one **Fig. 4.5a**, and the goal of the game is to puzzle these pieces together at the bottom so that you cover a full line with pieces across the grid

Fig. 4.5b. This line then disappears, and you receive points for each line. The game is over when the stack of bricks reaches the top of the grid **Fig. 4.5c**.



(a) Pieces fall down from the top

(b) Making complete lines gives points

(c) Game over when the stack reaches the top

Figure 4.5: Tetris

The TETRIS game used in this experiment was a clone of the original game, downloaded from GitHub. It was coded from scratch in the programming language Python and uses a series of functions from Pygame, which is a graphical interface pack compatible with Python. The fact that the whole game is constructed by just under 600 lines of code made i fairly easy to tweak for the purposes of the experiment, and design several versions of the game to achieve the different levels of affect intended. Moreover, it allowed for fast prototyping of the game and made it easy to test changes and see how they affected pilot subjects. The final versions are explained further in **section 4.9**.

4.7 Sensor platform and Measurements

In this section the different measurements, both subjective and objective, are described. Also the sensors and tools used to collect some of these measurements are described in this section.

4.7.1 Affect - self report measurements

To be able to have some control measurements in the experiment, some subjective measurements of affect was needed. The Russel Affect Grid is as mentioned in **chap. 2** an established agent of measuring both dimensions of affect (arousal and valence). However we felt it was needed to include at least one more control measurement of both dimensions, and PANAS and AD-ACL was chosen for their proven reliability. Following are some short explanation of these measurements, a more detailed explanation of all three is included in

Russel Affect Grid

The grid consist of a 9x9 matrix, the horizontal dimension represents valence and the vertical dimension represents arousal. This is described by Russel et al. (1989) as a map representation of feelings. The top right quadrant represents feelings of excitement, the bottom right quadrant represents feelings of relaxation, the bottom left represents feelings of depression and sadness, and lastly the top left quadrant represents distressing and tense feelings.

PANAS

The Positive and Negative Affect Schedule (PANAS) (Watson et al., 1988) is a questionnaire to measure valence, similar to the horizontal axis in the Russel Affect Grid. The subjects are presented with a mix of positively and negatively loaded adjectives. The subjects are to indicate the extent they feel these feelings on a scale with the following scale points: 1. Very slightly or not at all, 2. A little, 3. Moderately, 4. Quite a bit, 5. Extremely. The words are presented in **table 4.2**. The sub scales, *Positive affect (PA)* and *Negative affect (NA)*, contain the words 1, 3, 5, 9, 10, 12, 14, 16, 17, and 19, and 2, 4, 6, 7, 8, 11, 13, 15, 18, and 20, respectively. The subjective rating of each word is summed up, and the mean momentary scores of PA and NA are 29.7 and 14.8 respectively.

no.	Word
1.	Interested
2.	Distressed
3.	Excited
4.	Upset
5.	Strong
6.	Guilty
7.	Scared
8.	Hostile
9.	Enthusiastic
10.	Proud
11.	Irritable
12.	Alert
13.	Ashamed
14.	Inspired
15.	Nervous
16.	Determined
17.	Attentive
18.	Jittery
19.	Active
20.	Afraid

Table 4.2: Words of the PANAS questionnaire

AD ACL

The Activation-Deactivation Adjective Check List (AD ACL) is also a questionnaire where the subjects are presented with adjectives which they are to indicate on a scale their extent of feeling the specific feeling. But rather than being a measurement of valence like PANAS AD ACL is a measurement of arousal. The scale points in the Ad ACL questionnaire are: 1. Definitely do not feel, 2. Cannot decide, 3. Feel slightly, 4. Definitely feel

Number	Word
1	Active
2	Placid
3	Sleepy
4	Jittery
5	Energetic
6	Intense
7	Calm
8	Tired
9	Vigorous
10	At-rest
11	Drowsy
12	Fearful
13	Lively
14	Still
15	Wide-awake
16	Clutched-up
17	Quiet
18	Full-of-pep
19	Tense
20	Wakeful

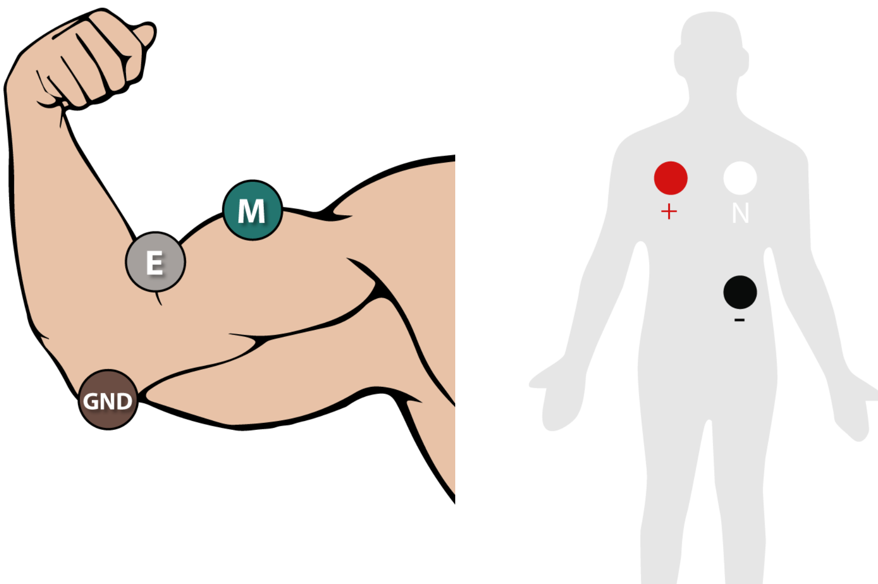
Table 4.3: Word of the AD-ACL questionnaire

4.7.2 Chair sensors and measurements

Force sensitive resistors (FSRs), were embedded in the chair. They acted as pressure sensors where the the resistance varied depending on the pressure applied. The intention of the chair sensor setup in this experiment was to provide measurements of the subjects posture during the experiment. This included sitting position, number of changes in sitting position in a given period of time, and in which degree the subjects was leaning forward. The forward leaning was also measured by using a Lidar distance sensor placed on the back of the chair, behind the subjects' heads. The Lidar sensor was chosen, even with some issues with stability in measurements, mainly because the only other option taht was available in this time scope was a HC-SR04 ultra sonic sensor, but the ultra sonic distance measurements gave sky high peak readings. In comparison the Lidar is highly reliable, although it has fairly low accuracy for the purpose of this experiment ($\pm 2.5cm$). All the above mentioned data was collected using an Arduino Mega, which had a sampling rate of 5 Hz.

4.7.3 Biometric sensors and measurements

The biometric sensors used in this experiment included two sets of electromyography (EMG) electrodes and one set of electrocardiography (ECG) electrodes. The EMG is an established method of measuring muscle tension (Merletti and Parker, 2004) and was used to measure muscle tension in the subjects' Flexor Digitorum Superficialis muscle on the arm and the Platysma muscle on the neck. Electrical impulses appear during a muscle contraction and hereby generates a voltage difference between the electrodes. The more contraction, the more voltage difference. ECG is a proven tool for measuring the muscle functions of the heart (Yakut et al., 2014). Also the ECG measurements was acquired by using three surface electrodes, the hearts muscle generates an electrical signal periodically (Yakut et al., 2014), this is represented by an analogical signal in volts by the ECG.



(a) EMG **(b)** ECG

Figure 4.6: Electrode placement (Hacks, 2017)

Both the EMG and the ECG signals were gathered using an Arduino Uno with an e-health shield **Fig. 4.7** at a sampling rate of 380 Hz to provide sufficient resolution of the ECG data. The e-health shield provided a simple way of gathering physiological data with the Arduino platform. Besides EMG and ECG it also has the capabilities to do other measurements such as blood pressure, body temperature and skin conductivity.

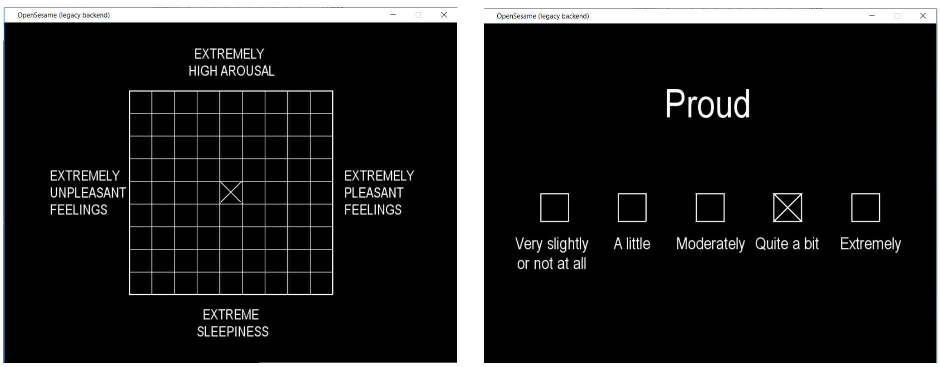
Figure 4.7: The e-Health Sensor Platform

4.8 User Interface

How the test subject is interacting with the tasks of an experiment is a vital part of every research that includes studying human behaviour. The interface created for this experiment was designed to facilitate easy interpretation of the necessary information, as well as focusing on consistency for all subjects. Furthermore, it was put effort to facilitate for a non biased response, as some of the main output used for data analysis was subjective (Affect grid, PANAS and AD-ACL). The interface was designed in a graphical experiment builder software, OpenSesame. This is a simple, free and open-source software for designing the graphical user interface (GUI) of experiments (Mathôt et al., 2012). OpenSesame provided tools for making the whole experiment, screen by screen (**Fig. 4.8**). **Appendix A** shows the GUI in its entirety. The subjects simply clicked their way through by using the controller. It also supports implementation of python scripts, which made it easy to integrate the Tetris code into the GUI. Even more convenient was the opportunity to code the affect grid(**Fig. 4.9a**) and the PANAS and AD-ACL (**Fig. 4.9b**) questionnaires in python as well, making the whole experiment on screen. This was done from scratch **Appendix B, C, D**. OpenSesame also provides data logging from scripts which allowed easy extraction of the subjects' answers, outputting all the data to a *.csv* file, ready for analysis. There were several other advantages of using this software. Firstly, it enabled complete separation between the subjects and the experimenters, which kept distractions and biases towards their evaluation of themselves to a minimum. It also provided continuity for the subjects, as well as allowing them to proceed in their own time without missing any information. Furthermore, the subjects were presented with the exact same information in the exact same order every test run. This not only provided consistency for the subjects, but also relieved the experimenters of the task of presenting all this information, freeing time to monitor and to keep an overview of the experiment.

(a) Information **(b)** Instructions

Figure 4.8: The GUI



(a) The affect grid, as presented to the subjects **(b)** One of 20 words in the PANAS questionnaire

Figure 4.9: Parts of the subjective evaluation screens

4.9 Procedure

This section presents the complete procedure for the experiment, including detailed explanation of the three scenarios (S1-S3) mentioned in the beginning of this chapter. It discusses how and why the various impulses were applied to achieve different levels of affect. As mentioned, the goal was to vary the affect of the subjects in three levels. These were as follows (**Table 4.4, Fig. 4.10**):

Scenario no.	Desired level of affect
S1	Low arousal, Positive valence
S2	High arousal, Negative valence
S3	High arousal, Positive valence

Table 4.4: The three desired levels of affect

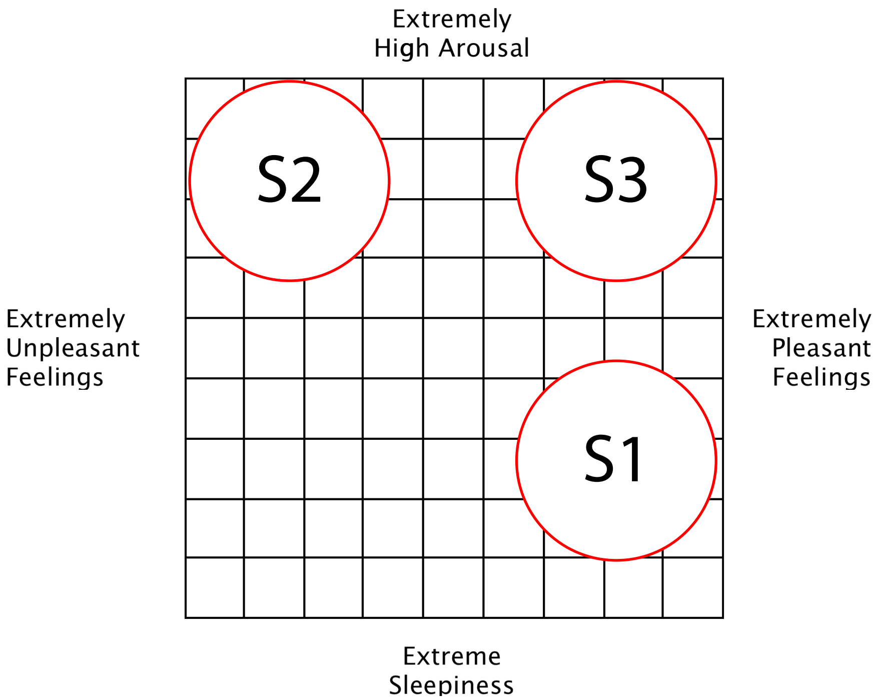


Figure 4.10: The desired placement of the three scenarios in the affect grid.

By achieving these three levels of affect this would have facilitatet both the authors' needs at the same time. The span in valence (S2 and S3) enables analysis according to this dimension and could at the same time confirm the change in physiological reactions also for low arousal (S1). The same goes for analysis of arousal, with span in this dimension (S1 and S3) and confirmation with negative valence (S2).

The scenarios were presented in the same order for every subject, with S1, considered as the "calmest" one, first. The reason for this was to avoid lingering effects from the high arousal scenarios (S2 and S3) into the low arousal scenario (S1), which was experienced during pilot testing. Another observation made in the pilot tests was an elevated arousal level in the beginning of the experiment. Many subjects seemed to get an increased arousal level by being wired up with electrodes and at all be excited, and even a bit nervous, to be a part of a research experiment. This confliced with the desired low arousal state, but the lingering effects of having S2 and S3 first was experienced to have a larger impact on arousal. Each scenario had a duration of 5 minutes, with approximately 2 minutes of questionnaires after each and a two minute break in between to ”reset” the emotional state. Including introduction and instructions the experiment had a total duration of about 30 minutes.

4.9.1 Initiation

The subject was welcomed to the experiment by reading and signing a standard consent form **Appendix J**. After this all the nine electrodes were placed on the subject's body. Three in the front upper body area for ECG, three on the right forearm and three on the neck for EMG **Fig. 4.11**. As both experimenters were males an image was presented to the female subjects, which showed the placement of the ECG electrodes. The correct placement of the electrodes was easily checked as real time sensor data was shown on the sampling computer. For the sake of simplicity the rest of the electrodes were placed by one of the experimenters. The subject was then given the headset and controller. Some information was conveyed personally, such as informing about the two information sheets linked to the affect grid **Appendix B**. After commencing the experiment the subjects were asked to report their initial state in an affect grid on screen. This was intended both for the subject to get to know the grid, but also to generate a starting point on how this person was feeling before starting.

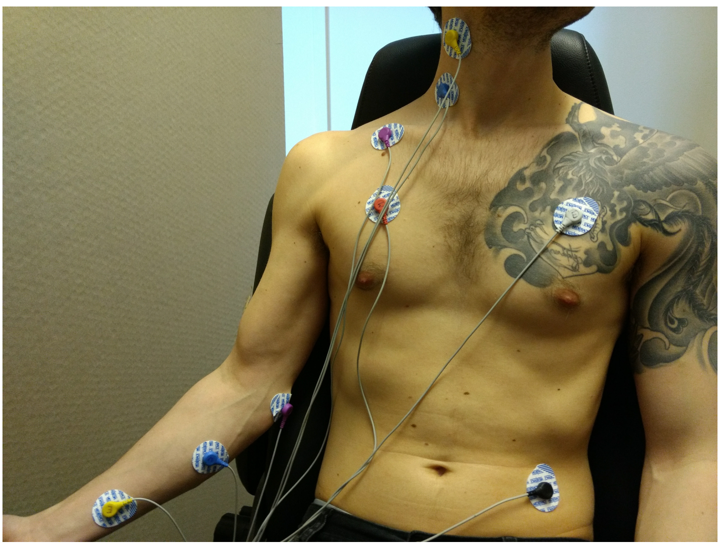


Figure 4.11: Connected sensor electrodes.

4.9.2 Scenario 1

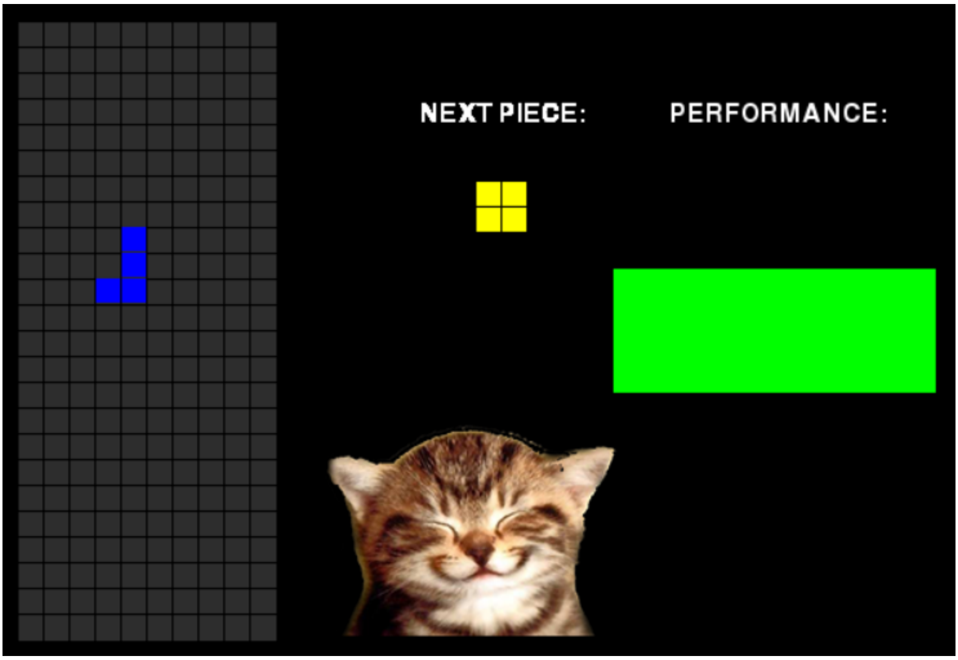


Figure 4.12: TETRIS, scenario 1.

Before starting the first scenario, instructions about the game layout and controls were given to the subjects. The layout of this situation included a pleasant picture of a cat. Cats are well known to be cute creatures and pilot testing indicated that the cat evoked pleasant feelings. In this scenario the layout did not include a score or a counter of how many lines the subjects had accomplished in the game, as the experience was intended to be as calm and as little competitive as possible. A performance bar was placed to the right of the game, actively giving the subjects feedback on their performance. The stimuli used in this scenario were as follows:

Difficulty

The speed of the game was set to a relative low setting, pilot testing showed that this was comfortable for the subjects, and far from being a challenging factor. To achieve the desired relaxed emotional state, the three pieces most of the pilot subjects found challenging were removed, this left only four pieces which gave the game a pleasant and calming effect. The game was too simple and unchallenging for the subjects to get aroused, yet it was fulfilling for the subjects to perform well.

Lights

The lights was set to a steady setting with a warm color and low intensity. The light was not supposed to stimulate on its own, it was a complimentary feature to make the visual experience from the screen more pleasant and less intense.

Music

The music the subjects were presented with in this scenario was slow paced and calming.

Feedback

In this scenario the subjects were only presented with positive feedback. Every time they cleared a line a positive sound was played, and the performance bar only moved in the positive direction, upwards. On the positive side, the bar had a green color, often associated with correct actions. This was done to keep the subjects from getting bored and unfulfilled.

4.9.3 Scenario 2

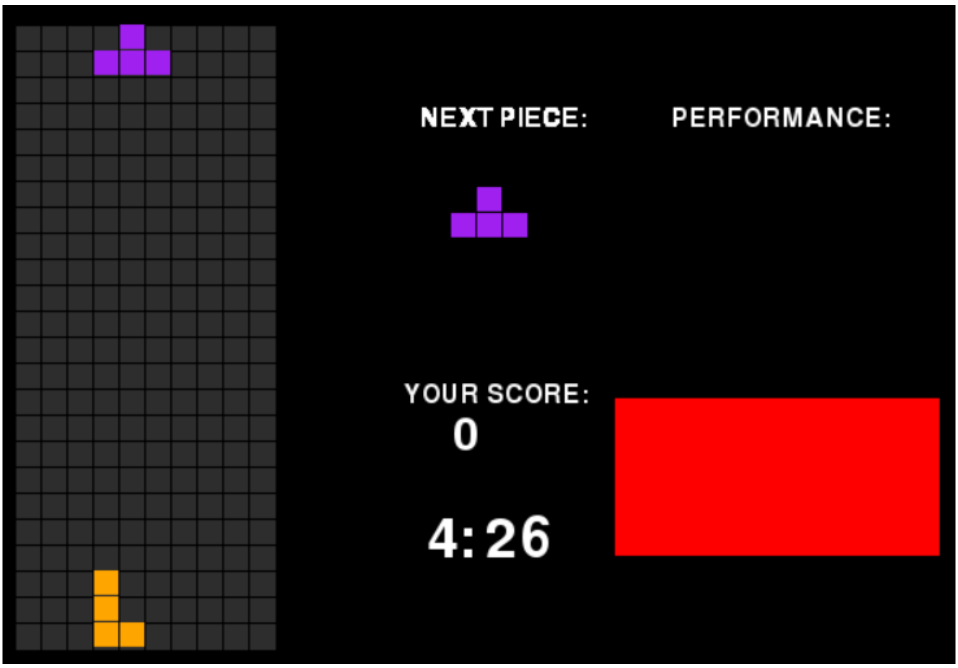


Figure 4.13: TETRIS, scenario 2.

Also before starting the second scenario instructions were given. The layout in this scenario did not include a picture of a cat, but here the score was presented in addition to the time remaining. Both the score and the remaining time were included to evoke the feeling of needing to perform, resulting in the desired arousal level. In addition, the subjects were given information stating that this scenario was a part of a competition, utilizing people's competitive spirit to achieve even higher arousal.

Difficulty

The speed was by default set at a really high level, and most pilot subjects found it to be too fast to be able to perform well. The speed also increased after 3 minutes to a level that even high performing Tetris players would find too high. The input controls were inverted so move right became move left and vice versa. The controls were switched back to normal after two and a half minutes, and then back to inverted after four minutes. This lead to a lot of frustration because when finally the controls were familiarized they were switched again. The pieces in this scenario included all seven pieces, however the order was manipulated in a way so that the three pieces the pilot subjects found most challenging appeared at a higher rate, and the most helpful piece at a much lower rate.

Lights

The lights in this scenario were strobing fast through four colors similar to police car flashers, red, blue, green and white. The intensity was set to the highest possible setting to make it as unpleasant and annoying as possible. This stimuli was kept in spite of one of the pilot subjects feeling it was over the top. Along with the difficulty, the lights appeared to be the most prominent traction stimuli for negative valence in this scenario. To clarify, the subjects were asked if they had any kind of epilepsy in the consent form before participation.

Music

Several kinds of music were subject to pilot tests for this scenario. Eventually a song was chosen based on feedback from pilot subjects. This song was ”up-beat”, powerful and ominous, similar to the music in a horror movie when the action level is on it's highest.

Feedback

In this scenario the subjects were not presented with any positive feedback, but given negative feedback both by an intense and negative buzz sound and the performance bar moving in the negative direction (downwards) when performing bad. On the negative side the bar had a red color, often associated with incorrect actions. Because of the scenario having a quite high level of difficulty, these negative feedback aspects were presented with high frequency, giving the subjects an even more negative experience.

4.9.4 Scenario 3

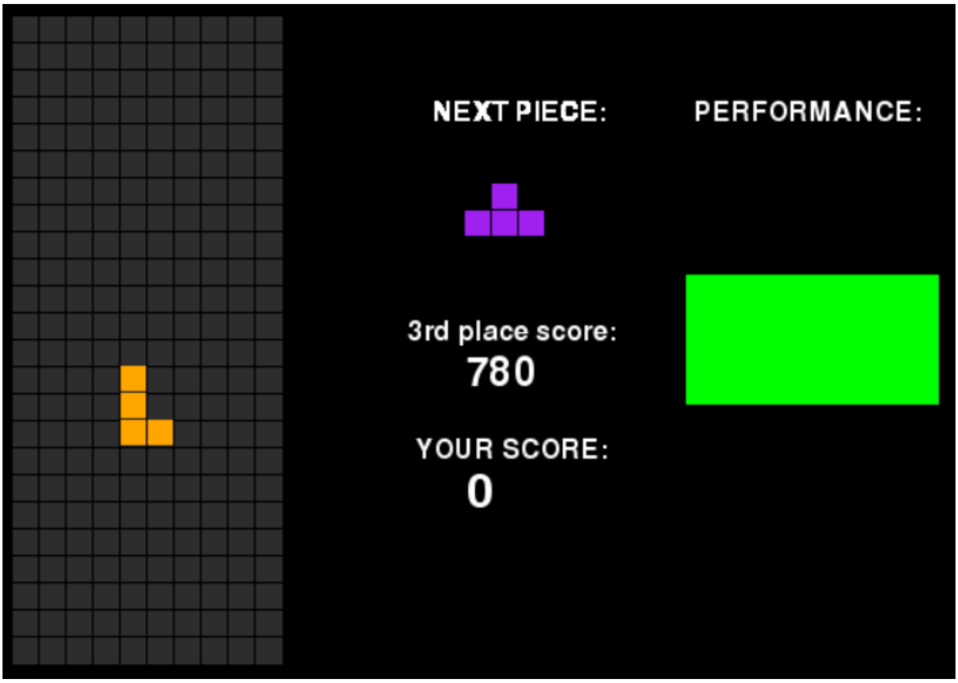


Figure 4.14: TETRIS, scenario 3.

Before starting the third scenario the subjects were again presented with the instructions and layout of the scenario. This time they were presented with the instruction about this scenario being part of a competition two times to make an even call to the competitive spirit to increase arousal levels. In this scenario the layout included a highscore list, starting at third place, this is more explained in the feedback description below.

Difficulty

The game speed wass set relatively high, but still achievable. The thought was to let the subjects make it, but only barely, to achieve a maximum positive valence and high-arousal effect. By giving the subjects a good fight the intention was for them to achieve an even greater feeling of accomplishment when they made it.

Lights

The lights in this scenario were intended to increase arousal without compromising positive valence. Though the flashing lights in S2 were perceived as very negative due to the fast flashes, flashing colors in a slower pace seemed to avoid getting negative valence. Quite the opposite, this switching of colors seemed to only reinforce positive feelings as the scenario in itself was positive. The intervals between each color was set to 961 ms to accompany the beat of the music.

Music

Several songs were tested for this scenario before the final version of the experiment, and various up-beat instrumental songs were perceived as only moderately arousing. What seemed to do the trick was the association people had with the song, which could often push them towards both higher arousal and more positive valence at the same time. This is possibly a bit risky, as associations could differ greatly from person to person. To minimize this risk, a song that lies in the hearts of many 90s children's memories is chosen, as the majority of the participants in the experiment was born in the early 90s. The song is called "Sandstorm" and is performed by Darude.

Feedback

The feedback of this scenario was exclusively positive. The bar was used in the same way as in S1, going up for each line taken, whilst playing a positive "ping" sound. In addition to this a highscore list was shown. As in **Fig. 4.14**, the game started by showing the 3rd place score. When the subject passed this score, the screen turned black with the text *"YOU ARE NOW IN 3RD PLACE!"* flashing in blue and green in the center, and a corresponding "level up" sound was played. The score on the right changed to show the 2nd place score, and further to 1st place score, also with a flashing text and sound in between. When the 1st place score was beaten, the text *"YOU ARE NOW IN THE LEAD!"* was shown. After recommencing the game, this text was also shown to the right instead of "x place score". The indications of increasing positioning on the highscore list evolved from being just the subtle change in the text to the right to a more pretentious and explicit notification, as subjects were often too deep into the game to notice this change.

Chapter 5

Results

From the experiment a lot of data was collected. In this chapter the data is presented and analyzed. The data collected involves a lot of variables, and only the most important ones are discussed. The participants are in some cases split in three groups, depending on the situation the data was collected from, in other cases the participants are split in two groups, high arousal and low arousal, based on the recorded data. Total number of participants in the experiment was 34, but since the experiment involved three scenarios, the data is handled as it was 102 independent data points. Efforts was made to minimize order- and learning effects in the experiment setup, and they are hence excluded from the following results. The Hawthorne effect was to some degree observed by the participants high heart rate before the experiment was started, because of its' complex nature, it is also excluded from the following results. Seven of the participants was female, and 27 was male. Three participants has a master degree, four a bachelor degree, and 27 has a high schoool diploma as their highest achieved education. A total of 19 was between 20 - 25 years of age, and 15 between 26 - 30. To extract the HRV data from the raw ECG data, Biomedical Workbench^{TM}, a software from National Instruments, was used. To analyze the data, SPSS Statistics 24^{TM}, a software from IBM, was used.

5.1 Preparing the data

In this section the methods used to prepare the data for further analyzes are presented.

5.1.1 Sorting the biometric data

The raw data form the biometric sensors (EMG and ECG) are in a continuous form for every participant. This means that there is only a manually coded signal for every scenario start and stop. The data is hence split in to the three scenarios, and the brakes are deleted. This gives the data in milliseconds, ECG-value and EMG-value for each scenario. One might argue that the data from the brakes might be interesting, but since there is only recorded subjective data for the discrete scenarios, the biometric data for the brakes are disregarded.

5.1.2 HRV extractions

To extract the HRV features from the ECG data, a software pack called Biomedical Workbench was used. The raw data was converted in to a usable format with a file converter included in the package, in this process the sampling rate was also put in to the data. Then a ECG feature extractor was used to extract the QRS complex from the ECG data **fig. 5.1**. Lastly the annotated file is opened in Heart Rate Variability Analyzer **fig. 5.2** where the HRV statistics are presented along with power spectrum from both a Fast Fourier Transform and an Auto Regressive AR model. The spectrum is divided in to three discrete components, very low frequency VLF(0,00-0,04 Hz), low frequency LF(0,04-0,15 Hz), and high frequency HF(0,15-0,40 Hz). To limit the data size, only the FFT power spectrum is used further.

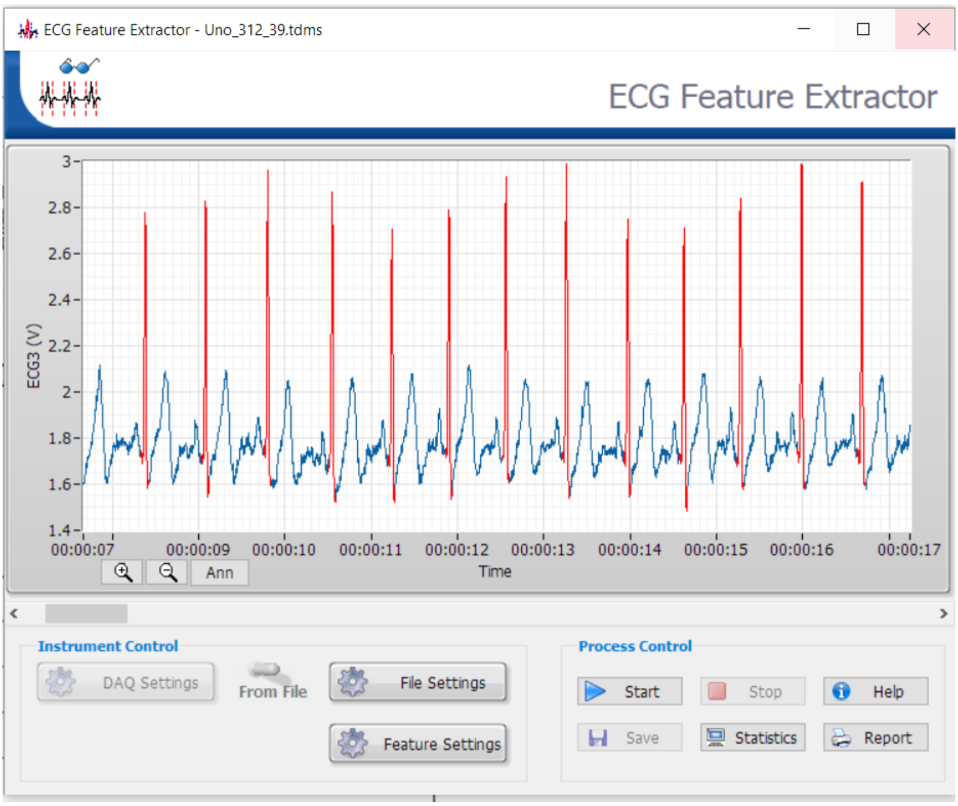


Figure 5.1: The QRS complex marked by the ECG Feature Extractor

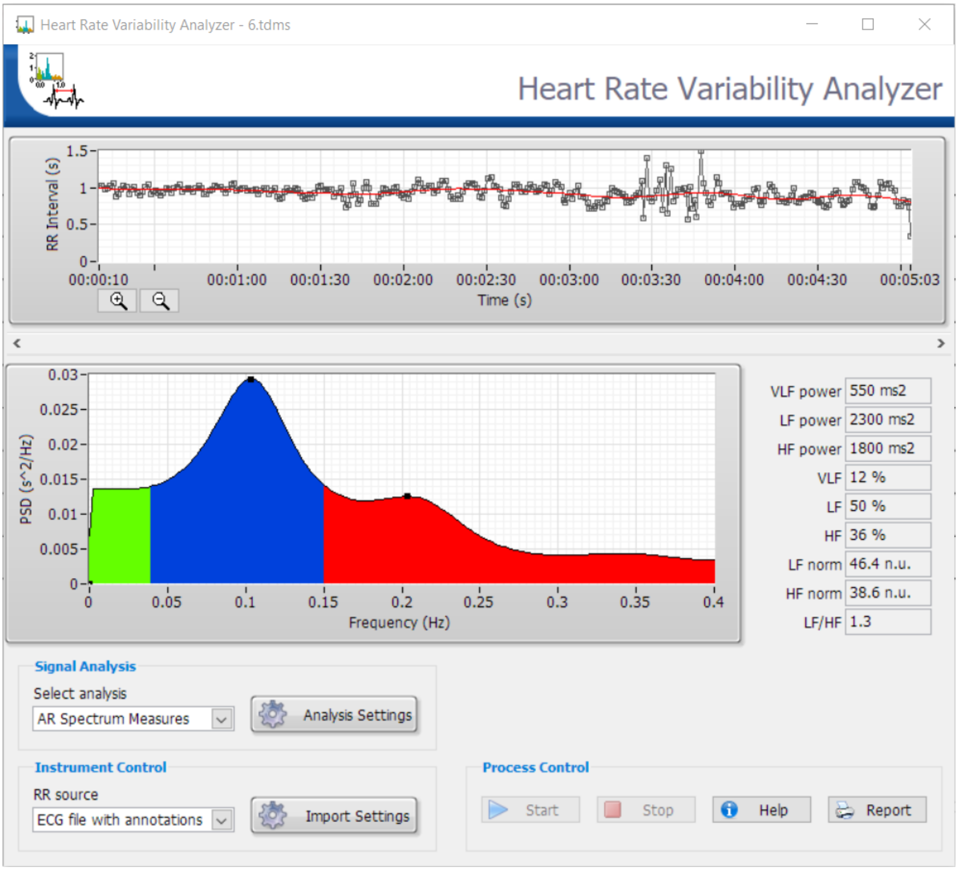


Figure 5.2: HRV data preesented in the Heart Rate Variability Analyzer

5.1.3 Machine learning

The pressure sensor data from the chair are re-coded in to two different variables, Number of position changes and level of leaning. To do this a classification algorithm is used. In this case a k-dimensional tree algorythm was used. A k-dimensional tree is a binary tree were every node is a k-dimensional point Ramasubramanian and Paliwal (1992) Hunt et al. (2006). It is an established data structure for organizing points in k-dimensional spaceZhou et al. (2008)Shevtsov et al. (2007). In this case there are 14 dimensions, one for each pressure sensor.

The classifying is done in two scripts, one for each output variable. The classifiers are trained with two sets of training data, one for position changes and one for leaning. The training data for the position changes consists of seven recorded positions:

1. Straigh with head on headrest
2. Straight without head on headrest

3. Leaning forward
4. ”Slouching” with head on headrest
5. ”Slouching” without head on headrest
6. Leaning to the right
7. leaning to the left

The training data for the leaning consists of 5 different levels of leaning forward, from the equivalent to the first postion from the position set, to all the way forward with no back contact to the chair at all.

The position data was further processed by splitting the scenarios, and deleting the brakes, and then the number of changes was counted for each scenario(case). Similar with leaning, the data was split, and then the mean of the levels of leaning was calculated. However the leaning data did not turn out dependable, this might be caused by over-fitting or the training data was not well enough recorded.

5.1.4 Grouping the cases

As it turned out, the self reported arousal from each scenario did not turn out to be as divergent as desired **Table 5.2**.

	Group	N	Mean	Std. Deviation	Std. Error Mean
pNN50	Scenario 1	32	15,6288	16,068279	2,94913
	Scenario 3	33	13,7406	14,14856	2,46295

Table 5.1: Group statistics of pNN50, with scenarios as groups

				t-test for Equality of Mean		95 % Conf.Interval of the Difference	
	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Higher
pNN50	0,493	63	0,624	1,88814	3,383253	-5,77056	9,54685

Table 5.2: Independent samples test of pNN, with scenarios as groups

As one can see there is no significant difference in pNN50 between the two groups. pNN50 is an established objective significant measure of arousal (Taelman et al., 2009) and the missing difference in the distribution between the two groups can hence be seen as an indicator that there is not significant difference in arousal. This is an indication that the stimuli in the different scenarios did not work as intended. To further be able to examine the effect arousal has on posture, the cases are grouped by the scores from the AD ACL questionnaire. To properly examine the relation the arousal has on posture, the cases that reported high or low arousal was chosen, while the cases that reported medium arousal was ignored. One might say that by doing so, the complete relation is not shown, but tendencies are amplified and more detectable. The cases that reported a score higher than 13 was put in the high arousal group, and cases that reported a score below 5 was put in the low arousal group. Following are the results of an independent T-tests run on the participants pNN50 values based on the new groups.

	Group	N	Mean	Std. Deviation	Std. Error Mean
pNN50	Low Arousal	33	19,4824	16,86438	2,93571
	High Arousal	32	11,4116	14,33361	2,53385

Table 5.3: Group statistics of pNN50, with self reported arousal as groups

				t-test for Equality of Mean		95 % Conf.Interval of the Difference	
	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Higher
pNN50	2,076	63	0,042	8,07086	3,88777	0,30178	15,83995

Table 5.4: Independent samples test of pNN, with self reported arousal as groups

Now there is a significant difference between the groups.

5.2 Descriptive Results

In this section descriptive results from the experiment is described. The descriptive results are sectioned according to the dependent variables of arousal listed in **section4.3**. The dependent variables related to valence is not discussed in this thesis. The distributions are presented in box plots, and tables containing number of cases, means, medians, standard deviations, and minimum and maximum values are presented for later analysis purposes.

5.2.1 Arousal - Objective Measurements

Following are the descriptives of heart rate and heart rate variability. Included in the descriptives statistics **tab. 5.5** and the presented distributions **fig. 5.3** are heart rate, the time based HRV feature pNN50, and three frequency based HRV features. The VLF feature from the power spectrum is ignored because of the short sampling time of 5 minutes.

Table 5.5: Descriptive statistics of heart rate and heart rate variability

		Heart Rate Mean	pNN50	LF Power	HF Power	LF/HF
Low Arousal	N	33	33	33	33	33
	Mean	76.33	19.4824	655.91	413.233	2.7939
	Median	76.00	14.0000	620.00	287.000	2.2000
	Std. Deviation	12.457	16.86438	469.717	530.7924	1.9254
	Minimum	55	.00	82	11.1	.3800
	Maxium	110	57.00	1900	2670.0	7.4000
High Arousal	N	32	32	32	32	32
	Mean	80.09	11.4116	544.69	209.400	3.6475
	Median	82.00	6.5500	320.00	130.500	2.9000
	Std. Deviation	10.593	14.33361	516.471	250.7182	2.3973
	Minimum	56	.00	100	19.7	.4700
	Maximum	100	46.00	2000	1350.0	9.2000

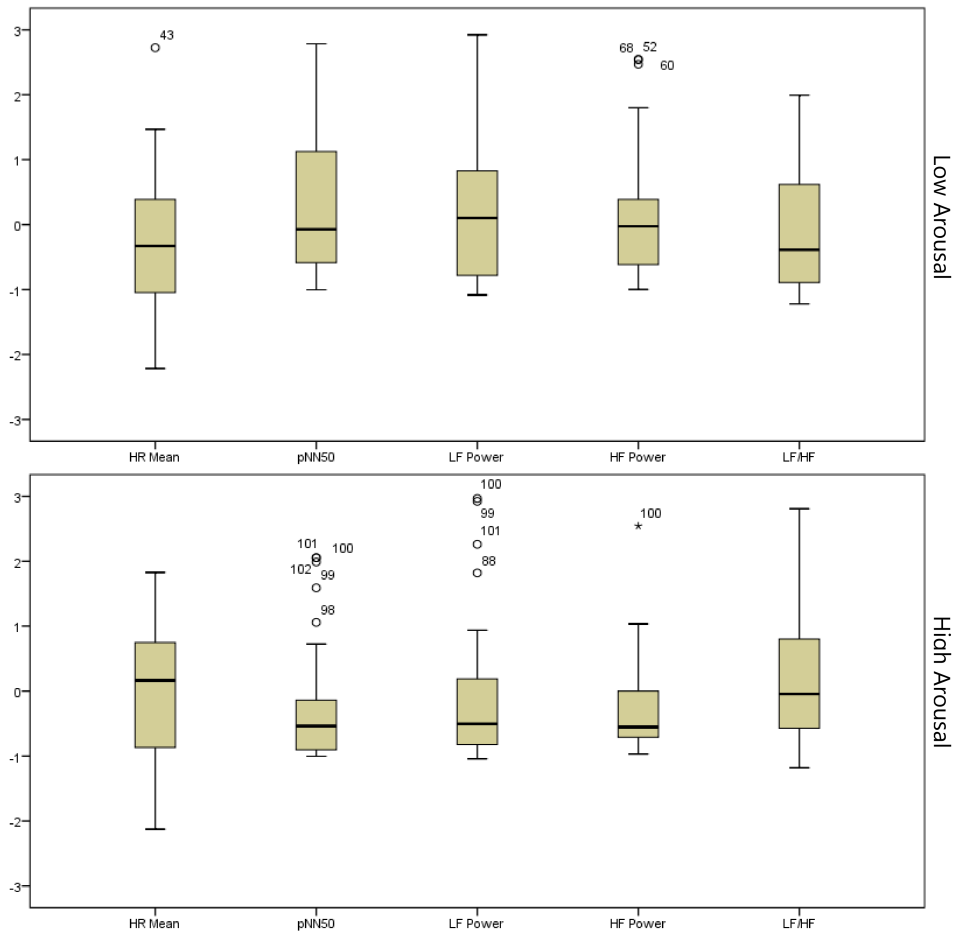


Figure 5.3: Distribution of heart rate and heart rate variability in the two groups. Mild outliers are marked with circles, extreme outliers are marked with asterisks

5.2.2 Arousal - Subjective Measurements

To investigate how the posture and position relates to arousal, two established subjective measures are included as countermeasures below. The first is the Activation-Deactivation Adjectvive Check List (AD-ACL) questionnaire, and the second is the y-score from the Russel Affect Grid

AD-ACL

The 20 words from the AD-ACL are divided in to four sub-dimensions(Thayer, 1990). The four sub-dimension are ”Energetic” A1, ”Tired” A2, ”Tension” B1, and ”Calmness” B2. Each sub-dimension are contrived from the associated words where each word are scored on a Likert scale with a range of 1-4. This gives each sub-dimension a minimum score of 5 and a maximum score of 20. As the sub-dimension's names suggest, a high score on A1 and B1 indicate high arousal, while a high score on A2 and B2 indicate low arousal. For a more complete picture a total AD ACL-score is calculated by adding the A1- and B1-scores and then subtracting the scores from A2 and B2. This gives a AD-ACL-score ranging from -30 to 30, where a score of -30 equals the lowest possible arousal, and 30 represents the highest possible arousal. The descriptive statistics of the scores are presented in **tab. 5.6** and the distribution of the calculated scores are presented in **fig. 5.4**

Table 5.6: Descriptive statistics of AD-ACL scores

		A1	A2	B1	B2	AD-ACL-score
Low Arousal	N	33	33	33	33	33
	Mean	11.39	11.42	8.36	14.64	-6.30
	Median	12.00	12.00	8.00	15.00	-7.00
	Std. Deviation	2.499	3.783	2.644	2.737	7.951
	Minimum	5	7	5	8	-24
	Maxium	16	19	15	19	4
High Arousal	N	34	34	34	34	34
	Mean	16.15	6.06	14.97	7.15	17.91
	Median	16.00	6.00	15.00	7.00	17.00
	Std. Deviation	1.708	1.229	2.289	2.047	3.379
	Minimum	12	5	11	5	14
	Maximum	20	11	19	15	29

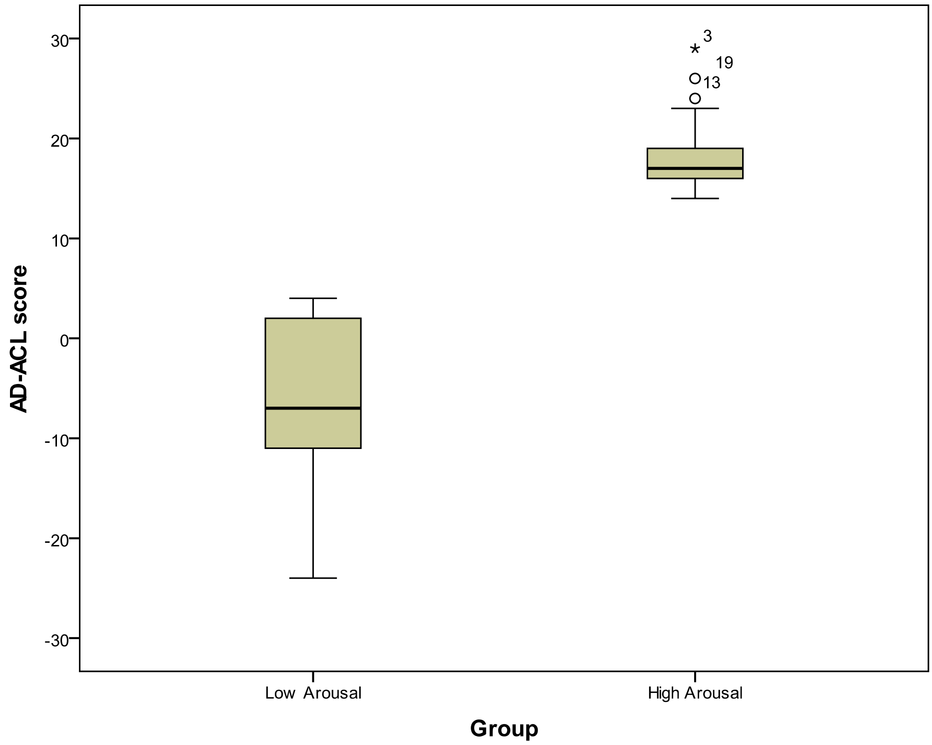


Figure 5.4: Distribution of AD-ACL scores in the two groups. Mild outliers are marked with circles, extreme outliers are marked with asterisks

Affect Grid

The descriptive statistics of the y-scores from the Russel Affect Grid are presented in **tab. 5.7** and the mean distribution are presented in **fig. 5.5**

	Low Arousal	High Arousal	Total
N	33	34	67
Mean	5,52	7,76	6,66
Median	6	8	7
Std. Deviation	1,302	0,819	1,562
Minimum	2	6	2
Maximum	7	9	9

Table 5.7: Descriptive statistics of grid y-scores

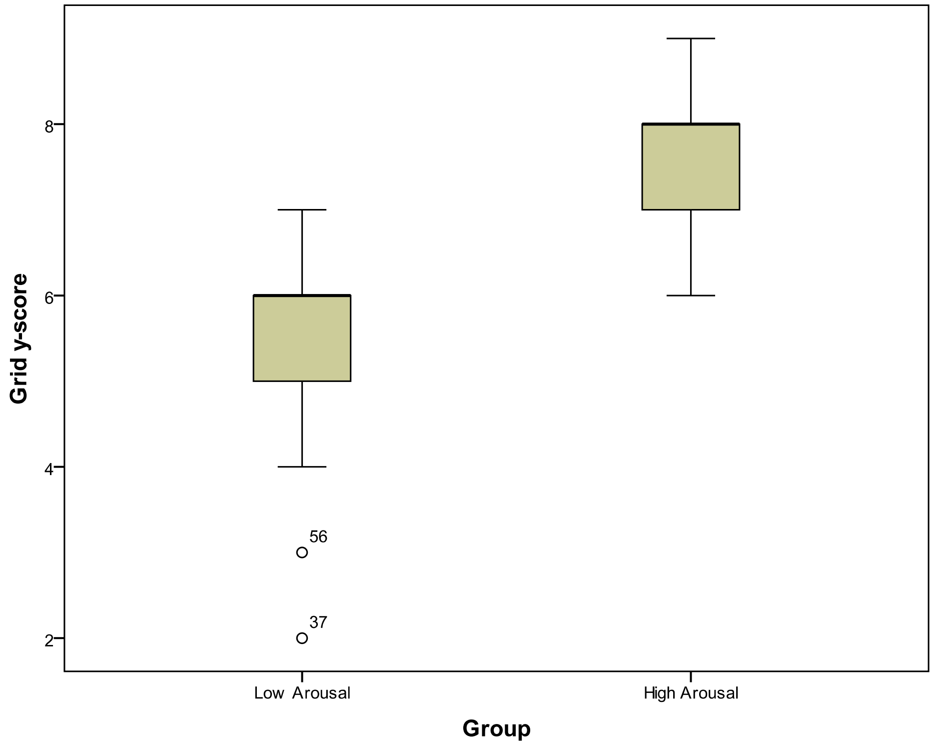


Figure 5.5: Distribution of grid y-scores in the two groups. Mild outliers are marked with circles

5.2.3 Number of changes in sitting position

The raw data from the chair was as previously mentioned classified by an algorithm to determine the position the participant was using. The number of position changes during the five minute period of each scenario was then counted and used as a new variable. **Tab. 5.8** shows the descriptive statistics from the number of position changes, and **fig. 5.6** shows the distribution in both groups.

	Low Arousal	High Arousal	Total
N	33	34	67
Mean	14,36	45,15	29,99
Median	3,00	31,00	7,00
Std. Deviation	27,521	48,107	42,010
Minimum	1	1	1
Maximum	113	187	187

Table 5.8: Descriptive statistics of number of changes in sitting position

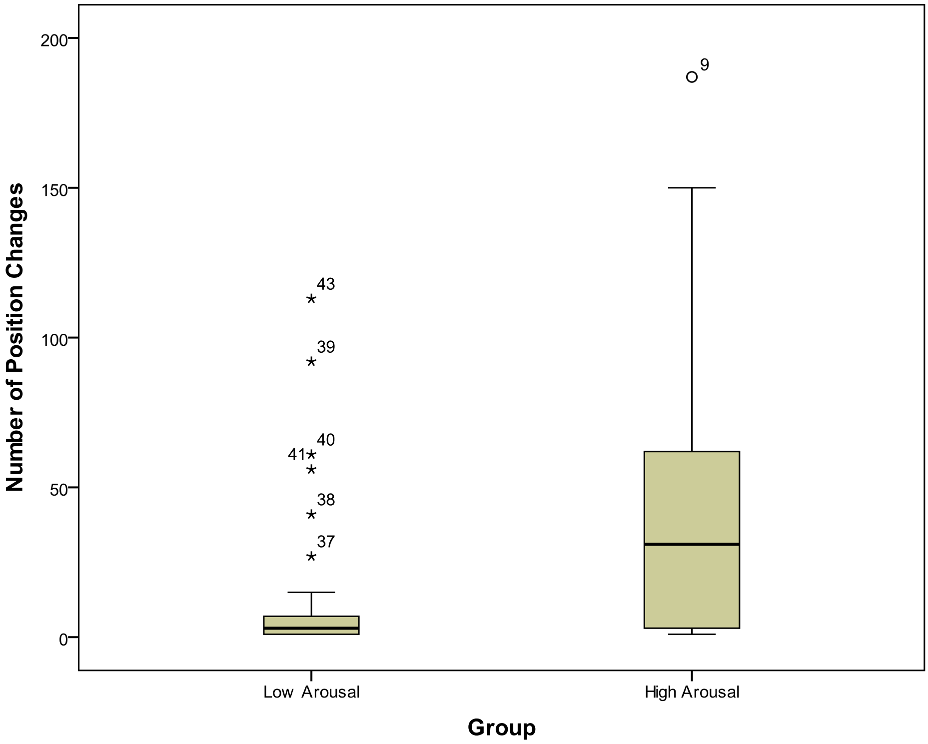


Figure 5.6: Distribution of number of changes in sitting position in the two groups. Mild outliers are marked with circles, extreme ouliers are marked with asterisks

5.2.4 Leaning

The leaning is obtained by two measures, one from the Lidar distance sensor and one from the classification script run on the raw chair data. The descriptive statistics and the distribution are presented for both variables. **Tab. 5.9 and fig. 5.7** presents information about the classified variable, while **Tab. 5.10 and fig. 5.8** represents the Lidar data.

Classified mean of leaning

	Low Arousal	High Arousal	Total
N	33	34	67
Mean	2,8718	2,7957	2,8332
Median	2,9050	2,9399	2,9050
Std. Deviation	0,70028	0,77167	0,73278
Minimum	1,94	1,09	1,09
Maximum	4	4	4

Table 5.9: Descriptive statistics of leaning

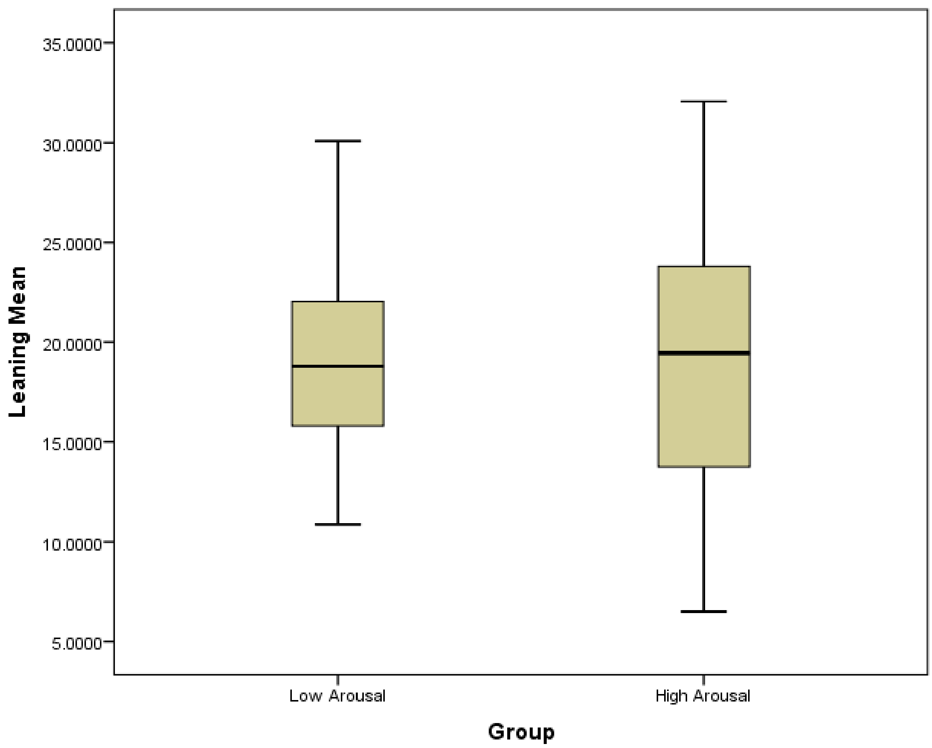


Figure 5.7: Distrubution of leaning in the two groups, based on chair data.

Lidar

	Low Arousal	High Arousal	Total
N	33	34	67
Mean	19,113568	19,532324	19,326071
Median	18,791734	19,451644	19,026899
Std. Deviation	4,1897754	6,2514866	5,3005822
Minimum	10,8638	6,5006	6,5006
Maximum	30,0928	32,0565	32,0565

Table 5.10: Descriptive statistics of leaning

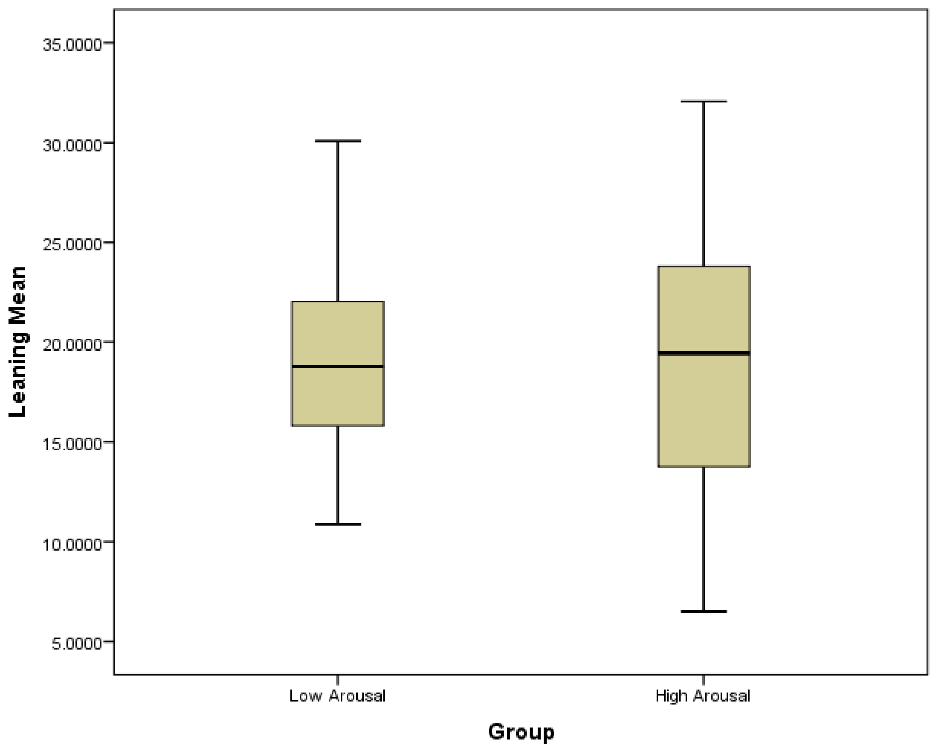


Figure 5.8: Distribution of leaning in the two groups, based on the distance sensor. Mild outliers are marked with circles

5.3 Statistical Evaluation

In this section the data presented previously are evaluated statistically. First the established measures are evaluated and tested internally. Then the new suggested measures are evaluated and related to the hypotheses. As mentioned earlier the data from each scenario of each participant is treated as one independent case (data point).

When analyzing the established measures, the independent samples t-test is used on HRV variables, since they are continuous, and on the obcjective measures the Mann-Whitney U-test is used.

5.3.1 Evaluating the established measures of arousal

HRV

To do an independent t test on the HRV variables three main criteria must be satisfied. Two of which, having a continuous dependent variable (in this case the HRV variables) and an independent variable with two levels or categories (in this case the arousal groups), are satisfied. But the third criteria is to have independent cases, which in this case is questionable since the cases are derived from participants doing three scenarios, but with each scenario from each participant being treated as one independent case. This can be seen as a critique of the experiment design. However since the learning and order effects have been neglected, the assumption that the cases are independent are maintained. The result of the independent samples t-test are presented in **tab. 5.11**. The first two columns represents the Lavene's test for equality of variances that checks if there is a significant difference in the variance between the two groups. However the t-test is robust and includes a test even if the variances have significant difference(the second line of each t-test).

Table 5.11: Independent Samples t-test of HRV variables

		Lavene's test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Heart Rate Mean	Equal variances assumed	.448	.506	-1.309	63	.195	-3.760	2.872	-9.500	1.979
	Equal variances not assumed			-1.312	61.957	.194	-3.760	2.865	-9.488	1.967
pNN50	Equal variances assumed	2.618	.111	2.076	63	.042	8.07086	3.88777	.30178	15.83995
	Equal variances not assumed			2.081	61.948	.042	8.07086	3.87799	.31875	15.82298
LF power	Equal variances assumed	.008	.927	.909	63	.367	111.222	122.381	-133.338	355.781
	Equal variances not assumed			.907	62.020	.368	111.222	122.563	-133.776	356.219
HF Power	Equal variances assumed	4.230	.044	1.969	63	.053	203.8333	103.5013	-2.9975	410.6642
	Equal variances not assumed			1.989	45.911	.053	203.8333	102.4791	-2.4571	410.1238
LF/HF	Equal variances assumed	1.521	.222	-1.585	63	.118	-.85356	.53850	-1.92966	.22254
	Equal variances not assumed			-1.580	59.398	.119	-.85356	.54032	-1.93458	.22746

AD-ACL

A visual inspection of the distribution of both groups for all the sub-dimensions and for the total score indicates similarity in distribution, which is an assumption for the Mann Whitney test. Hence the Mann Whitney test was run to determine if there were differences in self reported AD ACL-scores, and the results can be seen in **tab. 5.12**

Table 5.12: Mann-Whitney U test of AD-ACL

	A1	A2	B1	B2	AD-ACL-score
Total N	67	67	67	67	67
Mann-Whutney U	1078.000	61.000	1076.500	30.000	1122.000
Wilcoxon W	1673.000	656.000	1671.500	625.000	1717.000
Test Statistic	1078.000	61.000	1076.500	30.000	1122.000
Standard Error	79.245	78579	79.323	79.391	79.619
Standarized Test Statistic	6.524	-6.363	6.499	-6.688	7.046
Asymptotic Sig. (2-sided test)	.000	.000	.000	.000	.000

The significant differences are highlighted in yellow. It looks a little suspicious that all the sub-dimensions and the total score yields significant median differences, but the groups were set by cases that reported high or low AD ACL scores, which means that the degree of freedom from the AD-ACL questionnaire are already been used, and these results can not be used further in the evaluation.

Affect Grid

Also the distrubution of grid y-scores for both groups also indicates similarity by visual inspection, and hence is the assumption for the Mann-Whitney U test satisfied. The results of the following test is shown in **tab. 5.13**

Table 5.13: Mann-Whitney U test of y-scores from the affect grid

	Grid y-score
Total N	67
Mann-Whutney U	1056.000
Wilcoxon W	1651.000
Test Statistic	1056.000
Standard Error	77.822
Standarized Test Statistic	6.361
Asymptotic Sig. (2-sided test)	.000

Also here the difference in medians between the two groups are significant as seen by the highlighted significance. But the y-scores from the affect grid is also a subjective measurement of arousal and it is expected that the median difference between the cases that reported high and low arousal according to the AD ACL questionnaire also reported high and low arousal in the affect grid

Correlations

Here the correlations between the HRV measures and the two subjective measures are presented. The Spearman correlation test is used for this. This is because it is either a test of association between two ordinal variables or between one ordinal and one continuous variable. The correlation matrix is presented in **Appendix L** The correlation between the different HRV features are included in the table, but marked in grey because they are dependent on each other and also all continuous, which means Pearson correlation should have been used. Also the correlation between the sub-dimensions of AD-ACL and the AD ACL-score are marked in grey because they are dependent on each other. Some of the cells are marked in a light grey color, this is due to the values on the left side ,of the diagonal from the upper left corner to the lower right corner, are duplicates of the values on the upper right side of the diagonal.

Clearly the counter measures and the objective measure are not correlating that well, but there are some trends visible for the time domain feature pNN50, which as mentioned before have some identified ties to arousal. The more comprehensive frequency domain features does not have any significant correlation with the subjective measures, except for the HF power, which show significant correlation to the B2 sub-dimension, and some weak correlation all over. It is suspected that the lack of correlation for the LF power is due to the relatively short term observations of 5 minutes, the LF frequency have an absolute minimum observation term of 4 minutes and 10 seconds. The subjective measures all have highly significant high correlation within themselves, but this was also excepted.

5.3.2 Testing the arousal hypotheses

Following are some statistical tests on the measures of leaning and the position changes measure. Since all of the measures are continuous the independent samples t-test is used to see if there is a significant difference in the distribution in the two groups **tab. 5.14**.

Table 5.14: t-test on leaning and position changes measures

		Levene's test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Leaning Lidar	Equal variances assumed	4.830	.032	-.321	65	.749	-.4187565	1.3041745	-3.0233728	2.1858597
	Equal variances not assumed			-.323	57.837	.748	-.4187565	1.2966841	-3.0145072	2.1769942
Leaning Chair	Equal variances assumed	.464	.498	.422	65	.675	.07602	.18019	-.28386	.43589
	Equal variances not assumed			.422	64.713	.674	.07602	.17993	-.28336	.43539
Number of Position Changes	Equal variances assumed	9.939	.002	-3.202	65	.002	-30.783	9.614	-49.984	-11.583
	Equal variances not assumed			-3.227	52.814	.002	-30.783	9.540	-49.921	-11.646

In **tab. 5.14** the only measure with a significant difference in distribution is number of position changes, this indicates that the number of position changes might have some association with arousal, but it also indicates that the leaning might not have any association. To further test the association, the three measures are tested for correlation with the HRV measures and the subjective countermeasures. Pearson correlation test is used to test the measures with the HRV features **tab. 5.15**, this is because all the variables are continuous, while Spearman correlation test is used to test the correlation with the subjective countermeasures **tab. 5.16**.

Table 5.15: Correlation between the posture measures and the HRV features

		Number of Position Changes	Leaning Chair	Leaning Lidar	Heart Rate Mean	pNN50	LF power	HF power	LF/HF
Number of Position Changes	Pearson Correlation	1	-.107	.003	.318	-.235	-.027	-.124	.150
	Sig. (2-tailed)		.387	.983	.010	.059	.829	.324	.234
	N	67	67	67	65	65	65	65	65
Leaning Chair	Pearson Correlation	-.107	1	-.407	.088	-.261	-.283	-.164	.120
	Sig. (2-tailed)	.387		.001	.488	.036	.022	.191	.341
	N	67	67	67	65	65	65	65	65
Leaning Lidar	Pearson Correlation	.003	-.407	1	-.208	.336	.257	.094	.082
	Sig. (2-tailed)	.983	.001		.096	.006	.039	.457	.516
	N	67	67	67	65	65	65	65	65

Table 5.16: Correlation between the posture measures and the subjective measures

		Number of Position Changes	Leaning Chair	Leaning Lidar	A1	A2	B1	B2	AD-ACL Total	Grid y-score
Number of Position Changes	Correlation Coefficient	1.000	-.112	.032	.363	-.461	.276	-.464	.465	.433
	Sig. (2-tailed)	.	.365	.795	.003	.000	.024	.000	.000	.000
	N	67	67	67	67	67	67	67	67	67
Leaning Chair	Correlation Coefficient	-.112	1.000	-.391	.040	.153	.054	.074	-.057	.005
	Sig. (2-tailed)	.365	.	.001	.745	.216	.667	.552	.644	.966
	N	67	67	67	67	67	67	67	67	67
Leaning Lidar	Correlation Coefficient	.032	-.391	1.000	.118	-.190	.043	.021	.071	-.026
	Sig. (2-tailed)	.795	.001	.	.340	.124	.728	.866	.570	.833
	N	67	67	67	67	67	67	67	67	67

5.4 Hypothesis Evaluation

The previously presented statistical tests was conducted to evaluate the arousal hypotheses from 4.1. These two arousal hypotheses relates to different aspects of posture recorded. The evaluation of H3 and H4 are presented below. Both hypothesis are evaluated both subjectively and objectively. First the difference between the groups derived from subjective self report are evaluated. The correlations between the measure in question and the subjective and objective measures are then evaluated.

5.4.1 Position changes hypothesis H3

From the t-test it is clear that the difference between the two groups are highly significant, with a significance level of $p = .002$, and $t(52.814) = -3.227$. Further does the position changes measure have a correlation between $r = .267$ and $r = .464$ with a significance level of maximum $p = .024$ with the sub-dimensions of AD ACL, where the strongest correlations are with the dimensions of tiredness (A2) and calmness (B2), these two both have a negative correlation, which means the more tiredness and or calmness the subjects reported they felt, the less position changes were recorded. A2 and B2 also have the highest correlation $r = -.461$ and $r = -.464$, with significance levels of $p < .01$. Also with the total AD ACL score and the grid y-score, the correlations are relative high with $r = .465$ and $r = .433$ with significance levels of $p < .01$. Moving on to the correlation with the objective HRV measures. The only significant correlation position changes has with the features from the HRV analysis is a correlation of $r(63) = .318$ with a significance level of $p = .01$ with heart rate. the correlation with pNN50 are $r(63) = -.235$ $p = .059$, which is close to be significantly relevant.

As presented above there are some statistical evidence that the number of position changes have an association with arousal, more specific that the number of position changes increases with increased arousal. The evidence is at least strong enough for further research to be done on this feature of posture. Most interesting are the decrease in position changes connected with the increase in the tiredness and calmness sub-dimensions. The null hypothesis is rejected and the proposed alternate hypothesis is accepted.

5.4.2 Leaning hypothesis H4

From the t-test it is clear that both leaning measures does not have statistically significant differences between the groups. The p-values are far from $p \leq 0.05$. The significance levels are as high as $p = .748$ for the lidar measure and $p = .675$ for the chair leaning measure. When it comes to correlations the leaning measures actually show some correlation with the HRV features. Both measures show statistically significant correlation with both pNN50 and LF power. The lidar has correlations of $r = .336$, $p = .006$ and $r = .257$, $p = .039$ with those two features. The chair leaning have correlations of $r = -.261$, $p = 0.36$, and $r = -.283$, $p = 0.22$. The leaning measures show no statistically significant correlation with the subjective measures, with the correlation between the lidar and the A2 sub-dimension of $r = -.190$, $p = .124$ being the closest one.

The tests also show that the two leaning measures have negative correlation between themselves, which should indicate that they are opposite directed, but they are not. The most plausible explanation for this negative correlation is that the machine learning for the chair leaning was not accurate at all. Because of the lidar measure are considered the most reliable of the two, the chair measure are not further considered.

The small correlation between the objective HRV features and the lidar measure are not enough to reject the null hypothesis, but it show an indication of association between leaning and HRV. Therefore the null hypothesis still stands after this experiment, but the trends will be further discussed in **section 6.3**.

This page is be intentionally left blank.

Chapter 6

Discussion

In hinesight of the experiment described in this thesis, there are several interesting findings both expected and unexpected. Some findings are interesting regarding the intention of thesis, others are interesting because of the way they affected or limited the work done in this thesis. All of these aspects will be discussed in this chapter. Also some implications and recommendation for future similar research are included in the end of this chapter.

6.1 Result Discussion

First it must be mentioned that the statistical analyzes all have at least 30 cases in each group. This means that the result must be treated as implications, but they can not in any way be seen as a representative representation of the population, more on this in **sec. 6.2**.

The basis for evaluating the level of leaning in association with arousal, derives from subjective observations from real life situations, were for instance while playing video games, people tend to lean forward in thrilling and crucial situations. The results show little indication that the subjects leans more forward in situations with high activation. There are correlation with HRV, and the direection of this correlation is even the opposite direction compared to the assumption. The validity of the HRV data are suspected to be low, which will be discussed in **sec. 6.2**.

Regarding the number of position changes this thesis provides evidence that the hypothesis H3 can be supported. This is in spite of the many limitations discussed in **sec. 6.2**. Position changes showed highly significant difference in distribution between the groups that was derived from subjective arousal. The mean difference of more than 30 means that the distribution are centered at more than 30 position changes from each other. As mentioned earlier also the correlation test with the subjective measures show an overall strong association to the position changes variable. It is important to remember that the position changes are derived from nominal positions, and that the level of the movement are not considered in this variable, the measure can hence not be seen as a measure of activity, because the number of changes can be just between two positions that are close to each other and require little or no activity change.

6.2 Limitations

Being fresh in research and experimental work at this magnitude, the personal learning from this thesis is enormous. Most fascinating and at times frustrating was the amount of limitations that surfaced and had to be taken in to account in some way. Some of the challenges that surfaced from the pilot testing, was taken care of in the design, but not every challenge was possible handle in the scope of this thesis. Also some limitations was discovered in hindsight of the experiment. This section will describe these limitations witch regard the thesis as a whole.

Firstly the chair setup must be discussed. In **chap. 3** it was disclosed that the chair was the first stage in a development process for further research. However some aspects of this stage had to be modified. For instance was the arm rest sensors not used, they were excluded because of the sampling rate for the arduino was greatly reduced, and a decision was taken that it was more important to have good sampling rate for the pressure sensors rather then including the arm rest sensors at the cost of loosing sampling rate. In the case of the pressure sensors, they should have had higher resolution, in other words they reach their maximum reading to fast. This would have been possible by using smaller pull up-resistors in the circuit. To put it on the edge, they almost functioned as digital sensors, were the output was either under pressure or not.

Next up are the facilitation of the experiment. As mentioned in **sec. 4.4** the setup was supposed to be neutral, in other words somewhere between real situation and white room. This was somewhat achieved, but disturbances from the hallway was unavoidable, also in spite of having a sign on the door, some outsiders could not resist sticking their head in and distracting the subjects. Also the lighting was controlled by cardboard walls in front of the window, but different ambient lighting made it impossible to have the same conditions for every scenario for every participant. A camera was also used to monitor the experiment, but low lighting combined with difficulties in finding a good angle for the camera resulted in the utilization of the camera feed was low.

The central point of interactions in the experiment was the TV. Most of the time only the center part of the TV was used to display information and display the task. The fact that the main stimuli was from this one point might affect the posture. A setup were the interaction are placed at different spacial orientations might be more helpful if the affect from different types of events are to be evaluated.

Moving on to the contextual part of the experiment design. In the scope of the work done in this thesis, it made more sense to set up an experiment in collaboration with another master student. The rewards from doing so out-weighted the costs. The rewards include a better integration of the different stimuli and self report questionnaires, and a much better designed task for the experiment. The cost is that the experiment is designed to measure two aspects of affective emotions at the same time, and when selecting stimuli it is needed to consider both dimensions at the same time. In combination with other aspects the result was that the arousal levels from the scenarios was not as differentiated as desired, and the new subjective grouping was hence needed. To reduce the duration of the experiment a real baseline scenario was not included, in a set up that only considered one of the dimensions there would be at most two scenarios, even one might suffice. Also the order of the scenarios was always the same, and even if the learning effect was not important in this study, the fact that initial experimental effects always affected the same scenario are not optimal. Another way of exploring the effects of arousal, is having one group or scenario with an activating task, and having another group or scenario that is basically doing nothing ”rest” or ”brake”. This will however compare the task of doing nothing against doing a mentally demanding task, and will not give nuances in the arousal levels of doing tasks with different stimuli and difficulties.

It is an unavoidable truth that the human factors are going to be challenging in a study of human emotions. The participants of this experiment imposes some more limitations concerning the work done. It is once more repeated that the participants does not represent the entire population in any way. In fact they represent a quite monotonic part of the population, they are mostly engineering students or freshly graduated engineering students. Almost all of them are in their twenties and there are only seven female of a total of 34. Not only are most of them engineering students, but many of them are in the process of finishing their masters degree, which can be an emotional load that they have not experienced before. Since some of the participants belong to the same department as the author, they might have some bias towards what is being measured, and this might affect the subjective measures especially. The fact that some of the participants knew each other and was aware of each others participation, had some effects regarding arousal caused by competitive spirit, this was most visible in parts of the experiment that was not supposed to be inspired by competition.

As mentioned in the beginning of **chap. 5** the Hawthorne Effect is probably present in most of the participants. This involves that the subjects modify their behavior because of the fact that they are being observed. The initial procedure of embedding the subjects with sensors, are probably also affecting the behavior of the subjects. It is suspected that this effect is strongest in the beginning, and it was observed that the heart rate of most of the subjects were unnaturally high before starting the experiment. Again if this effect was strongest in the beginning of the experiment, the order of the scenarios are hence affected. The lack of randomizing the order of the scenarios can hence be seen as weakness in the design of the experiment, but it was suspected that by having a high arousal scenario before a low arousal scenario would be greatly affected by the delay of secretion of hormones caused by the high arousal in previous scenario, hence the low arousal scenario was always the first scenario. The learning effects are not important in this study since the performance is unimportant, giving the choice to not have random order more precedence.

The process were the scientist or the person performing the research influence the results in order to achieve a certain outcome are often referred to as research or experimenter bias. The grouping done in this thesis, which is based on data collected in the experiment and not set as part of the experiment design can be considered as experimenter bias. To further criticize the decision to use such a grouping, this sort of manipulation can be seen as hunting for significant results. As previously mentioned the reasoning for making such a controversial decision was low distribution difference in arousal between the predetermined scenarios. Again he goal of the experiment was not to evaluate the effect of the different stimuli in the different scenarios, but to see how the new proposed measures associated with the established measures of arousal. It was hence more useful to use a separation in subjective reported arousal to group the cases, at least to see if there was any trends based on self perceived activation.

It was mentioned in the previous chapter that the HRV measures might not be as valid as desired. In some cases the measures are not valid or missing, this is because of problems with the signal from the sensor platform. This is neglectable because there are enough valid cases even with these missing data points. A more worrying aspect is the sampling rate, other studies including HRV have at least a sampling rate of 500 Hz, but the platform used in this experiment could only collect data at a rate of approximately 380 Hz, it is hard to tell if this was enough to have valid data, but it is important to be aware that the data might not be sufficiently precise. Another factor regarding HRV, at least concerning the frequency domain is the duration of the measurement. The five minute scenarios are in the world of HRV considered short term, it should be sufficient for the LF spectrum, but it is barely within the limits.

The machine learning algorithm used for the two chair measures is adopted from the setup used by Jensen et al. (2016) and is not specially adopted for the system in this thesis. Limited knowledge regarding machine learning in general expedited the decision to not design or test alternate algorithm for the work in this thesis. The fact that the leaning measure failed totally is an indication that the classification for the leaning was poorly designed. The position changes was more properly tested, and during continuous real term monitoring it showed high accuracy compared to subjective observations.

The framework and background procedure for this experiment was quite streamlined, but it still relied on having two experimenters present during the entire experiment, were one was in control of stimuli and feedback in addition to ensuring the progress from predetermined stops. The other was in charge of syncing the data from two different data streams. This was done manually by assigning markers in the raw data for starts and stops of each scenario. This process could have been automated, and the data could have been automatically sorted according to the sync markers, which would have saved a lot of work in the data processing.

Finally a lot of external variables was not taken in to consideration in this experiment. Temperature, time of day, personal mood and shape, day of week and so forth was not considered. For instance the day after the Norwegian Independence day, some of the subjects reported poor form caused by activities the day before. Also previous experience with TETRIS was not considered, but it was to some degree visible that it affected the arousal. Some of these variables was collected but not used further in the analyzes, for instance was the amount of coffee consumed the same day collected. In **fig. 6.1** an obvious effect on position changes can be observed.

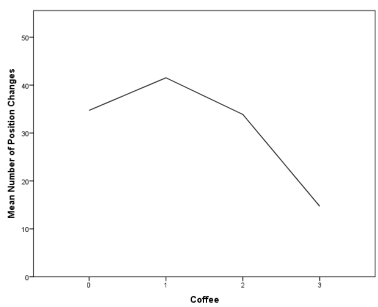


Figure 6.1: Coffee effect on number of position changes

6.3 Future Research

The work done is this thesis can be split in two parts. Piloting and testing of a chair sensor setup, and piloting and development of a experimental framework for human interaction studies.

As for the framework it is done in collaboration with another master student, and can with small modifications be used in a variety of studies regarding human interaction and or user experiences. It is highly recommended that future studies regarding product development utilizes this or a similar framework to evaluate the emotional aspects of the design or product. Even if the framework has its flaws it has potential for further development, and Open Sesame in combination with python scrips has almost limitless possibilities.

Regarding further development of the chair sensor setup. This stage of the development was always meant as the initial stage of a longer development process of a setup for interaction studies. The next obvious step would be to investigate if the current setup makes more sense if raw data from the chair are evaluated instead of features derived from the k-tree classification algorithm. Further to include the arm rest data, and possibly tune the pressure sensors or even exchange the sensors for a sheet-based setup, where a complete pressure map can be extracted. If the chair are to be used for real time monitoring, the capability to extract HRV data from the chair would be extremely useful, but it might be difficult to implement sensors that are sensitive enough to sense the muscular contractions in the heart and at the same time not be affected by ”noise” from other muscles such as the respiration movement.

As for the results from the analyzes in this thesis, the indications that number of position changes are associated with arousal, can further be tested, maybe even regarding the level of activity in mind. Even to include the magnitude of the position changes and investigate if there are some real association present migh be useful. I am confident that further and more comprehensive studies of the posture in this chair can lead to findings that can with confidence be used in interaction studies for real life situations.

This page is be intentionally left blank.

Chapter 7

Conclusion

This thesis started out with the goal to pilot a chair sensor setup that can sense various activity patterns, and explore how much information about the user that can be read out of this activity. It ended up as a study of this setup's capabilities to measure arousal in an experimental setting. To start off the thesis, knowledge of Affective Engineering, emotions and body language was gathered. Then a phase of deriving the needed features the sensor setup needed for the context of the planned experiment, and selection of the technologies used in the sensor setup. Many of the decisions made for the setup was rushed, but most of the wanted features was fulfilled.

The most comprehensive part of this thesis is the experimental setup and procedure, which is done in collaboration with another master student. A lot of time was spent developing and fine tuning everything from physical setup to procedure, and the framework of the experiment can be further used as a foundation for future studies.

In spite of the limitation of the work in this thesis, I would strongly advise and encourage researchers to look further in to the concept of using a chair to gather information about the users emotional state.

In the introduction for this thesis you were presented with the following narrative:
"Imagine you are a ship captain in the future, in fact your title is not ship captain, but ship monitor. You are responsible for monitoring several autonomous ships, and if needed take over the control. You feel tired and unresponsive and you are counting the seconds before your shift ends. Suddenly you have to take controll over two of the ships at the same time, there are alarms going off all around you. You fail your task, and one of the ships crashes in to the docks."
Followed by the question: *"What if the system was able to know what your capabilities were based on your emotion state?"* Have the results of the work done in this thesis helped in reaching the goals in said situation? Maybe not, but it may describe a potential tool for getting one step closer to such a system.

This page is be intentionally left blank.

Bibliography

Acharya, U. R., Joseph, K. P., Kannathal, N., Lim, C. M., Suri, J. S., 2006. Heart rate variability: a review. Medical and biological engineering and computing 44 (12), 1031–1051.

Akselrod, S., Gordon, D., Ubel, F. A., Shannon, D. C., Barger, A. C., 1981. Power spectrum analysis of heart rate fluctuation: a quantitative probe of beat-to-beat cardiovascular control. Tech. rep., MASSACHUSETTS INST OF TECH CAMBRIDGE HARVARD-MIT DIV OF HEALTH SCIENCES AND TECHNOLOGY.

Anttonen, J., Surakka, V., 2005. Emotions and heart rate while sitting on a chair. In: Proceedings of the SIGCHI conference on Human factors in computing systems. ACM, pp. 491–499.

Balters, S., Steinert, M., 2014. Decision-making in engineering-a call for affective engineering dimensions in applied engineering design and design sciences. In: Innovative Design and Manufacturing (ICIDM), Proceedings of the 2014 International Conference on. IEEE, pp. 11–15.

Balters, S., Steinert, M., 2015. Capturing emotion reactivity through physiology measurement as a foundation for affective engineering in engineering design science and engineering practices. Journal of Intelligent Manufacturing, 1–23.

Carney, R. M., Saunders, R. D., Freedland, K. E., Stein, P., Rich, M. W., Jaffe, A. S., 1995. Association of depression witk reduced heart rate variability in coronary artery disease. The American journal of cardiology 76 (8), 562–564.

Coulson, M., 2004. Attributing emotion to static body postures: Recognition accuracy, confusions, and viewpoint dependence. Journal of nonverbal behavior 28 (2), 117–139.

Dahlgaard, J. J., Dahlgaard, J. J., Schütte, S., Ayas, E., Mi Dahlgaard-Park, S., 2008. Kansei/affective engineering design: A methodology for profound affection and attractive quality creation. The TQM Journal 20 (4), 299–311.

Darwin, C., Ekman, P. E., 1872. The expression of the emotions in man and animals.

De Gelder, B., 2006. Towards the neurobiology of emotional body language. Nature Reviews Neuroscience 7 (3), 242–249.

Duffy, E., 1957. The psychological significance of the concept of" arousal" or" activation.". Psychological review 64 (5), 265.

Duffy, E., 1962. Activation and behavior.

Ekkekakis, P., 2013. The measurement of affect, mood, and emotion: A guide for health-behavioral research. Cambridge University Press.

Ekman, P., Friesen, W. V., 1971. Constants across cultures in the face and emotion. Journal of personality and social psychology 17 (2), 124.

Freud, S., Strachey, J. E., 1964. The standard edition of the complete psychological works of sigmund freud.

Gunes, H., Piccardi, M., 2007. Bi-modal emotion recognition from expressive face and body gestures. Journal of Network and Computer Applications 30 (4), 1334–1345.

Hacks, C., 2017. e-health documentation.
URL https://www.cooking-hacks.com/documentation/tutorials/ehealth-biometric-sensor-platform-arduino-raspberry-pi-medical#step4_2

Hunt, W., Mark, W. R., Stoll, G., 2006. Fast kd-tree construction with an adaptive error-bounded heuristic. In: Interactive Ray Tracing 2006, IEEE Symposium on. IEEE, pp. 81–88.

Jensen, M., Wulvik, A., Kriesi, C., Boe, O., Phillip, A., Jors, E., Steinert, M., 2016. Interactions in a world of intelligent products-a case study of a smart and learning office chair.

Kapoor, A., Picard, R. W., Ivanov, Y., 2004. Probabilistic combination of multiple modalities to detect interest. In: Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on. Vol. 3. IEEE, pp. 969–972.

Kärki, S., Lekkala, J., Kuokkanen, H., Halttunen, J., 2009. Development of a piezoelectric polymer film sensor for plantar normal and shear stress measurements. Sensors and Actuators A: Physical 154 (1), 57–64.

Kay, S. M., Marple, S. L., 1981. Spectrum analysis—a modern perspective. Proceedings of the IEEE 69 (11), 1380–1419.

Lacey, J. I., 1967. Somatic response patterning and stress: Some revisions of activation theory. Psychological stress: Issues in research, 14–42.

Malik, M., 1996. Heart rate variability. Annals of Noninvasive Electrocardiology 1 (2), 151–181.

Mathôt, S., Schreij, D., Theeuwes, J., 2012. Opensesame: An open-source, graphical experiment builder for the social sciences. Behavior research methods 44 (2), 314–324.

Matsubara, Y., Nagamachi, M., 1997. Hybrid kansei engineering system and design support. International Journal of Industrial Ergonomics 19 (2), 81–92.

Merletti, R., Parker, P. A., 2004. Electromyography: physiology, engineering, and noninvasive applications. Vol. 11. John Wiley & Sons.

Nagamachi, M., 1995. Kansei engineering: a new ergonomic consumer-oriented technology for product development. International Journal of industrial ergonomics 15 (1), 3–11.

Picard, R. W., Picard, R., 1997. Affective computing. Vol. 252. MIT press Cambridge.

Ramasubramanian, V., Paliwal, K. K., 1992. Fast k-dimensional tree algorithms for nearest neighbor search with application to vector quantization encoding. IEEE Transactions on Signal Processing 40 (3), 518–531.

Rottman, J. N., Steinman, R. C., Albrecht, P., Bigger, J. T., Rolnitzky, L. M., Fleiss, J. L., 1990. Efficient estimation of the heart period power spectrum suitable for physiologic or pharmacologic studies. The American journal of cardiology 66 (20), 1522–1524.

Russel, J. A., Weiss, A., Mendelsohn, G. A., 1989. Affect grid: A single-item scale of pleasure and arousal. Journal of Personality and Social Psychology 57 (3), 493–502.

Russell, J. A., 1978. Evidence of convergent validity on the dimensions of affect. Journal of personality and social psychology 36 (10), 1152.

Russell, J. A., Weiss, A., Mendelsohn, G. A., 1989. Affect grid: A single-item scale of pleasure and arousal. Journal of Personality and Social Psychology 57 (3), 493–502.

Saykrs, B. M., 1973. Analysis of heart rate variability. Ergonomics 16 (1), 17–32.

Shevtsov, M., Soupikov, A., Kapustin, A., 2007. Highly parallel fast kd-tree construction for interactive ray tracing of dynamic scenes. In: Computer Graphics Forum. Vol. 26. Wiley Online Library, pp. 395–404.

Steinert, M., Leifer, L., Jalbokow, K., 2012. Eager: Analyzed—analyzing engineering design activities. In: NSF engineering research and innovation conference, sponsored by the National Science Foundation's Division of Civil, Mechanical and Manufacturing Innovation (CMMI), Boston, USA.

Taelman, J., Vandeput, S., Spaepen, A., Huffel, S., 2009. Influence of mental stress on heart rate and heart rate variability. In: 4th European conference of the international federation for medical and biological engineering. Springer, pp. 1366–1369.

Thayer, J. F., Åhs, F., Fredrikson, M., Sollers, J. J., Wager, T. D., 2012. A meta-analysis of heart rate variability and neuroimaging studies: implications for heart rate variability as a marker of stress and health. Neuroscience & Biobehavioral Reviews 36 (2), 747–756.

Thayer, R. E., 1967. Measurement of activation through self-report. Psychological reports 20 (2), 663–678.

Thayer, R. E., 1986. Activation-deactivation adjective check list: Current overview and structural analysis. Psychological Reports 58 (2), 607–614.

Thayer, R. E., 1989. The biopsychology of mood and arousal: Oxford university press. New York.

Thayer, R. E., 1990. The biopsychology of mood and arousal. Oxford University Press.

Van Houdenhove, B., 2005. In wankel evenwicht: Over stress, levensstijl en welvaartziekten. Lannoo Uitgeverij.

Watson, D., Clark, L. A., Tellegen, A., 1988. Development and validation of brief measures of positive and negative affect: the panas scales. Journal of personality and social psychology 54 (6), 1063.

Watson, D., Tellegen, A., 1985. Toward a consensual structure of mood. Psychological bulletin 98 (2), 219.

Welch, P., 1967. The use of fast fourier transform for the estimation of power spectra: a method based on time averaging over short, modified periodograms. IEEE Transactions on audio and electroacoustics 15 (2), 70–73.

Williams, J. M. E., 1993. Applied sport psychology: Personal growth to peak performance. Mayfield Publishing Co.

Woolf, B. P., Arroyo, I., Cooper, D., Burleson, W., Muldner, K., 2010. Affective tutors: Automatic detection of and response to student emotion. In: Advances in Intelligent Tutoring Systems. Springer, pp. 207–227.

Wulvik, A., Erichsen, J., Steinert, M., 2016. Capturing body language in engineering design–tools and technologies. DS 85-1: Proceedings of NordDesign 2016, Volume 1, Trondheim, Norway, 10th-12th August 2016.

Wundt, W., Pintner, R. T., 1912. An introduction to psychology.

Wundt, W. M., 1896. Grundriss der psychologie. W. Engelmann.

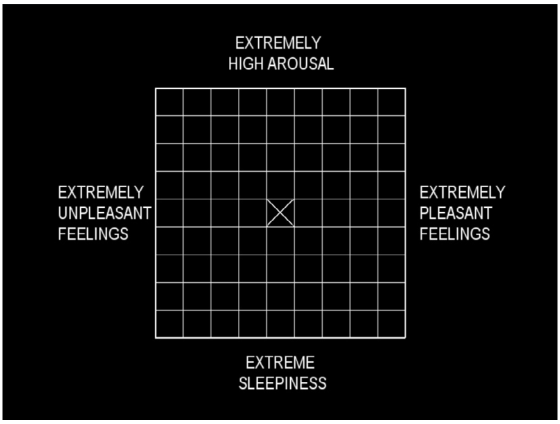
Yakut, O., Solak, S., Bolat, E. D., 2014. Measuring ecg signal using e-health sensor platform. In: International Conference on Chemistry, Biomedical and Environment Engineering, Antalya. pp. 71–75.

Zhou, K., Hou, Q., Wang, R., Guo, B., 2008. Real-time kd-tree construction on graphics hardware. ACM Transactions on Graphics (TOG) 27 (5), 126.

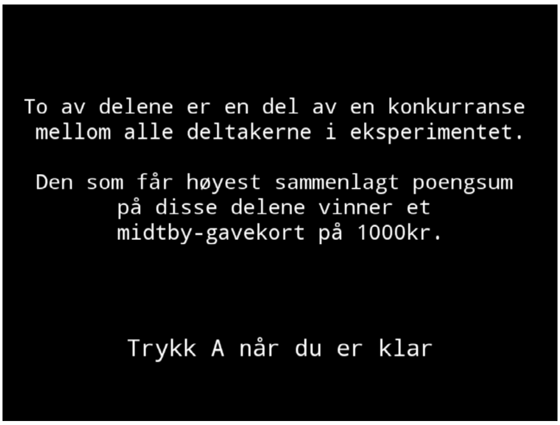
Appendix A - Experimental GUI


Velkommen


Først vil vi be deg om å
rapportere hvordan du føler
deg akkurat nå.
Sett av kryss i følgende skjema
for å indikere din status.
Instruksjoner for skjema
finner du på din venstre side.
Trykk A når du er klar.


EXTREMELY
HIGH AROUSAL
EXTREMELY
UNPLEASANT
FEELINGS
EXTREMELY
PLEASANT
FEELINGS
EXTREME
SLEEPINESS


Dette eksperimentet består av tre deler.
Du skal spille TETRIS i hver del.
Du vil bli utsatt for diverse stimuli,
og spillet vil variere i vanskelighetsgrad.
Trykk A når du er klar.


To av delene er en del av en konkurranse
mellom alle deltakerne i eksperimentet.
Den som får høyest sammenlagt poengsum
på disse delene vinner et
midtby-gavekort på 1000kr.
Trykk A når du er klar


Denne delen er IKKE en del
av konkurransen
Trykk A for å gå videre


Instruksjoner
I denne runden vil spillet se slik ut:
NEXT PIECE:
PERFORMANCE:
SCORE: 0
LINES: 0
Poengsum vises til høyre
for spillområdet.
Prestasjonen din i forhold til
gjennomsnittet vises
helt til høyre.
Trykk A når du er klar.

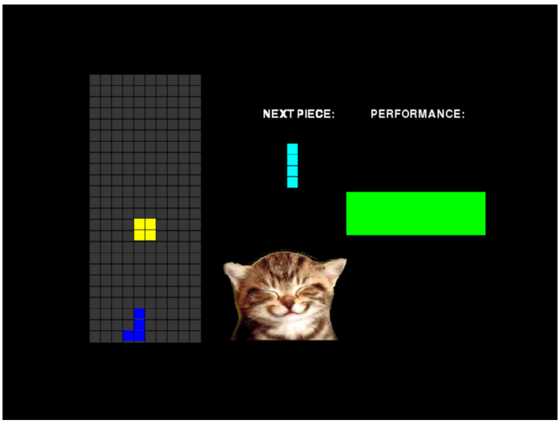

Instruksjoner
Rotate brick
Move
brick
Move
brick
A
Hard drop
Soft drop
Trykk A når du er klar.

Vennligst vent...

3

2

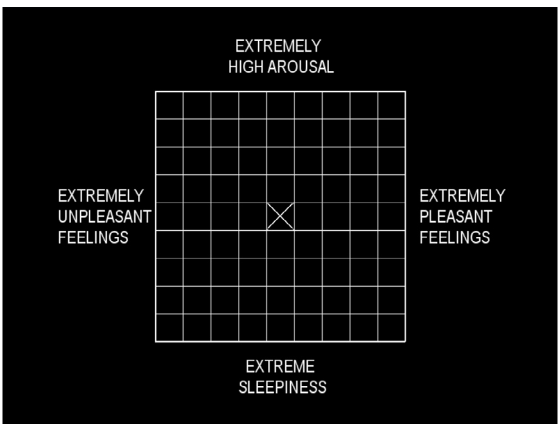
1



Denne delen er nå ferdig.

Vi ber deg om å rapportere hvordan du opplevde denne delen av eksperimentet i tilsvarende skjema som i starten.

Trykk A når du er klar.



Du blir nå presentert med 20 ord som beskriver forskjellige følelser.

Kryss av i passende rute for i hvilken grad du hadde denne følelsen i denne delen av eksperimentet

Din initielle reaksjon er best.

Trykk A når du er klar

Interested

☐	☐	☒	☐	☐
Very slightly or not at all	A little	Moderately	Quite a bit	Extremely

Distressed

☐	☐	☒	☐	☐
Very slightly or not at all	A little	Moderately	Quite a bit	Extremely

Excited

☐	☐	☒	☐	☐
Very slightly or not at all	A little	Moderately	Quite a bit	Extremely

Upset

☐	☐	☒	☐	☐
Very slightly or not at all	A little	Moderately	Quite a bit	Extremely

Strong

☐	☐	☒	☐	☐
Very slightly or not at all	A little	Moderately	Quite a bit	Extremely

Guilty

☐	☐	☒	☐	☐
Very slightly or not at all	A little	Moderately	Quite a bit	Extremely

Scared

☐	☐	☒	☐	☐
Very slightly or not at all	A little	Moderately	Quite a bit	Extremely

Hostile

☐	☐	☒	☐	☐
Very slightly or not at all	A little	Moderately	Quite a bit	Extremely

Enthusiastic

☐	☐	☒	☐	☐
Very slightly or not at all	A little	Moderately	Quite a bit	Extremely

Proud

☐	☐	☒	☐	☐
Very slightly or not at all	A little	Moderately	Quite a bit	Extremely

Irritable

☐	☐	☒	☐	☐
Very slightly or not at all	A little	Moderately	Quite a bit	Extremely

Alert

☐	☐	☒	☐	☐
Very slightly or not at all	A little	Moderately	Quite a bit	Extremely

Ashamed

☐	☐	☒	☐	☐
Very slightly or not at all	A little	Moderately	Quite a bit	Extremely

Inspired

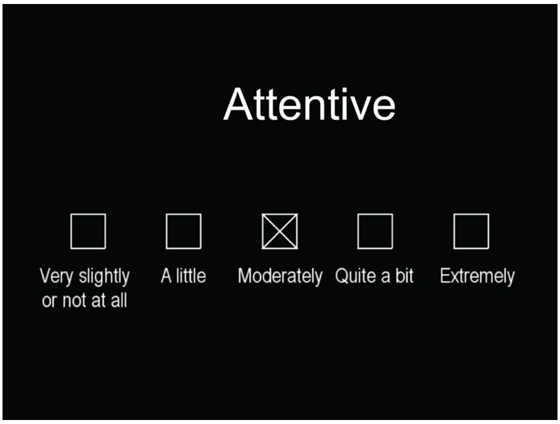
☐	☐	☒	☐	☐
Very slightly or not at all	A little	Moderately	Quite a bit	Extremely

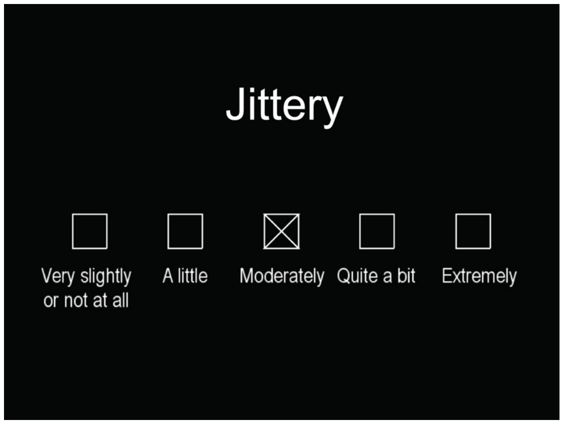
Nervous

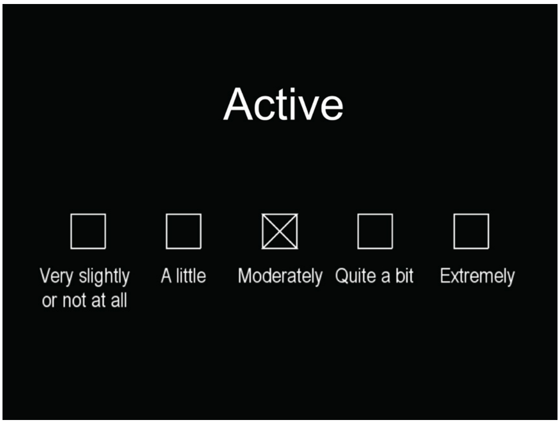
☐	☐	☒	☐	☐
Very slightly or not at all	A little	Moderately	Quite a bit	Extremely

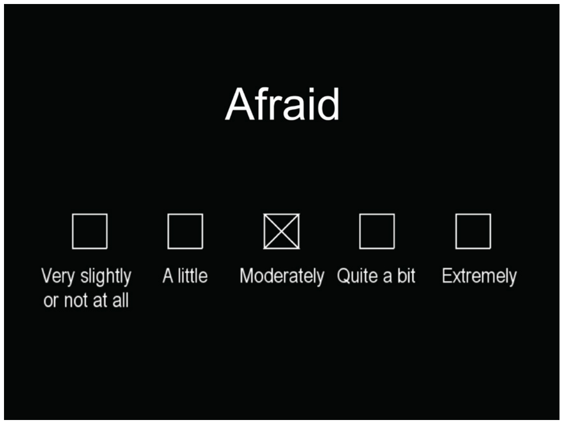
Determined

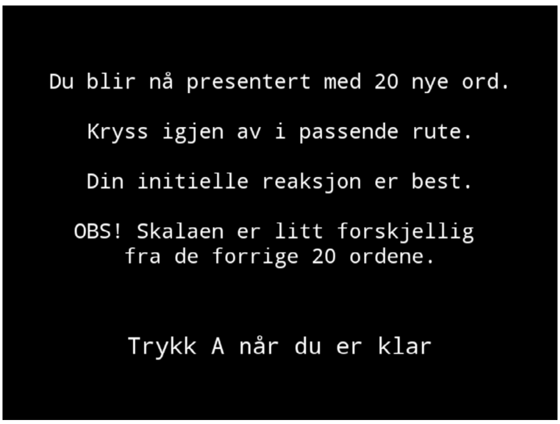
☐	☐	☒	☐	☐
Very slightly or not at all	A little	Moderately	Quite a bit	Extremely

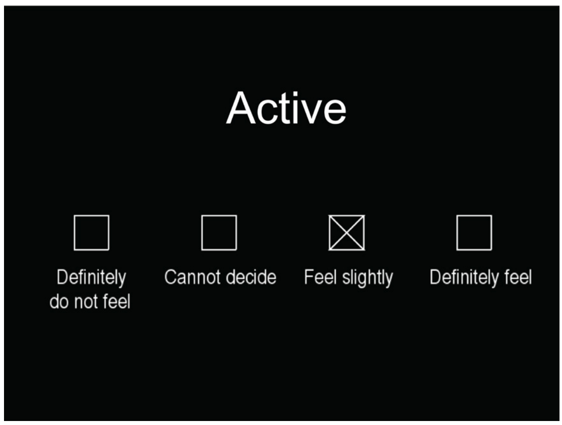

Attentive
Very slightly or not at all
A little
Moderately
Quite a bit
Extremely

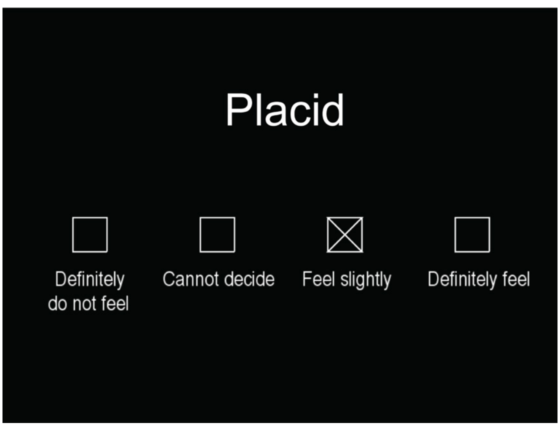

Jittery
Very slightly or not at all
A little
Moderately
Quite a bit
Extremely

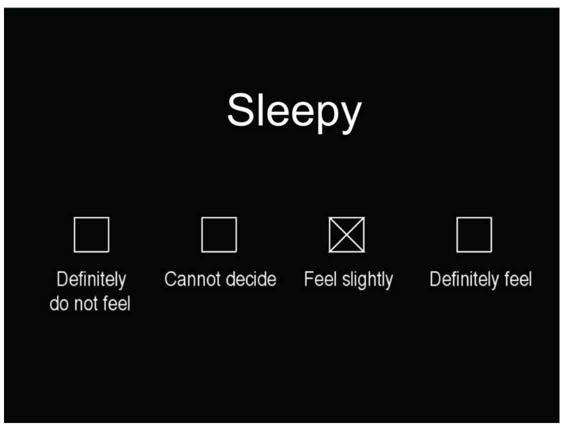

Active
Very slightly or not at all
A little
Moderately
Quite a bit
Extremely


Afraid
Very slightly or not at all
A little
Moderately
Quite a bit
Extremely


Du blir nå presentert med 20 nye ord.
Kryss igjen av i passende rute.
Din initielle reaksjon er best.
OBS! Skalaen er litt forskjellig fra de forrige 20 ordene.
Trykk A når du er klar


Active
Definitely do not feel
Cannot decide
Feel slightly
Definitely feel


Placid
Definitely do not feel
Cannot decide
Feel slightly
Definitely feel


Sleepy
Definitely do not feel
Cannot decide
Feel slightly
Definitely feel

Jittery

☐	☐	☒	☐
Definitely do not feel	Cannot decide	Feel slightly	Definitely feel

Energetic

☐	☐	☒	☐
Definitely do not feel	Cannot decide	Feel slightly	Definitely feel

Intense

☐	☐	☒	☐
Definitely do not feel	Cannot decide	Feel slightly	Definitely feel

Calm

☐	☐	☒	☐
Definitely do not feel	Cannot decide	Feel slightly	Definitely feel

Tired

☐	☐	☒	☐
Definitely do not feel	Cannot decide	Feel slightly	Definitely feel

Vigorous

☐	☐	☒	☐
Definitely do not feel	Cannot decide	Feel slightly	Definitely feel

At-rest

☐	☐	☒	☐
Definitely do not feel	Cannot decide	Feel slightly	Definitely feel

Drowsy

☐	☐	☒	☐
Definitely do not feel	Cannot decide	Feel slightly	Definitely feel

Fearful

☐ Definitely do not feel ☐ Cannot decide ☒ Feel slightly ☐ Definitely feel

Lively

☐ Definitely do not feel ☐ Cannot decide ☒ Feel slightly ☐ Definitely feel

Still

☐ Definitely do not feel ☐ Cannot decide ☒ Feel slightly ☐ Definitely feel

Wide-awake

☐ Definitely do not feel ☐ Cannot decide ☒ Feel slightly ☐ Definitely feel

Clutched-up

☐ Definitely do not feel ☐ Cannot decide ☒ Feel slightly ☐ Definitely feel

Quiet

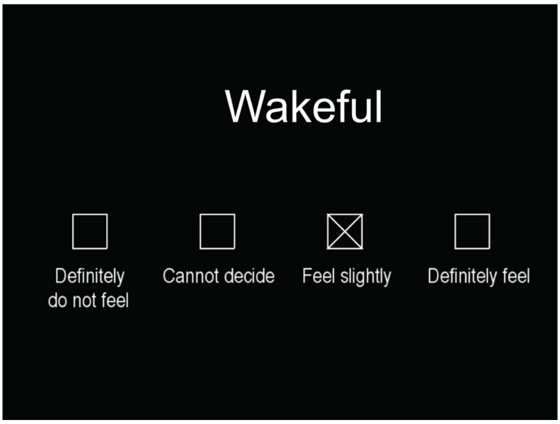
☐ Definitely do not feel ☐ Cannot decide ☒ Feel slightly ☐ Definitely feel

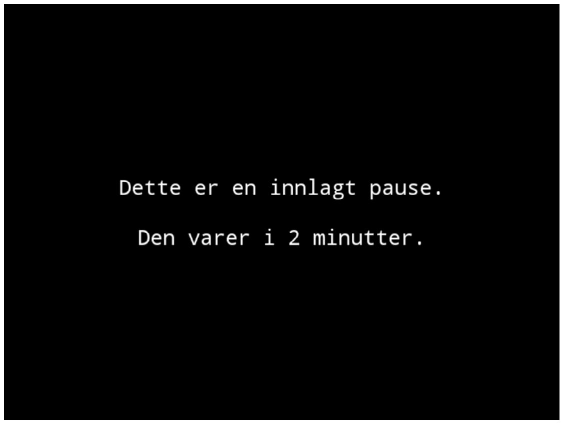
Full-of-pep

☐ Definitely do not feel ☐ Cannot decide ☒ Feel slightly ☐ Definitely feel

Tense

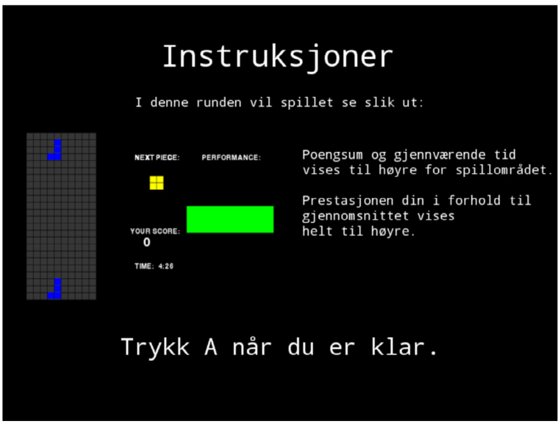
☐ Definitely do not feel ☐ Cannot decide ☒ Feel slightly ☐ Definitely feel


Wakeful
Definitely do not feel
Cannot decide
Feel slightly
Definitely feel


Dette er en innlagt pause.
Den varer i 2 minutter.

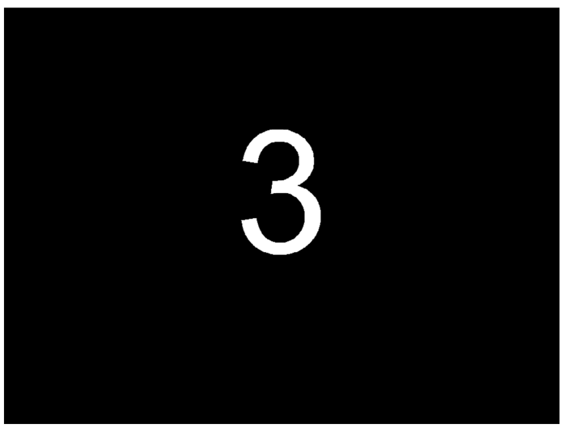

Gjør deg klar


Denne delen er en del av konkurransen.
Ditt mål er å få mest mulig poeng.
Trykk A når du er klar

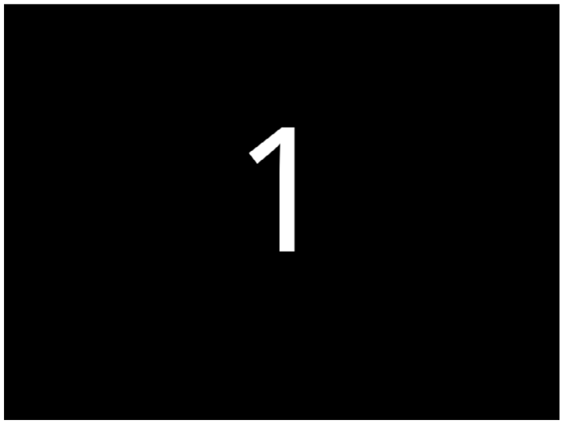

Instruksjoner
I denne runden vil spillet se slik ut:
NEXT PIECE:
PERFORMANCE:
YOUR SCORE:
0
TIME: 4:26
Poengsum og gjenværende tid vises til høyre for spillområdet.
Prestasjonen din i forhold til gjennomsnittet vises helt til høyre.
Trykk A når du er klar.

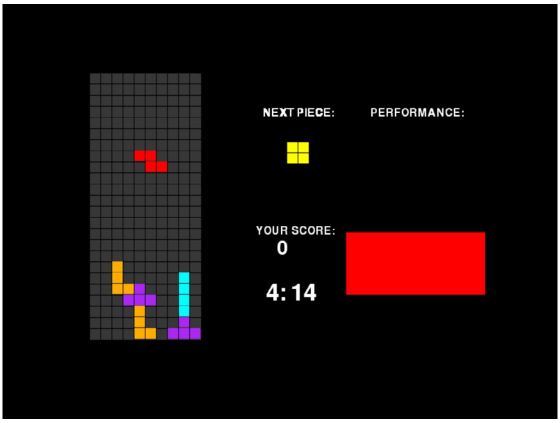

Instruksjoner
Rotate brick
Move brick
Move brick
Soft drop
A
Hard drop
Trykk A når du er klar.

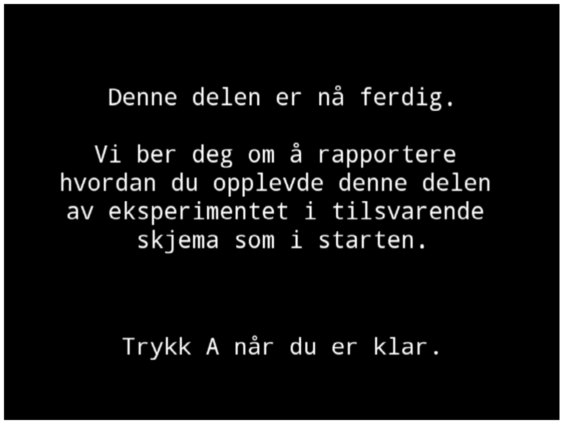

Vennligst vent...

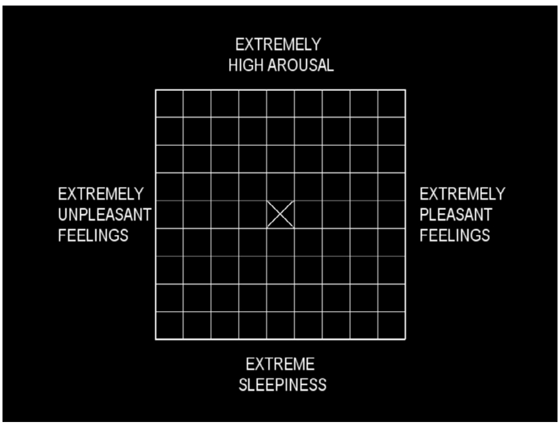

3

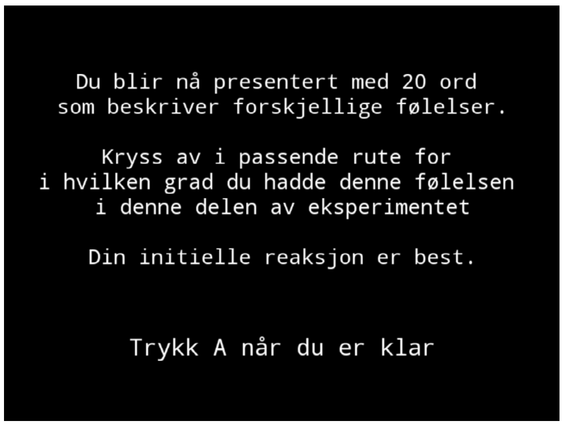

2

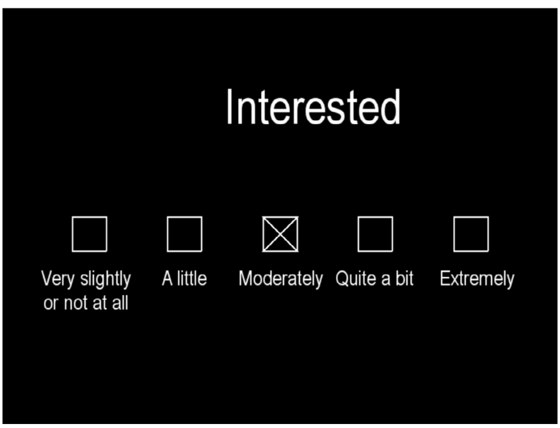

1

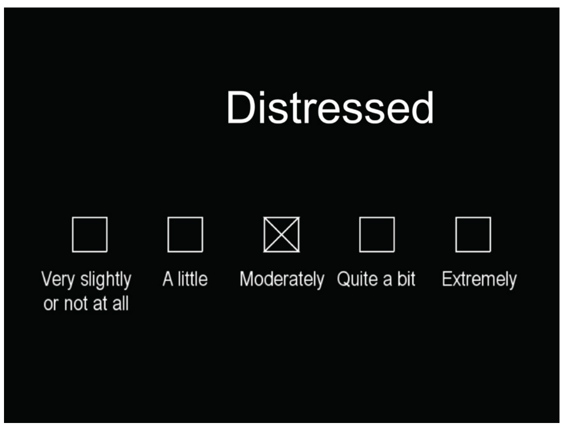

NEXT PIECE:
PERFORMANCE:
YOUR SCORE:
0
4:14


Denne delen er nå ferdig.
Vi ber deg om å rapportere hvordan du opplevde denne delen av eksperimentet i tilsvarende skjema som i starten.
Trykk A når du er klar.


EXTREMELY HIGH AROUSAL
EXTREMELY UNPLEASANT FEELINGS
EXTREMELY PLEASANT FEELINGS
EXTREME SLEEPINESS


Du blir nå presentert med 20 ord som beskriver forskjellige følelser.
Kryss av i passende rute for i hvilken grad du hadde denne følelsen i denne delen av eksperimentet
Din initielle reaksjon er best.
Trykk A når du er klar


Interested
Very slightly or not at all
A little
Moderately
Quite a bit
Extremely


Distressed
Very slightly or not at all
A little
Moderately
Quite a bit
Extremely

Excited

☐ Very slightly or not at all ☐ A little ☒ Moderately ☐ Quite a bit ☐ Extremely

Upset

☐ Very slightly or not at all ☐ A little ☒ Moderately ☐ Quite a bit ☐ Extremely

Strong

☐ Very slightly or not at all ☐ A little ☒ Moderately ☐ Quite a bit ☐ Extremely

Guilty

☐ Very slightly or not at all ☐ A little ☒ Moderately ☐ Quite a bit ☐ Extremely

Scared

☐ Very slightly or not at all ☐ A little ☒ Moderately ☐ Quite a bit ☐ Extremely

Hostile

☐ Very slightly or not at all ☐ A little ☒ Moderately ☐ Quite a bit ☐ Extremely

Enthusiastic

☐ Very slightly or not at all ☐ A little ☒ Moderately ☐ Quite a bit ☐ Extremely

Proud

☐ Very slightly or not at all ☐ A little ☒ Moderately ☐ Quite a bit ☐ Extremely

Irritable

☐	☐	☒	☐	☐
Very slightly or not at all	A little	Moderately	Quite a bit	Extremely

Alert

☐	☐	☒	☐	☐
Very slightly or not at all	A little	Moderately	Quite a bit	Extremely

Ashamed

☐	☐	☒	☐	☐
Very slightly or not at all	A little	Moderately	Quite a bit	Extremely

Inspired

☐	☐	☒	☐	☐
Very slightly or not at all	A little	Moderately	Quite a bit	Extremely

Nervous

☐	☐	☒	☐	☐
Very slightly or not at all	A little	Moderately	Quite a bit	Extremely

Determined

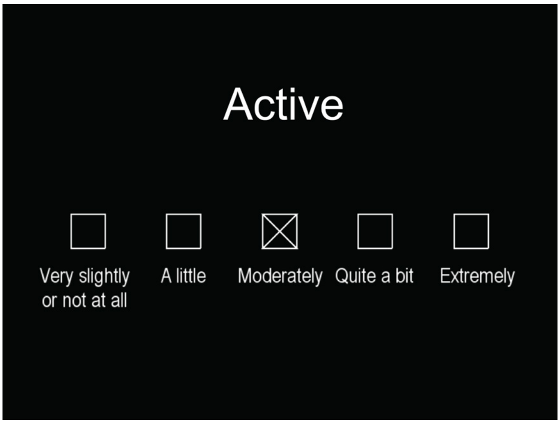
☐	☐	☒	☐	☐
Very slightly or not at all	A little	Moderately	Quite a bit	Extremely

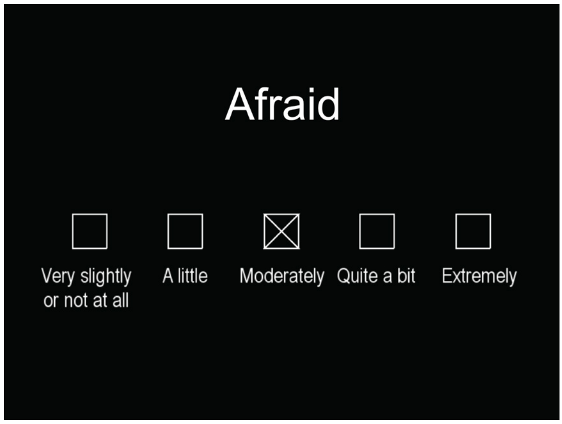
Attentive

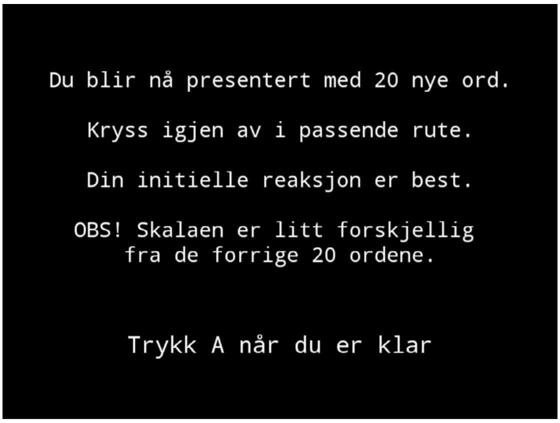
☐	☐	☒	☐	☐
Very slightly or not at all	A little	Moderately	Quite a bit	Extremely

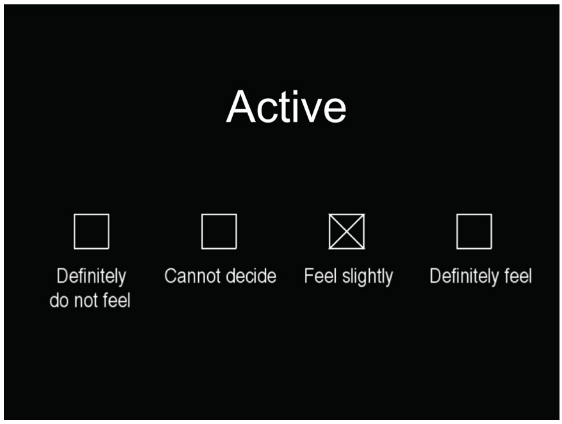
Jittery

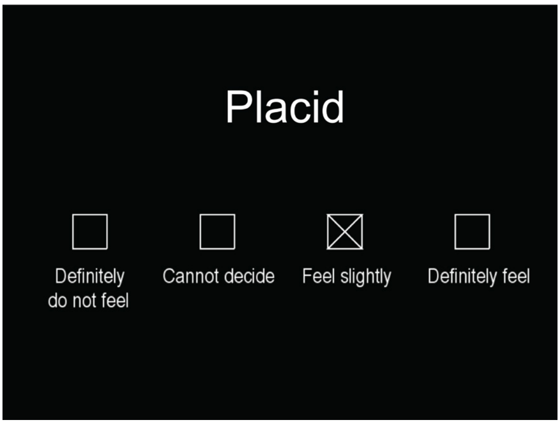
☐	☐	☒	☐	☐
Very slightly or not at all	A little	Moderately	Quite a bit	Extremely

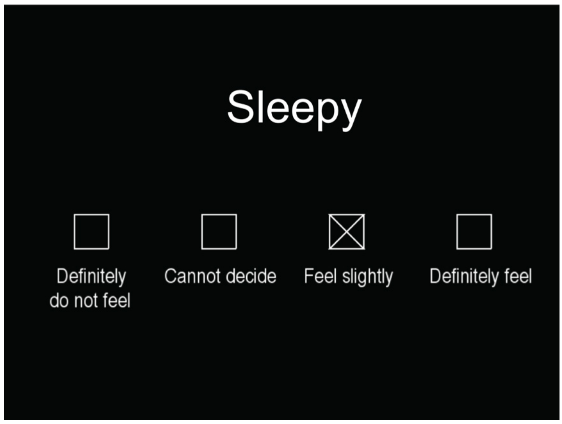

Active
Very slightly or not at all
A little
Moderately
Quite a bit
Extremely

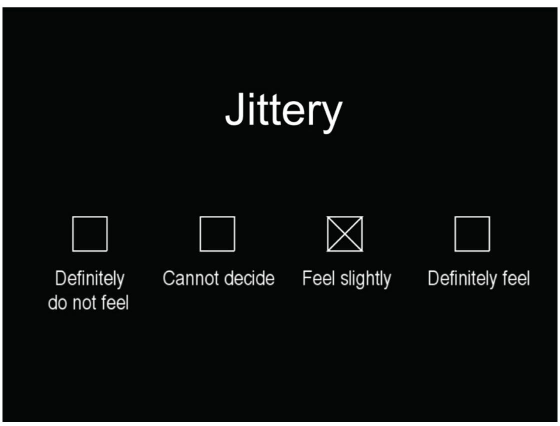

Afraid
Very slightly or not at all
A little
Moderately
Quite a bit
Extremely

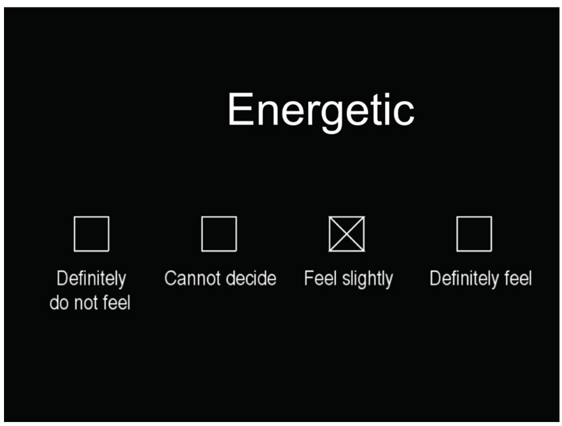

Du blir nå presentert med 20 nye ord.
Kryss igjen av i passende rute.
Din initielle reaksjon er best.
OBS! Skalaen er litt forskjellig fra de forrige 20 ordene.
Trykk A når du er klar


Active
Definitely do not feel
Cannot decide
Feel slightly
Definitely feel


Placid
Definitely do not feel
Cannot decide
Feel slightly
Definitely feel


Sleepy
Definitely do not feel
Cannot decide
Feel slightly
Definitely feel


Jittery
Definitely do not feel
Cannot decide
Feel slightly
Definitely feel


Energetic
Definitely do not feel
Cannot decide
Feel slightly
Definitely feel

Intense

☐ Definitely do not feel ☐ Cannot decide ☒ Feel slightly ☐ Definitely feel

Calm

☐ Definitely do not feel ☐ Cannot decide ☒ Feel slightly ☐ Definitely feel

Tired

☐ Definitely do not feel ☐ Cannot decide ☒ Feel slightly ☐ Definitely feel

Vigorous

☐ Definitely do not feel ☐ Cannot decide ☒ Feel slightly ☐ Definitely feel

At-rest

☐ Definitely do not feel ☐ Cannot decide ☒ Feel slightly ☐ Definitely feel

Drowsy

☐ Definitely do not feel ☐ Cannot decide ☒ Feel slightly ☐ Definitely feel

Fearful

☐ Definitely do not feel ☐ Cannot decide ☒ Feel slightly ☐ Definitely feel

Lively

☐ Definitely do not feel ☐ Cannot decide ☒ Feel slightly ☐ Definitely feel

Still

☐ Definitely do not feel ☐ Cannot decide ☒ Feel slightly ☐ Definitely feel

Wide-awake

☐ Definitely do not feel ☐ Cannot decide ☒ Feel slightly ☐ Definitely feel

Clutched-up

☐ Definitely do not feel ☐ Cannot decide ☒ Feel slightly ☐ Definitely feel

Quiet

☐ Definitely do not feel ☐ Cannot decide ☒ Feel slightly ☐ Definitely feel

Full-of-pep

☐ Definitely do not feel ☐ Cannot decide ☒ Feel slightly ☐ Definitely feel

Tense

☐ Definitely do not feel ☐ Cannot decide ☒ Feel slightly ☐ Definitely feel

Wakeful

☐ Definitely do not feel ☐ Cannot decide ☒ Feel slightly ☐ Definitely feel

Dette er en innlagt pause.

Den varer i 2 minutter.

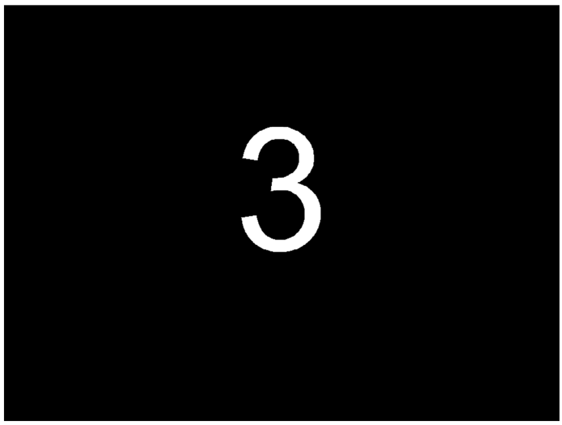

Gjør deg klar


Denne delen er en del av konkurransen.
Ditt mål er å få mest mulig poeng.
Trykk A når du er klar

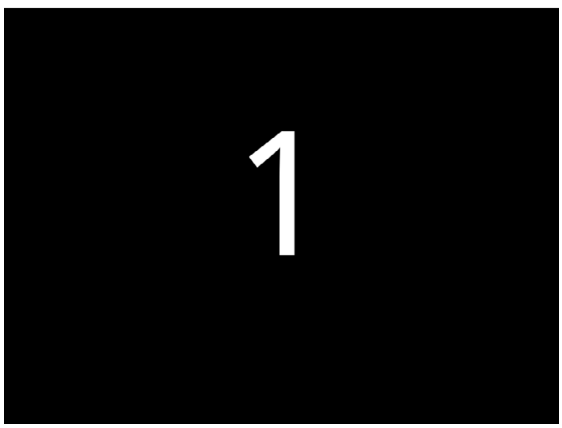

Instruksjoner
I denne runden vil spillet se slik ut:
NEXT PIECE:
PERFORMANCE:
3rd place score:
780
YOUR SCORE:
0
Poengsum vises til høyre
for spillområdet,
her vises også poengsum
satt av tidligere spillere.
Prestasjonen din i forhold til
gjennomsnittet vises
helt til høyre.
Trykk A når du er klar.

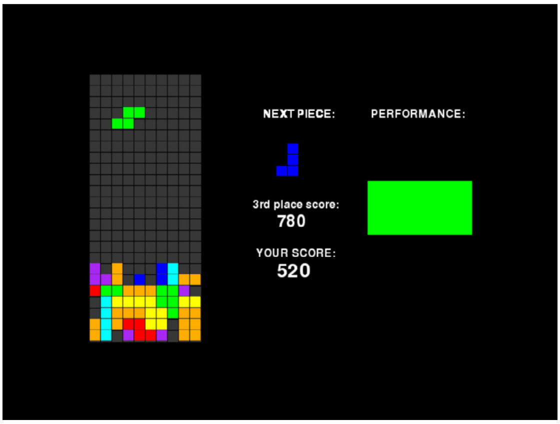

Instruksjoner
Rotate brick
Move brick
Move brick
A
Hard drop
Soft drop
Trykk A når du er klar.


Vennligst vent...
Og husk! Dette er en konkurranse!

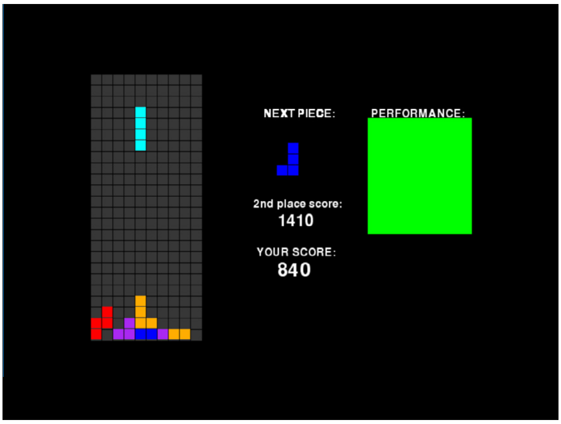

3


2

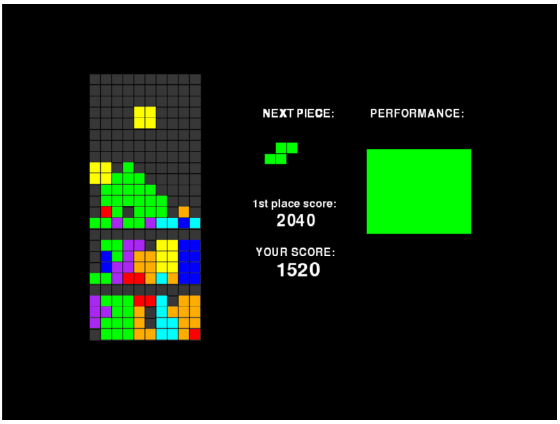

1

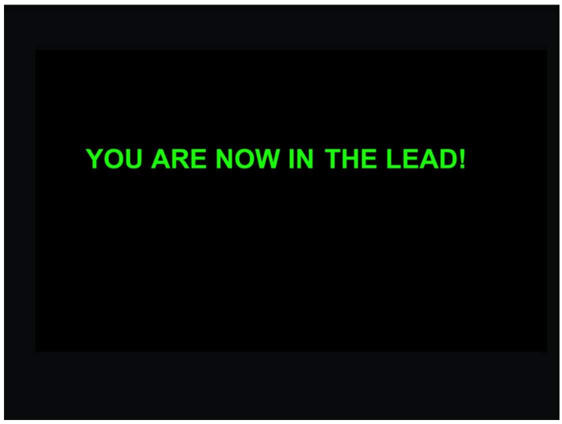

NEXT PIECE:
PERFORMANCE:
3rd place score:
780
YOUR SCORE:
520

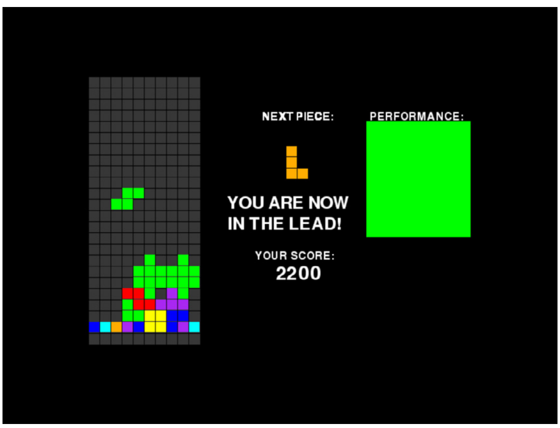

YOU ARE NOW IN 3RD PLACE!

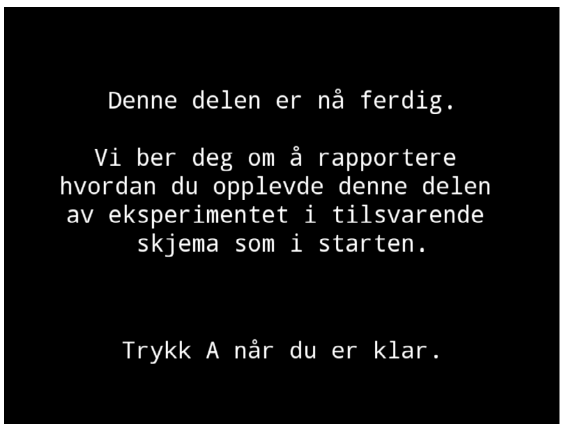

NEXT PIECE:
PERFORMANCE:
2nd place score:
1410
YOUR SCORE:
840

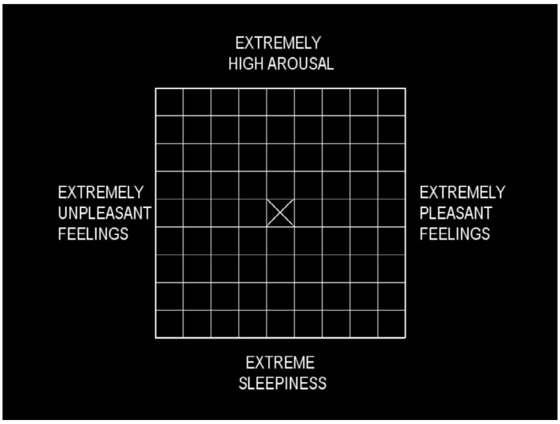

YOU ARE NOW IN 2nd PLACE!

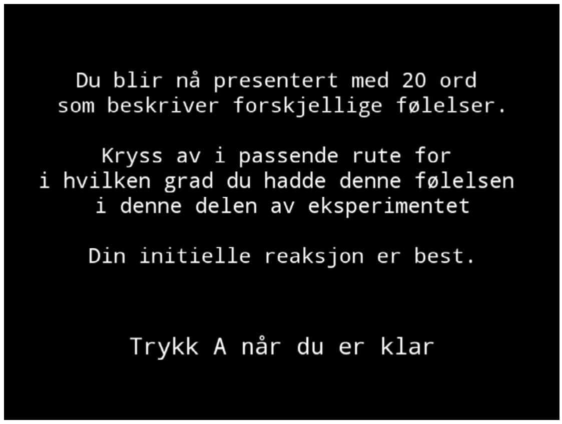

NEXT PIECE:
PERFORMANCE:
1st place score:
2040
YOUR SCORE:
1520

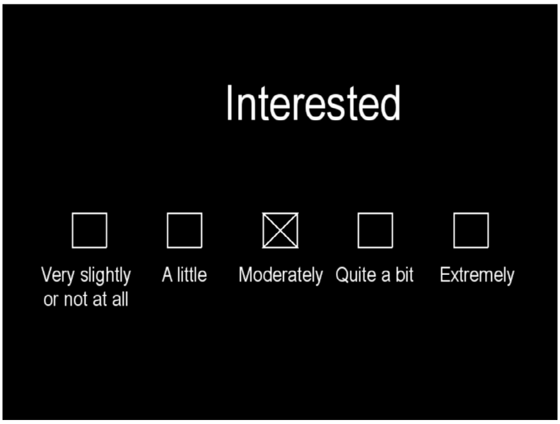

YOU ARE NOW IN THE LEAD!

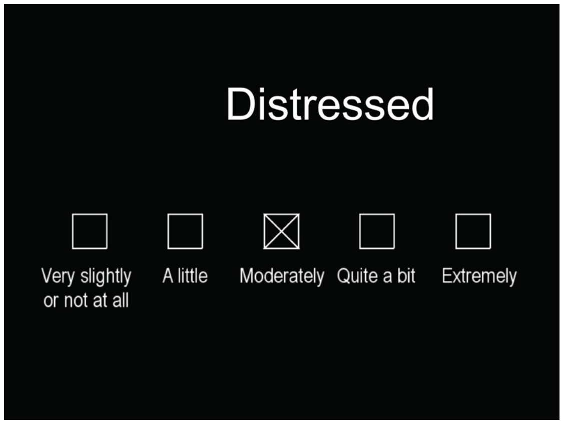

NEXT PIECE:
PERFORMANCE:
YOU ARE NOW IN THE LEAD!
YOUR SCORE:
2200

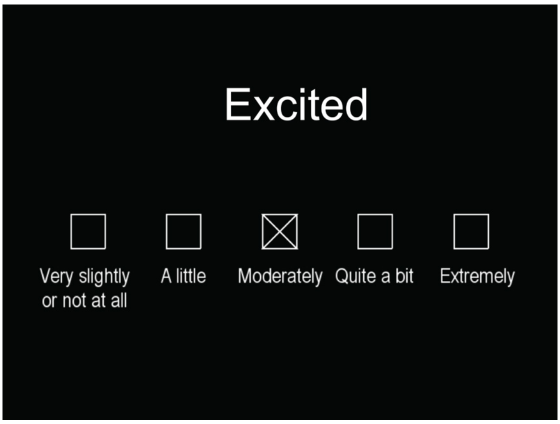

Denne delen er nå ferdig.
Vi ber deg om å rapportere hvordan du opplevde denne delen av eksperimentet i tilsvarende skjema som i starten.
Trykk A når du er klar.

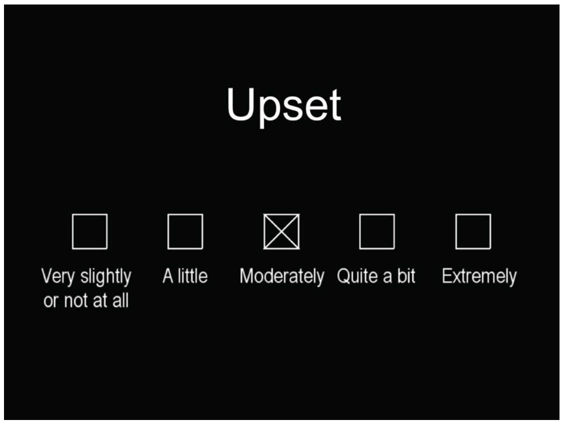

EXTREMELY
HIGH AROUSAL
EXTREMELY
UNPLEASANT
FEELINGS
EXTREMELY
PLEASANT
FEELINGS
EXTREME
SLEEPINESS

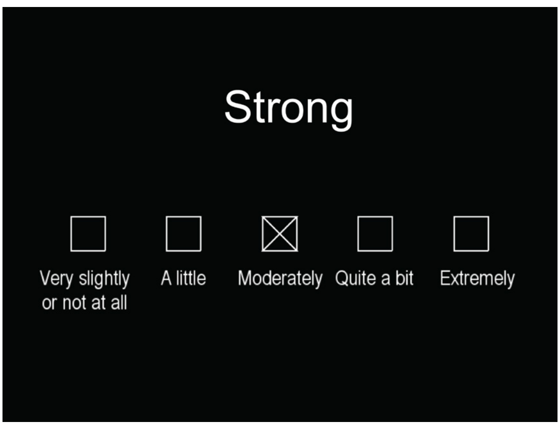

Du blir nå presentert med 20 ord
som beskriver forskjellige følelser.
Kryss av i passende rute for
i hvilken grad du hadde denne følelsen
i denne delen av eksperimentet
Din initielle reaksjon er best.
Trykk A når du er klar

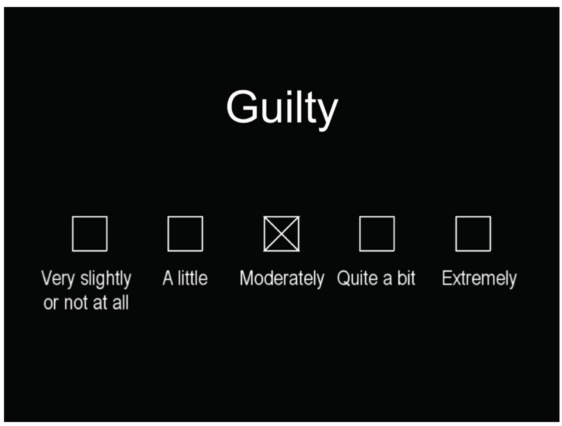

Interested
Very slightly
or not at all
A little
Moderately
Quite a bit
Extremely


Distressed
Very slightly
or not at all
A little
Moderately
Quite a bit
Extremely


Excited
Very slightly
or not at all
A little
Moderately
Quite a bit
Extremely


Upset
Very slightly
or not at all
A little
Moderately
Quite a bit
Extremely


Strong
Very slightly
or not at all
A little
Moderately
Quite a bit
Extremely


Guilty
Very slightly
or not at all
A little
Moderately
Quite a bit
Extremely

Scared

☐ Very slightly or not at all ☐ A little ☒ Moderately ☐ Quite a bit ☐ Extremely

Hostile

☐ Very slightly or not at all ☐ A little ☒ Moderately ☐ Quite a bit ☐ Extremely

Enthusiastic

☐ Very slightly or not at all ☐ A little ☒ Moderately ☐ Quite a bit ☐ Extremely

Proud

☐ Very slightly or not at all ☐ A little ☒ Moderately ☐ Quite a bit ☐ Extremely

Irritable

☐ Very slightly or not at all ☐ A little ☒ Moderately ☐ Quite a bit ☐ Extremely

Alert

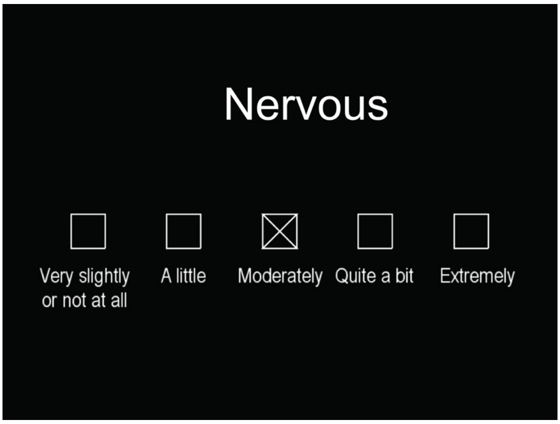
☐ Very slightly or not at all ☐ A little ☒ Moderately ☐ Quite a bit ☐ Extremely

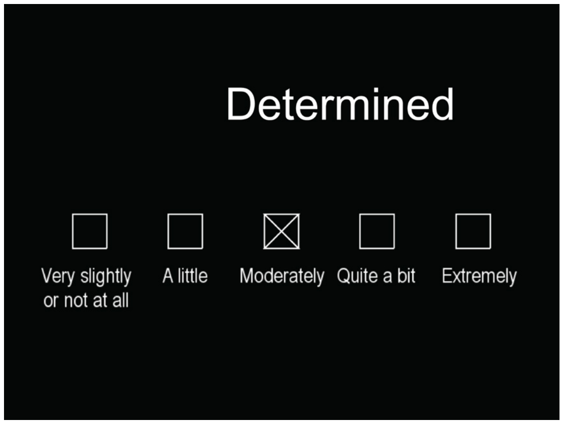
Ashamed

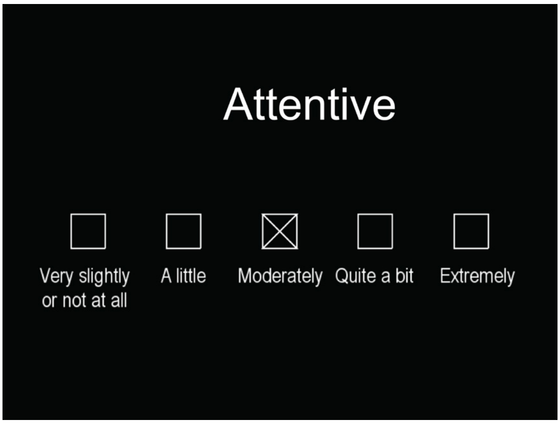
☐ Very slightly or not at all ☐ A little ☒ Moderately ☐ Quite a bit ☐ Extremely

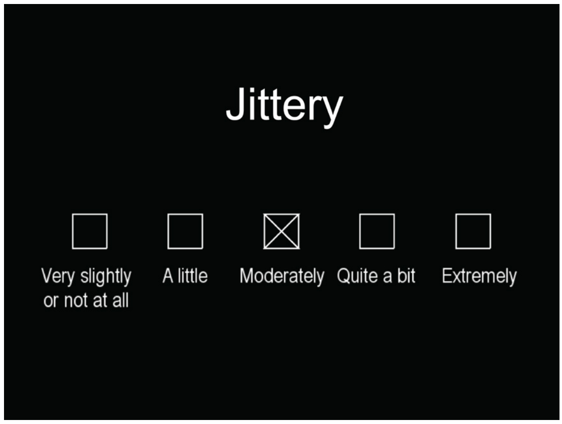
Inspired

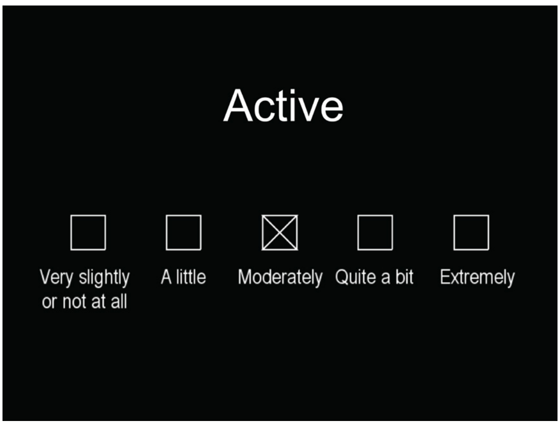
☐ Very slightly or not at all ☐ A little ☒ Moderately ☐ Quite a bit ☐ Extremely

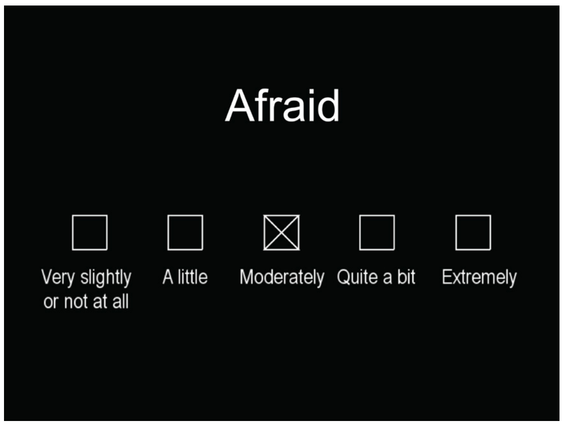

Nervous
Very slightly or not at all
A little
Moderately
Quite a bit
Extremely

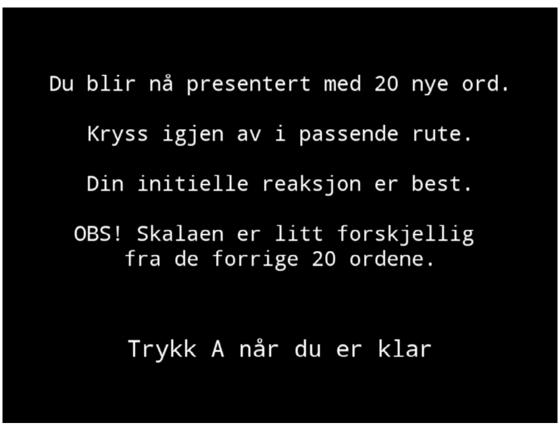

Determined
Very slightly or not at all
A little
Moderately
Quite a bit
Extremely

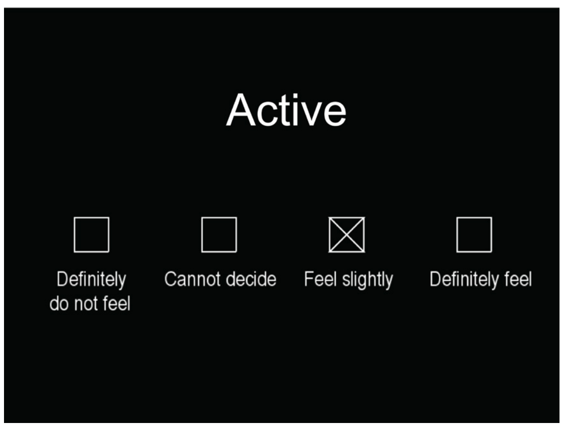

Attentive
Very slightly or not at all
A little
Moderately
Quite a bit
Extremely


Jittery
Very slightly or not at all
A little
Moderately
Quite a bit
Extremely


Active
Very slightly or not at all
A little
Moderately
Quite a bit
Extremely


Afraid
Very slightly or not at all
A little
Moderately
Quite a bit
Extremely


Du blir nå presentert med 20 nye ord.
Kryss igjen av i passende rute.
Din initielle reaksjon er best.
OBS! Skalaen er litt forskjellig fra de forrige 20 ordene.
Trykk A når du er klar


Active
Definitely do not feel
Cannot decide
Feel slightly
Definitely feel

Placid

☐ Definitely do not feel ☐ Cannot decide ☒ Feel slightly ☐ Definitely feel

Sleepy

☐ Definitely do not feel ☐ Cannot decide ☒ Feel slightly ☐ Definitely feel

Jittery

☐ Definitely do not feel ☐ Cannot decide ☒ Feel slightly ☐ Definitely feel

Energetic

☐ Definitely do not feel ☐ Cannot decide ☒ Feel slightly ☐ Definitely feel

Intense

☐ Definitely do not feel ☐ Cannot decide ☒ Feel slightly ☐ Definitely feel

Calm

☐ Definitely do not feel ☐ Cannot decide ☒ Feel slightly ☐ Definitely feel

Tired

☐ Definitely do not feel ☐ Cannot decide ☒ Feel slightly ☐ Definitely feel

Vigorous

☐ Definitely do not feel ☐ Cannot decide ☒ Feel slightly ☐ Definitely feel

At-rest

☐ Definitely do not feel ☐ Cannot decide ☒ Feel slightly ☐ Definitely feel

Drowsy

☐ Definitely do not feel ☐ Cannot decide ☒ Feel slightly ☐ Definitely feel

Fearful

☐ Definitely do not feel ☐ Cannot decide ☒ Feel slightly ☐ Definitely feel

Lively

☐ Definitely do not feel ☐ Cannot decide ☒ Feel slightly ☐ Definitely feel

Still

☐ Definitely do not feel ☐ Cannot decide ☒ Feel slightly ☐ Definitely feel

Wide-awake

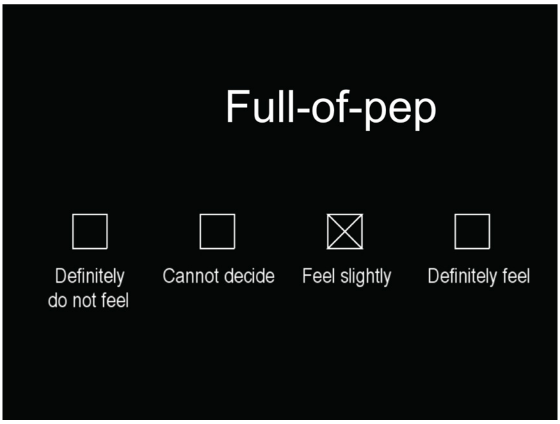
☐ Definitely do not feel ☐ Cannot decide ☒ Feel slightly ☐ Definitely feel

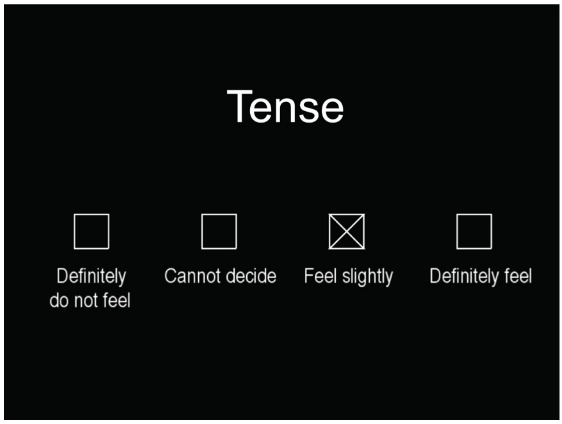
Clutched-up

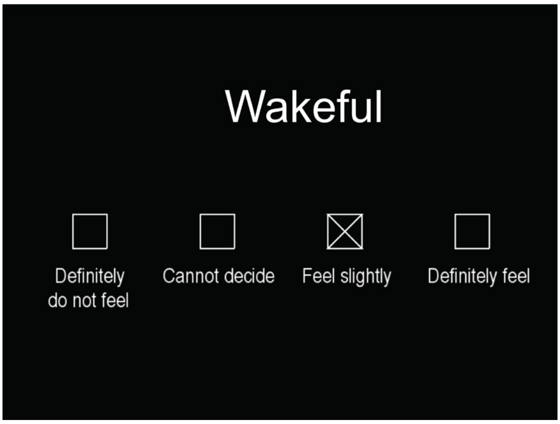
☐ Definitely do not feel ☐ Cannot decide ☒ Feel slightly ☐ Definitely feel

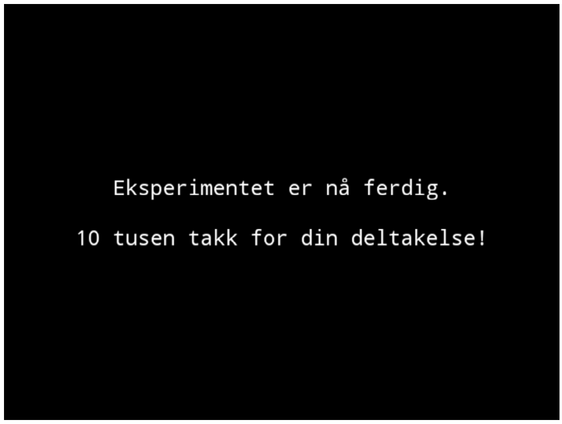
Quiet

☐ Definitely do not feel ☐ Cannot decide ☒ Feel slightly ☐ Definitely feel


Full-of-pep
Definitely do not feel
Cannot decide
Feel slightly
Definitely feel


Tense
Definitely do not feel
Cannot decide
Feel slightly
Definitely feel


Wakeful
Definitely do not feel
Cannot decide
Feel slightly
Definitely feel


Eksperimentet er nå ferdig.
10 tusen takk for din deltakelse!

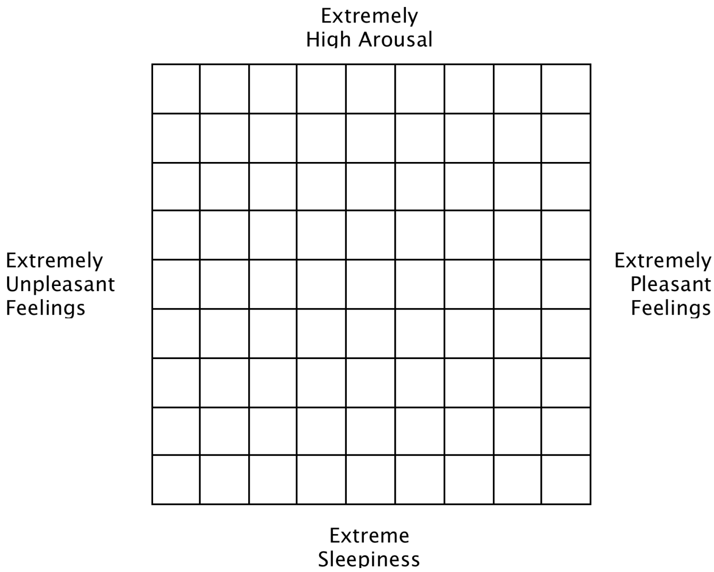
This page is be intentionally left blank.

Appendix B - The Affect Grid

Originally from the appendix of (Russel et al., 1989).

Please use the affect grid below to describe how you feel right now.

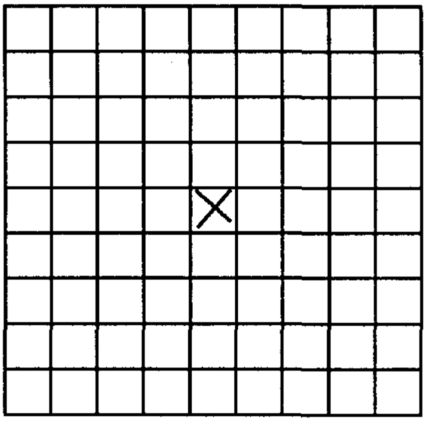
(For instructions on how to use the affect grid below, please refer to the following two pages)



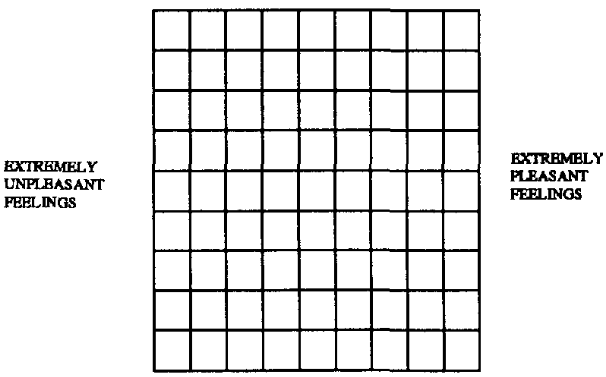
Appendix

The Affect Grid

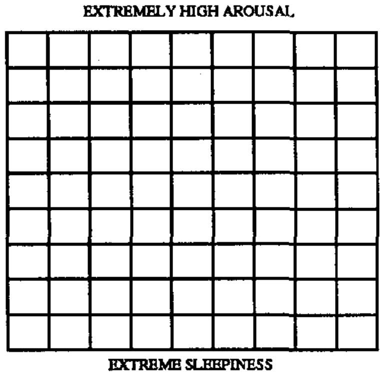
You use the "affect grid" to describe feelings. It is in the form of a square—a kind of map for feelings. The center of the square (marked by X in the grid below) represents a neutral, average, everyday feeling. It is neither positive nor negative.



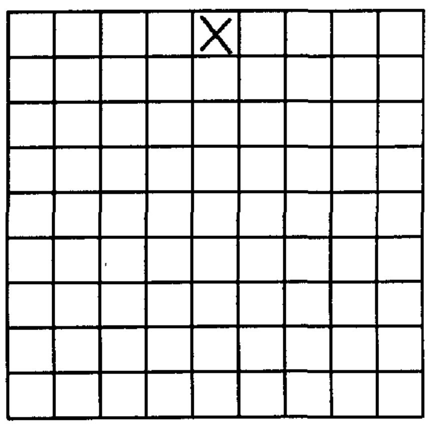
The right half of the grid represents pleasant feelings. The farther to the right the more pleasant. The left half represents unpleasant feelings. The farther to the left, the more unpleasant.



The vertical dimension of the map represents degree of arousal. Arousal has to do with how wide awake, alert, or activated a person feels—independent of whether the feeling is positive or negative. The top half is for feelings that are above average in arousal. The lower half for feelings below average. The bottom represents sleep, and the higher you go, the more awake a person feels. So, the next step up from the bottom would be half awake/half asleep. At the top of the square is maximum arousal. If you imagine a state we might call frantic excitement (remembering that it could be either positive or negative), then this feeling would define the top of the grid.

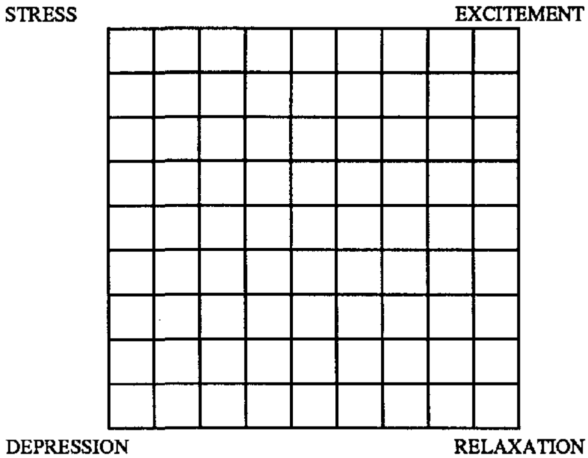


If the "frantic excitement" was positive it would, of course, fall on the right half of the grid. The more positive, the farther to the right. If the "frantic excitement" was negative, it would fall on the left half of the grid. The more negative, the farther to the left. If the "frantic excitement" was neither positive nor negative, then it would fall in the middle square of the top row, as shown below.



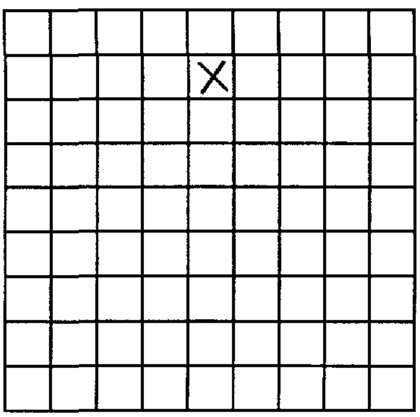
Other areas of the grid can be labeled as well. Up and to the right are feelings of ecstasy, excitement, joy. Opposite these, down and to the left, are feelings of depression, melancholy, sadness, and gloom.

Up and to the left are feelings of stress and tension. Opposite these, down and to the right, are feelings of calm, relaxation, serenity.

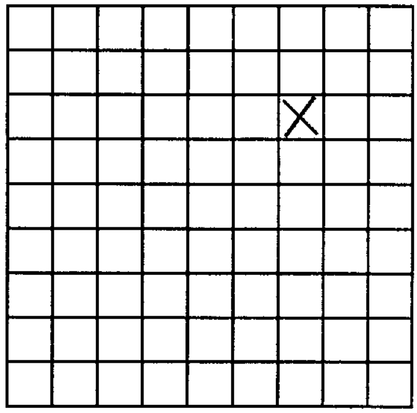


Feelings are complex. They come in all shades and degrees. The labels we have given are merely landmarks to help you understand the affect grid. When actually using the grid, put an X anywhere in the grid to indicate the exact shade and intensity of feeling. Please look over the entire grid to get a feel for the meaning of the various areas.

EXAMPLE: Suppose that you were just surprised. Suppose further that the surprise was neither pleasant nor unpleasant. Probably you would feel more aroused than average. You might put your mark as shown.



EXAMPLE: Suppose, instead, that you were only mildly surprised but that the surprise was a mildly pleasant one. You might put your mark as shown below.



Received October 22, 1987
Revision received January 13, 1989
Accepted February 10, 1989 ■

Affect Grid GUI Code

```
import pygame
import time
import sys
from pygame import K_SPACE, K_w, K_s, K_a, K_d
from pygame.locals import *

def get_pygame_events():
pygame_events = pygame.event.get()
return pygame_events

st = 1
affectresult = (0,0)
pygame.init()
sizex = 800
sizey = 600
screen = pygame.display.set_mode((sizex,sizey))

#colors
red = (255,0,0)
green = (0,255,0)
blue = (0,0,255)
black = (0,0,0)
white = (255,255,255)

myfont = pygame.font.SysFont("arial", 25)
extremely = myfont.render("EXTREMELY", 1, white)
extreme = myfont.render("EXTREME", 1, white)
highar = myfont.render("HIGH AROUSAL",1,white)
lowar = myfont.render("SLEEPINESS", 1, white)
unpleasant = myfont.render("UNPLEASANT", 1, white)
feelings = myfont.render("FEELINGS", 1, white)
pleasant = myfont.render("PLEASANT", 1, white)


pygame.draw.rect(screen, white, ((sizex/2)-180,(sizey/2)-180, 360, 360),


'''
x = 0
y = 0
for i in range(0,9):
for j in range(0,9):
pygame.draw.rect(screen, white, (200+i, 100+i, 40, 40), 1)
```

```
x += 40
x=0
y+=40
'''

offset = 40
theight = 400
twidth = 400
for i in xrange(9): pygame.draw.line(screen, white, (220+i*offset, 120),
for i in xrange(9): pygame.draw.line(screen, white, (220, 120+i*offset),

screen.blit(extremely, (335, 40))
screen.blit(highar, (325, 70))
screen.blit(extreme, (350, 500))
screen.blit(lowar, (340, 530))
screen.blit(extremely, (80, 255))
screen.blit(unpleasant, (80, 285))
screen.blit(feelings, (80, 315))
screen.blit(extremely, (600, 255))
screen.blit(pleasant, (600, 285))
screen.blit(feelings, (600, 315))

posx = 380
posy = 280

pygame.draw.line(screen,white,(posx,posy),(posx+40,posy+40))
pygame.draw.line(screen,white,(posx+40,posy),(posx,posy+40))

def move():
pygame.draw.rect(screen,black,(lastposx+2,lastposy+2,36,36),0)
pygame.draw.line(screen,white,(posx+2,posy+2),(posx+36,posy+36),2)
pygame.draw.line(screen,white,(posx+36,posy+2),(posx+2,posy+36),2)

while st == 1:
lastposx=posx
lastposy=posy
keys_pressed = get_pygame_events()
for event in keys_pressed:
if event.type == pygame.KEYDOWN:
if event.key == K_w:
posy-=40
if event.key == K_s:
posy+=40
if event.key == K_a:
posx-=40
```

```
if event.key == K_d:
posx+=40
if event.key == K_SPACE:
affectresult = (((posx-220)/40)+1, ((posy-120)/40)+1)
log.write(affectresult)
st = 0
if posx<220 or posx>540:
posx = lastposx
if posy<120 or posy>440:
posy = lastposy
move()
pygame.display.update()
```

This page is be intentionally left blank.

Appendix C - PANAS

Originally from the appendix of (Watson et al., 1988).

Appendix

The PANAS

This scale consists of a number of words that describe different feelings and emotions. Read each item and then mark the appropriate answer in the space next to that word. Indicate to what extent [INSERT APPROPRIATE TIME INSTRUCTIONS HERE]. Use the following scale to record your answers.

1	2	3	4	5
very slightly or not at all	a little	moderately	quite a bit	extremely

______ interested	______ irritable
______ distressed	______ alert
______ excited	______ ashamed
______ upset	______ inspired
______ strong	______ nervous
______ guilty	______ determined
______ scared	______ attentive
______ hostile	______ jittery
______ enthusiastic	______ active
______ proud	______ afraid

We have used PANAS with the following time instructions:

Moment	(you feel this way right now, that is, at the present moment)
Today	(you have felt this way today)
Past few days	(you have felt this way during the past few days)
Week	(you have felt this way during the past week)
Past few weeks	(you have felt this way during the past few weeks)
Year	(you have felt this way during the past year)
General	(you generally feel this way, that is, how you feel on the average)

Received May 10, 1987
Revision received September 14, 1987
Accepted November 11, 1987 ■

PANAS GUI Code

```
import pygame
import time
import sys
from pygame import K_SPACE, K_w, K_s, K_a, K_d
from pygame.locals import *

def get_pygame_events():
        pygame_events = pygame.event.get()
        return pygame_events

wordcount = 0
st = 1
pygame.init()
sizex = 800
sizey = 600
screen = pygame.display.set_mode((sizex,sizey))
pos = 3
results = []

#colors
red = (255,0,0)
green = (0,255,0)
blue = (0,0,255)
black = (0,0,0)
white = (255,255,255)

myfont = pygame.font.SysFont("arial", 70)
myfont2 = pygame.font.SysFont("arial", 30)
alt1 = myfont2.render("Very slightly",1,white)
alt12 = myfont2.render("or not at all",1,white)
alt2 = myfont2.render("A little",1,white)
alt3 = myfont2.render("Moderately",1,white)
alt4 = myfont2.render("Quite a bit",1,white)
alt5 = myfont2.render("Extremely",1,white)
screen.blit(alt1, (55, 370))
screen.blit(alt12, (59, 405))
screen.blit(alt2, (230, 370))
screen.blit(alt3, (340, 370))
screen.blit(alt4, (480, 370))
screen.blit(alt5, (630, 370))


wordlist = ['Interested', 'Distressed', 'Excited','Upset','
   Strong','Guilty','Scared','Hostile','Enthusiastic','
```

```
    Proud','Irritable','Alert','Ashamed','Inspired','Nervous
    ','Determined','Attentive','Jittery','Active','Afraid']

def words(counter):
        return wordlist[counter]

def drawWord(counter):
        pygame.draw.rect(screen,black,(0,0,800,200),0)
        word = myfont.render(wordlist[counter], 1, white)
        screen.blit(word, (320, 100))

count=0
for count in xrange(5): pygame.draw.rect(screen, white,
    (100+count*(87.5+50),300,50,50),2);count+=1


posx = 375
posy = 300

pygame.draw.line(screen,white,(posx,posy),(posx+50,posy+50)
    ,3)
pygame.draw.line(screen,white,(posx+50,posy),(posx,posy+50)
    ,3)

def move():
        pygame.draw.rect(screen,black,(lastposx+2,lastposy
            +2,47,47),0)
        pygame.draw.line(screen,white,(posx,posy),(posx+50,
            posy+50),3)
        pygame.draw.line(screen,white,(posx+50,posy),(posx,
            posy+50),3)

def erase():
        pygame.draw.rect(screen,black,(100+2,300+2,47,47)
            ,0)
        pygame.draw.rect(screen,black,(237.5+2,300+2,47,47)
            ,0)
        pygame.draw.rect(screen,black,(375+2,300+2,47,47)
            ,0)
        pygame.draw.rect(screen,black,(512.5+2,300+2,47,47)
            ,0)
        pygame.draw.rect(screen,black,(650+2,300+2,47,47)
            ,0)

while wordcount<20:
```

```
lastposx=posx
lastposy=posy
lastpos=pos
keys_pressed = get_pygame_events()
for event in keys_pressed:
        if event.type == pygame.KEYDOWN:
                if event.key == K_a:
                        posx-=87.5+50
                        pos-=1
                if event.key == K_d:
                        posx+=87.5+50
                        pos+=1
                if event.key == K_SPACE:
                        result = pos
                        results.append(result)
                        print(words(wordcount)+str(
                           result))
                        if wordcount == 19:
                                PA = results[0]+
                                   results[2]+
                                   results[4]+
                                   results[8]+
                                   results[9]+
                                   results[11]+
                                   results[13]+
                                   results[15]+
                                   results[16]+
                                   results[18]
                                NA = results[1]+
                                   results[3]+
                                   results[5]+
                                   results[6]+
                                   results[7]+
                                   results[10]+
                                   results[12]+
                                   results[14]+
                                   results[17]+
                                   results[19]
                                print('PA='+str(PA)
                                   )
                                print('NA='+str(NA)
                                   )
                        wordcount+=1
                        erase()
                        pos=3
```

```
                        posx=375
                        move()
        if posx<100 or posx>=700:
                posx = lastposx
                pos=lastpos

        move()
        if wordcount<20:
                drawWord(wordcount)
        pygame.display.update()
```

Appendix D - The Activation-Deactivation Adjective Check List (AD ACL)

Originally from the appendix of (Thayer, 1989)

APPENDIX I

The Activation-Deactivation Adjective Check List (AD ACL)

The AD ACL is a multidimensional test of various transitory arousal states, including energetic and tense arousal (see Chapter 3). It has been used widely in many psychophysiological (e.g., Mackay, 1980) and psychological contexts,* and it has taken a variety of language forms (e.g., Bohlin & Kjellberg, 1973—Swedish version; Grzegolowska-Klarkowska, 1980—Polish version; Mackay et al., 1978—Anglicized version). Within the wider dimensions of energetic and tense arousal are four subscales—Energy (General Activation), Tiredness (Deactivation-Sleep), Tension (High Activation), and Calmness (General Deactivation).

The above parenthetical designations were given in the 1960s (Thayer, 1967), before the multidimensional arousal model was conceptualized in its present form. If these parenthetical names were to be modifed at the present time, they would be somewhat different. For example, the parenthetical name associated with Tiredness would probably now be *General Deactivation,* thus indicating that it is likely to represent the opposite pole from *General Activation.* Other names associated with Tension and Calmness might be High and Low Preparatory-Emergency Activation (or Arousal), thus indicating the likely function of these kinds of arousal.

The self-rating response format used in this test originally followed a format employed by Nowlis (1965) with the Mood Adjective Check List. This four-point self-rating system is slightly unconventional in comparison with the more usual three-, five-, or seven-point formats used in a number of other adjective checklists. Also, the verbal anchors of the AD ACL (as well as of the Mood Adjective Check List), although quite meaningful, are not completely symmetrical.

In order to determine if these somewhat unconventional features result in important differences, a study was recently completed to compare factor structures using different self-rating formats (Thayer, 1986). In this research, little difference was observed between the usual AD ACL format and others. Additional evidence for the validity of the AD ACL format may be found in other studies that employed it, and that obtained findings consistent with both mood and general arousal theories (Purcell, 1982; Watson & Tellegen, 1985). Therefore, the format most often employed with the AD ACL appears to be satisfactory. Alternatively, other more conventional

* In addition to studies reviewed in this book, see *Social Science Citation Index* with Thayer (1967, 1978a, 1986) as search references.

formats probably can be employed with little difference in results so long as the factor groupings are maintained.

Following is the AD ACL Short Form with the self-descriptive adjectives of Energy (A1), Tiredness (A2), Tension (B1), and Calmness (B2). Scoring is based on four possible points for each adjective. A common procedure in many studies has been to score only A1 and B1, since they are the best indications of energetic and tense arousal, respectively. A2 and B2 are particularly useful if the primary purpose of a study is to focus on the low arousal states of each dimension (Tiredness and Calmness). However, use of the full range of dimensions tends to reduce somewhat the strength of the relationships observed between arousal and other behaviors. This may be because people often do not make good discriminations of states of calmness, or it may occur because different processes underlie the pole opposites of each dimension (see Chapter 3).

AD ACL Short Form

Each of the words on the back describes feelings or mood. Please use the rating scale next to each word to describe your feelings *at this moment.*

EXAMPLES:

relaxed	(vv)	v	?	no	If you circle the double check (vv) it means that you *definitely feel* relaxed *at the moment.*
relaxed	vv	(v)	?	no	If you circle the single check (v) it means that you feel slightly relaxed *at the moment.*
relaxed	vv	v	(?)	no	If you circle the question mark (?) it means that the word does not apply or you cannot decide if you feel relaxed *at the moment.*
relaxed	vv	v	?	(no)	If you circle the no it means that you are *definitely not* relaxed *at the moment.*

Work rapidly, but please mark all the words. Your first reaction is best. This should take only a minute or two.

(Back page)

(vv)	v	?	no	:	definitely feel
vv	(v)	?	no	:	feel slightly
vv	v	(?)	no	:	cannot decide
vv	v	?	(no)	:	definitely do not feel

active	vv	v	?	no	drowsy	vv	v	?	no
placid	vv	v	?	no	fearful	vv	v	?	no
sleepy	vv	v	?	no	lively	vv	v	?	no
jittery	vv	v	?	no	still	vv	v	?	no
energetic	vv	v	?	no	wide-awake	vv	v	?	no
intense	vv	v	?	no	clutched-up	vv	v	?	no
calm	vv	v	?	no	quiet	vv	v	?	no
tired	vv	v	?	no	full-of-pep	vv	v	?	no
vigorous	vv	v	?	no	tense	vv	v	?	no
at-rest	vv	v	?	no	wakeful	vv	v	?	no

The AD ACL is scored by assigning 4, 3, 2, and 1, respectively to the "vv, v, ?," and "no" scale points, and summing or averaging the five scores for each subscale. (An appropriate cardboard template can be easily constructed.) In order of appearance, the subscale adjectives are as follows: Energetic (active, energetic, vigorous, lively, full-of-pep); Tired (sleepy, tired, drowsy, wide-awake, wakeful); Tension (jittery, intense, fearful, clutched-up, tense); Calmness (placid, calm, at-rest, still, quiet). Scoring for "wakeful" and "wide-awake" must be reversed for the Tiredness subscale. Also, if full bipolar dimensions of energetic and tense arousal are of interest (see above), Tiredness and Calmness scores must be reversed (but not wakeful and wide-awake, in this case) before summing the ten scores.

The AD ACL Long Form (Thayer, 1967, 1978a) includes additional activation adjectives as well as filler adjectives to disguise the purpose of the test. It contains the same instructions except that respondents are told that the test will take only a couple of minutes to complete. Based on previous analyses (Thayer, 1967, 1978a), the following adjectives are included on this form. The designations A1, A2, A3, and A4 after each significantly loaded activation adjective represent the subscales of Energy, Tiredness, Tension, and Calmness, respectively.

In order of appearance, the adjectives are: carefree, serious, peppy (A1), pleased, placid (A4), leisurely (A4), sleepy (A2), jittery (A3), intense (A3), grouchy, energetic (A1), egotistic, calm (A3, A4), suspicious, tired (A2), regretful, stirred-up (A3), warm-hearted, vigorous (A1), engaged-in-thought, at-rest (A4), elated, drowsy (A2), witty, anxious (A3), aroused, fearful (A3), lively (A1), defiant, still (A4), self-centered, wide-awake (A1, A2), skeptical, activated (A1), sad, full-of-pep (A1), affectionate, quiet (A4), concentrating, sluggish (A1, A2), overjoyed, quick (A1), nonchalant, quiescent (A4), clutched-up (A3), wakeful (A1, A2), rebellious, active (A1), blue, alert (A1), tense (A3). Since different numbers of activation adjectives are included in the four factors, these factor scores must be averaged instead of just summed if interfactor comparisons are to be made.

AD ACL GUI Code

```
import pygame
import time
import sys
from pygame import K_SPACE, K_w, K_s, K_a, K_d
from pygame.locals import *

def get_pygame_events():
        pygame_events = pygame.event.get()
        return pygame_events

wordcount = 0
st = 1
pygame.init()
sizex = 800
sizey = 600
screen = pygame.display.set_mode((sizex,sizey))
pos = 3
results = []

#colors
red = (255,0,0)
green = (0,255,0)
blue = (0,0,255)
black = (0,0,0)
white = (255,255,255)

myfont = pygame.font.SysFont("arial", 70)
myfont2 = pygame.font.SysFont("arial", 30)
alt1 = myfont2.render("Definitely",1,white)
alt12 = myfont2.render("do not feel",1,white)
alt2 = myfont2.render("Cannot decide",1,white)
alt3 = myfont2.render("Feel slightly",1,white)
alt4 = myfont2.render("Definitely feel",1,white)
screen.blit(alt1, (75, 370))
screen.blit(alt12, (65, 405))
screen.blit(alt2, (230, 370))
screen.blit(alt3, (430, 370))
screen.blit(alt4, (610, 370))


wordlist = ['Active', 'Placid', 'Sleepy','Jittery','
   Energetic','Intense','Calm','Tired','Vigorous','At-rest
   ','Drowsy','Fearful','Lively','Still','Wide-awake','
   Clutched-up','Quiet','Full-of-pep','Tense','Wakeful']
```

```
def words(counter):
        return wordlist[counter]

def drawWord(counter):
        pygame.draw.rect(screen,black,(0,0,800,200),0)
        word = myfont.render(wordlist[counter], 1, white)
        screen.blit(word, (320, 100))

count=0
for count in xrange(4): pygame.draw.rect(screen, white,
    (100+count*(133.33+50),300,50,50),2);count+=1


posx = 466.66
posy = 300

pygame.draw.line(screen,white,(posx,posy),(posx+50,posy+50)
    ,3)
pygame.draw.line(screen,white,(posx+50,posy),(posx,posy+50)
    ,3)

def move():
        pygame.draw.rect(screen,black,(lastposx+2,lastposy
            +2,47,47),0)
        pygame.draw.line(screen,white,(posx,posy),(posx+50,
            posy+50),3)
        pygame.draw.line(screen,white,(posx+50,posy),(posx,
            posy+50),3)

def erase():
        pygame.draw.rect(screen,black,(100+2,300+2,47,47)
            ,0)
        pygame.draw.rect(screen,black
            ,(283.33+2,300+2,47,47),0)
        pygame.draw.rect(screen,black
            ,(466.66+2,300+2,47,47),0)
        pygame.draw.rect(screen,black
            ,(649.99+2,300+2,47,47),0)

while wordcount<20:
        lastposx=posx
        lastposy=posy
        lastpos=pos
        keys_pressed = get_pygame_events()
```

```
for event in keys_pressed:
        if event.type == pygame.KEYDOWN:
                if event.key == K_a:
                        posx-=(133.33+50)
                        pos-=1
                if event.key == K_d:
                        posx+=133.33+50
                        pos+=1
                if event.key == K_SPACE:
                        result = pos
                        if wordcount == 14 or
                           wordcount == 19:
                                if result == 4:
                                        result = 1
                                elif result == 3:
                                        result = 2
                                elif result == 2:
                                        result = 3
                                elif result == 1:
                                        result = 4
                        results.append(result)
                        print(words(wordcount)+' '+
                           str(result))
                        if wordcount == 19:
                                A1 = results[0]+
                                   results[4]+
                                   results[8]+
                                   results[12]+
                                   results[17]
                                A2 = results[2]+
                                   results[7]+
                                   results[10]+
                                   results[14]+
                                   results[19]
                                B1 = results[3]+
                                   results[5]+
                                   results[11]+
                                   results[15]+
                                   results[18]
                                B2 = results[1]+
                                   results[6]+
                                   results[9]+
                                   results[13]+
                                   results[16]
                                print('A1='+str(A1)
```

```
                                                    )
                                                print('A2='+str(A2)
                                                    )
                                                print('B1='+str(B1)
                                                    )
                                                print('B2='+str(B2)
                                                    )
                                        wordcount+=1
                                        erase()
                                        pos = 3
                                        posx=466.66
                                        move()
        if posx<99 or posx>=700:
                posx = lastposx
                pos=lastpos

        move()
        if wordcount < 20:
                drawWord(wordcount)
        pygame.display.update()
```

Appendix E - Arduino Schematics

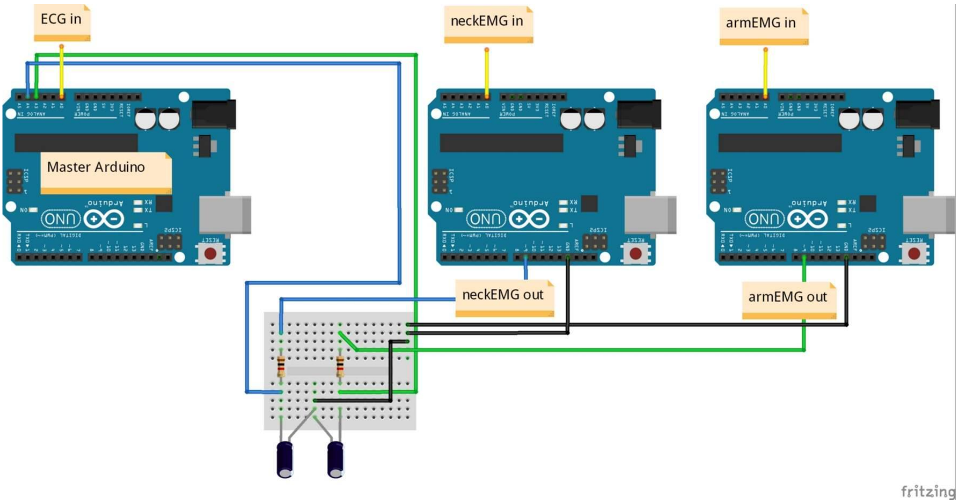


Figure 7.1: Schematics of biometric set up

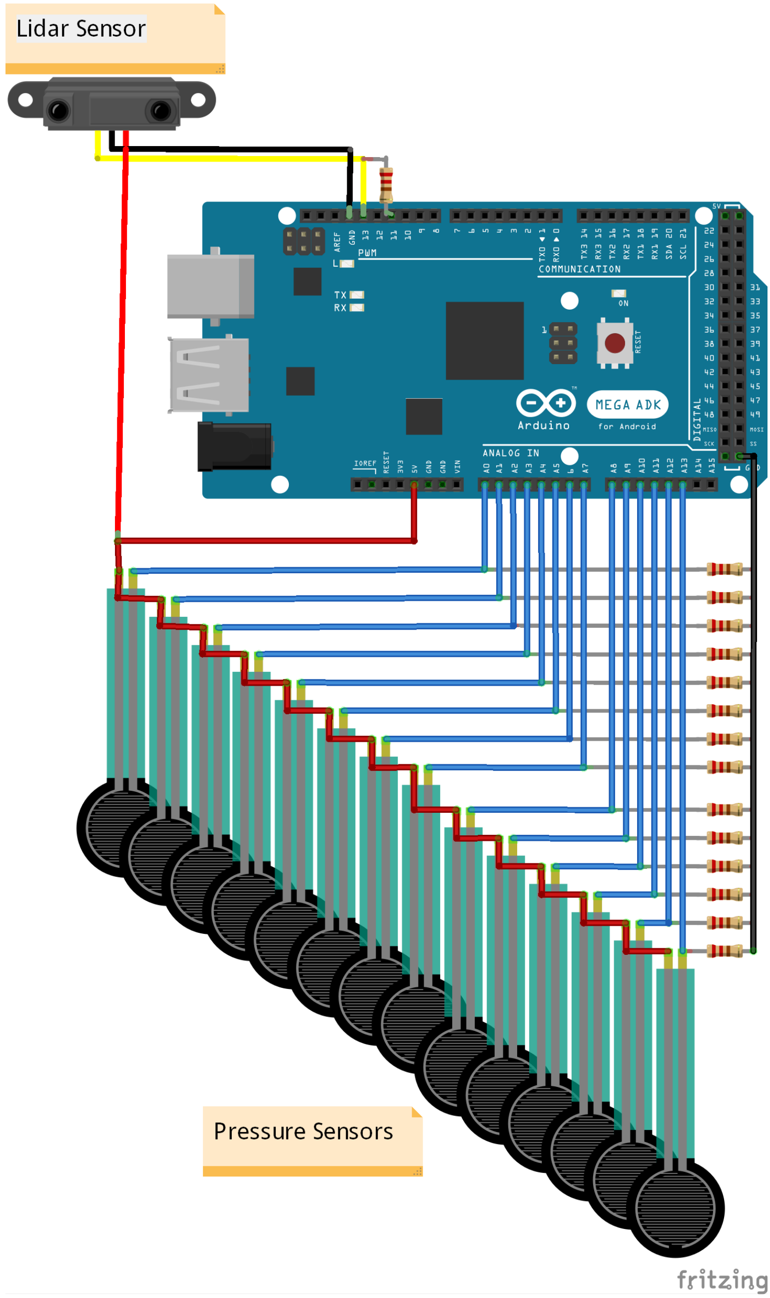


Figure 7.2: Schematics of chair set up

This page is be intentionally left blank.

Appendix F - Arduino Code

Arduino code for biometric sensors

```
eHealth sensor platform for Arduino and Raspberry from Cooking-hacks.

Description:  "The e-Health Sensor Shield allows Arduino and
Raspberry Pi users to perform biometric and medical applications
by using 9 different sensors:  Pulse and Oxygen in Blood Sensor
(SPO2), Airflow Sensor (Breathing), Body Temperature, Electrocardiogram
Sensor (ECG), Glucometer, Galvanic Skin Response Sensor (GSR
- Sweating), Blood Pressure (Sphygmomanometer) and Patient Position
(Accelerometer)."

In this example we read the values in volts of ECG sensor and
show these values in the serial monitor.

Copyright (C) 2012 Libelium Comunicaciones Distribuidas S.L.
http://www.libelium.com

This program is free software:  you can redistribute it and/or
modify it under the terms of the GNU General Public License as
published by the Free Software Foundation, either version 3 of
the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program.  If not, see <http://www.gnu.org/licenses/>.

Version 2.0
Author:  Luis Martin  Ahmad Saad Modified by Helge Soltvedt Garsmark


#include <eHealth.h>

byte serialByte;
float ECG;
int EMGneck;
int EMGarm;
int i = 0;
//double distance;
// The setup routine runs once when you press reset:
void setup() {
  Serial.begin(115200);
  pinMode(A1,INPUT);
```

```
  pinMode(A4,INPUT);
}

// The loop routine runs over and over again forever:
void loop() {

  while (Serial.available()>0){
    serialByte=Serial.read();
    if (serialByte=='C'){
      Serial.println("Time , ECG , EMGneck , EMGarm");
      while(1){
        EMGneck = analogRead(A4);
        EMGarm = analogRead(A1);
        ECG = eHealth.getECG();
        //distance = analogRead(A1);


        //Serial.print("ECG value :  ");
        Serial.print(millis());
        Serial.print(";");
        Serial.print(ECG,5);
        Serial.print(";");
        Serial.print(EMGneck);
        Serial.print(";");
        Serial.println(EMGarm);
        //Serial.print(" V");
        //Serial.println("");
        delay(1);  // wait for a millisecond
        if (Serial.available()>0){
          serialByte=Serial.read();
          if (serialByte=='r')   {Serial.print("Situation
              ");
          Serial.print(i); Serial.println(" end"); delay
              (50);}
          else if (serialByte=='t')  { i += 1;
          Serial.print("Situation "); Serial.print(i);
          Serial.println(" start"); delay(50); }
          else if (serialByte=='F')  break;
        }
      }
    }
  }
}
```

Arduino code for chair sensors

```
    //#include <CapacitiveSensor.h>

byte serialByte;
int i = 0;
long previousMillis = 0;
int numSensors = 14;
#define LtrigPin 12
#define LmonPin 11
double distance;
//long rightarm;
//long leftarm;

//CapacitiveSensor   cs_4_2 =
CapacitiveSensor(4,2); // 2 is sensor pin
//CapacitiveSensor   cs_4_6 =
CapacitiveSensor(4,6); //6 is sensor pin



void setup() {
  pinMode(LtrigPin, OUTPUT);
  digitalWrite(LtrigPin, LOW);
  pinMode(LmonPin, INPUT);

  Serial.begin(115200);
}


void loop() {

while (Serial.available()>0){
  serialByte=Serial.read();
  if (serialByte=='C'){
     while(1){
         if (millis()-previousMillis>199){
              String dataString = "";
              Serial.print(millis());
              Serial.print(";");
              //Pressure 1-16
              // read three sensors and append to the
                 string:
              for (int analogPin = 0; analogPin <
                 numSensors;
              analogPin++) {
```

```
                int sensor = analogRead(analogPin);
                Serial.print(sensor);
                Serial.print(";");
            }

            //rightarm =  cs_4_2.capacitiveSensor(60);
            //leftarm =  cs_4_6.capacitiveSensor(60);
            //Serial.print(rightarm);
            //Serial.print(";");
            //Serial.print(leftarm);
            //Serial.print(";");

            distance = pulseIn(LmonPin, HIGH);
            distance = distance/10;
            Serial.print(distance);
            Serial.println(";");

            previousMillis = millis();
            if (Serial.available()>0){
                serialByte=Serial.read();
                if (serialByte=='r') {
                  Serial.print("Situation ");
                  Serial.print(i);
                  Serial.println(" end");
                  delay(50);
                  }
                else if (serialByte=='t') {
                  i += 1;
                  Serial.print("Situation ");
                  Serial.print(i);
                  Serial.println(" start");
                  delay(50);
                  }
                else if (serialByte=='F') {
                  break;
                  }
            }
          }
      }
    }
  }
}
```

This page is be intentionally left blank.

Appendix G - Complete Dataset

Participant	Gender	Education	Age	Coffee	0Grid_x	0Grid_y	1EMGn_M	1EMGn_st	1EMGn_Va	1EMGn_M	1EMGn_(S	1EMGa_M
6	0	0	25	2	7	3	175.4591	3.083753	9.509535	175		207.8262
7	0	2	29	0	6	6	224.1257	2.629375	6.913612	224		229.1308
8	0	0	25	2	7	4	195.5588	2.781894	7.738934	195		196.8901
9	0	0	25	2	5	5	176.1089	2.284992	5.22119	176		191.9126
10	0	0	26	3	7	4	173.0589	2.34527	5.500289	173		190.7914
11	1	0	24	2	5	3	183.2389	2.714734	7.369779	183		228.4236
12	0	0	25	2	6	6	173.7692	2.361205	5.575288	174		212.9603
13	0	0	25	0	4	6	207.8031	2.586786	6.691461	208		203.7219
14	0	0	25	2	7	3	214.6144	2.989564	8.937492	214		215.4242
15	0	0	24	1	4	3	181.9698	2.805458	7.870597	182		211.9988
16	0	1	27	3	7	4	177.3393	2.936033	8.620287	177		204.7099
17	0	1	27	3	8	3	168.1937	2.720918	7.403397	168		198.9112
18	0	0	25	2	7	4	201.1221	2.865259	8.209708	201		213.7427
19	0	0	26	1	7	5	181.1925	2.609722	6.810648	181		194.466
20	0	0	25	2	5	5	167.8087	2.456662	6.03519	168		195.9458
21	0	0	25	2	7	4	176.979	3.639972	13.2494	177		212.3416
22	0	0	26	2	6	3	224.336	2.390642	5.715168	224		248.5631
23	0	0	27	2	6	5	224.2277	2.418467	5.848983	224		248.9014
24	1	0	26	1	7	4	213.9019	3.346579	11.19959	214		188.8192
25	0	0	26	3	5	5	172.9742	3.740395	13.99055	172		194.5568
26	0	0	26	2	4	2	183.5341	3.009155	9.055013	183		191.1929
27	0	1	25	1	5	5	576.3101	16.42362	269.7353	577		142.6393
28	0	2	26	0	7	7	189.5632	2.775958	7.705943	189		202.4218
29	1	0	24	1	6	4	202.6318	4.878639	23.80112	202		200.6659
30	0	2	25	3	6	6	176.1946	2.662227	7.087454	176		202.8263
31	0	0	28	2	7	3	185.3633	2.776762	7.710407	185		207.0207
32	1	0	25	1	3	4	179.3005	2.575687	6.634162	179		207.194
33	1	0	24	2	6	6	188.7848	2.666371	7.109534	189		203.1843
34	1	0	26	1	4	6	187.7064	3.312588	10.97324	187		193.2163
35	0	1	26	0	6	7	176.8357	2.484319	6.171841	177		190.7498
36	0	0	26	3	3	7	171.2247	2.456777	6.035754	171		190.293
37	0	0	24	2	6	4	209.3488	2.478581	6.143366	209		197.9149
38	0	0	24	2	6	7	174.2648	2.470446	6.103105	174		191.0804
39	1	0	25	0	5	7	195.2655	2.765131	7.645948	195		186.5437

1EMGa_std	1EMGa_Va	1EMGa_M	1EMGa_(Se	1ECG_Mea	1ECG_std	1ECG_Var	1ECG_Med	1ECG_RR n	1ECG_RR s	1ECG_HR n	1ECG_HR s	1ECG_RMS
2.347485	5.510688	208		1.80147	0.211122	0.044572	1.77908	874	120	70	10	160
2.391404	5.718814	229		1.786415	0.23473	0.055098	1.77908	754	50	80	5.3	34
2.275372	5.177318	197		1.821456	0.245093	0.060071	1.78397	788	36	76	4	30
2.247401	5.050813	192		1.803884	0.468902	0.219869	1.73998	742	36	81	5.9	34
2.551987	6.512638	191		1.816996	0.22911	0.052491	1.78397	695	53	87	6.7	40
2.488064	6.19046	228		1.817652	0.244115	0.059592	1.78397	640	27	94	502	11
2.693976	7.257508	213		1.807458	0.308152	0.094958	1.73021	739	26	82	6	20
3.326508	11.06566	203		1.839347	0.441141	0.194605	1.73998	715	22	84	2.9	19
3.103935	9.634411	215		1.823429	0.416377	0.17337	1.73998	696	29	87	9.3	22
3.023188	9.139663	212		1.757977	0.289309	0.083699	1.71065	753	44	80	7	29
4.644112	21.56778	205		1.836252	0.418971	0.175537	1.74976	560	25	110	6.8	16
2.498066	6.240334	199		1.813893	0.327482	0.107244	1.78397	762	34	81	14	37
2.371349	5.623294	214		1.824747	0.29065	0.084477	1.80841	749	38	81	9.6	34
2.783387	7.747243	194		1.817534	0.378757	0.143457	1.78397	1030	76	58	5.4	77
2.301034	5.29476	196		1.820135	0.464033	0.215326	1.75464	804	39	75	4.6	34
2.476978	6.13542	212		1.763002	0.55625	0.309414	1.78397	861	70	70	10	110
2.408668	5.80168	248		1.807736	0.228454	0.052191	1.78886	1100	65	55	5.5	84
3.622137	13.11988	248		1.816461	0.159643	0.025486	1.80841	990	71	61	5.3	61
3.097781	9.596248	188		1.824387	0.469722	0.220639	1.79374	650	16	92	2.7	12
3.331205	11.09693	194		4.234101	1.367629	1.870409	5					
3.290286	10.82598	191		1.818601	0.275295	0.075787	1.77419	764	72	80	10	95
10.66548	113.7525	143		1.829401	0.692654	0.479769	1.78886					
3.241692	10.50856	202		1.824931	0.174594	0.030483	1.80352	891	48	68	4.5	43
3.3419	11.1683	200		1.793475	0.17916	0.032098	1.77419	717	41	84	6.2	29
3.672974	13.49074	202		1.80389	0.167314	0.027994	1.78397	1050	56	58	3.6	62
3.655623	13.36358	206		1.828638	0.208584	0.043507	1.80841	827	51	73	7.2	31
3.36703	11.33689	207		1.807244	0.256201	0.065639	1.78886	656	23	92	9	15
2.683179	7.199451	203		1.818731	0.147686	0.021811	1.79863	728	18	83	3.6	16
3.026065	9.157067	193		1.824029	0.132264	0.017494	1.80841	681	24	88	4.1	17
2.50378	6.268915	190		1.823205	0.235575	0.055496	1.80352	827	46	73	5.1	41
2.467592	6.089011	190		1.820615	0.174424	0.030424	1.76931	675	45	90	7.6	29
2.625421	6.892837	198		1.815647	0.306089	0.09369	1.77419	559	25	110	7.4	19
2.553836	6.522079	191		1.808746	0.205774	0.042343	1.78886	868	42	69	5.2	49
2.920707	8.530527	186		1.803828	0.178491	0.031859	1.76442	632	39	96	9.6	34

1ECG_NN5	1ECG_PNN	1ECG_RR t	1ECG_TINN	1ECG_VLF	1ECG_LF p	1ECG_HF p	1ECG_VLF	1ECG_LF	1ECG_HF	1ECG_LF n	1ECG_HF n	1ECG_LF/H
192	57	22.6	364.3	260	1400	2670	6.1	32	62	32.5	69.9	0.52
32	8.2	10.6	156.3	240	1100	242	15	69	16	79.2	18.1	4.4
31	8.3	8.7	126.5	42	590	257	4.7	67	29	68.7	29.6	2.3
52	14	10.3	150.7	57	560	168	7.3	71	21	72.9	22	3.3
61	14	9.9	154.3	270	950	153	20	69	11	79.6	12.9	6.2
0	0	7.2	101.9	51	300	28	13	79	7.3	90.9	8.41	11
5	1.3	6.7	102.8	36	210	77.8	11	64	24	70.2	26.3	2.7
3	0.72	5.6	84.8	13	98	81	6.7	51	42	51.5	42.8	1.2
13	3	7.6	110.5	50	310	82.8	11	70	19	75.7	20.1	3.8
29	7.4	11.5	171.3	140	1000	222	9.8	74	16	81.1	17.4	4.7
6	1.1	5.9	87.1	19	190	112	5.9	59	35	61.2	36.5	1.7
71	18	8.1	124.5	28	290	141	6.1	63	31	51.9	25.5	2
54	14	9.6	140.7	70	410	332	8.6	51	41	53.5	43	1.2
138	47	16.2	241.8	600	1500	744	21	53	26	63.5	30.8	2.1
41	11	9.4	145	170	680	332	15	58	28	64.9	31.5	2.1
42	12	8.6	124.6	42	390	408	5	46	49	34.4	36	0.95
147	55	14.9	235.3	53	880	804	3	51	46	48.7	44.3	1.1
101	34	13.3	192.7	700	1900	404	23	63	13	81.6	17.2	4.8
0	0	4	62.3	14	95	20.3	11	74	16	77.7	16.6	4.7
82	21	11.8	164.9	100	770	525	7.5	55	38	50	34.1	1.5
80	24	12.7	189.1	170	650	299	15	58	27	66.7	30.6	2.2
32	7.8	10	156.9	110	430	174	16	60	24	69.5	28.4	2.5
134	48	14.1	222.6	77	620	651	5.7	46	48	46.5	48.7	0.96
37	10	11.1	184.9	320	760	119	27	64	9.9	85.3	13.3	6.4
2	0.44	6.2	84.1	89	190	49.5	27	57	15	75.8	20.2	3.7
0	0	5.1	77.3	27	82	11.1	22	69	9.2	67.9	9.15	7.4
2	0.46	6.4	97.9	56	220	49.2	17	68	15	76.3	16.7	4.6
77	22	11.5	185.4	120	770	328	10	63	27	68.2	29.1	2.3
36	8.2	13.3	208.7	100	810	212	9	72	19	77.4	20.3	3.8
17	3.2	6.8	94.2	31	160	139	9.6	48	42	50.4	44.8	1.1
117	34	10.6	169.3	98	170	380	15	26	59	28.1	63.4	0.44
68	15	10.2	155.5	180	250	413	21	30	49	36	58.5	0.61

1ECG_(Seri	1Lean_mea	1Lean_std.	1Lean_Var	1Lean_Median	1Lean_(Ser	1Chair_lea	1Chair_std	1Chair_Var	1Chair_me	1Chair_NO	1Chair_(Se	1Grid_x
	18.79173	2.394676	5.734471	18.9		4				1		8
	25.90531	2.269929	5.152579	26		4				1		4
	21.38431	1.666396	2.776875	21.5		2.993179				8		8
	14.74577	1.994813	3.979277	14.7		3				1		6
	17.25041	1.689469	2.854306	17.3		2.997948				56		8
	15.05404	1.846756	3.410508	15		3.000679				99		7
	14.98406	2.052178	4.211435	15		2.634063				27		8
	14.50918	2.166642	4.694338	14.6		1.99865				1		7
	16.90221	1.635644	2.675333	16.9		3.880936				15		8
	23.97316	2.509822	6.299204	24		2.010246				13		8
	17.06435	4.187832	17.53794	17		2.902473				11		3
	29.05619	3.114665	9.701136	29.1		3				3		7
	17.3739	3.592463	12.90579	17.4		3				1		7
	20.62162	2.020648	4.08302	20.6		4				1		7
	21.12114	2.537052	6.436633	21.1		2.00684				5		8
	15.55962	1.713085	2.934659	15.6		4				1		6
	30.09275	2.141719	4.58696	30		2.369788				7		7
	18.6201	3.247329	10.54515	18.5		2.734133				3		7
	19.0269	1.936453	3.749851	19		2				1		9
	20.52882	1.989331	3.957438	20.6		2.000684				1		8
	23.26516	2.020838	4.083786	23.4		3.989071				45		8
	18.89667	2.133429	4.55152	18.8		2.998639				1		7
	10.8638	2.030105	4.121327	10.9		4				1		7
	15.37988	2.662201	7.087316	15.4		3.57974				25		8
	23.17089	2.724746	7.424242	23.1		2.99863				1		7
	14.93712	2.567247	6.590755	14.9		4				1		7
	25.37173	2.371816	5.62551	25.3		2.605938				3		7
	13.11516	2.25058	5.065112	13.1		3.97541				3		6
	19.4663	2.898746	8.402727	19.5		1.859589				1		6
	22.03762	3.401109	11.56755	22.1		1.327633				41		6
	21.2111	2.878452	8.285483	21.2		2.993146				8		6
	16.58077	2.058476	4.237323	16.5		2.560575				92		7
	24.52909	2.179095	4.748456	24.6		2.009582				7		7
	15.80842	2.521446	6.357689	15.7		3				1		7

1Grid_y	1PANAS_P	1PANAS_N	1PANAS_(S	1ADACL_A	1ADACL_A	1ADACL_B	1ADACL_B	1ADACL_(S	2EMGn_M	2EMGn_st	2EMGn_Va	2EMGn_M
4	35	16		13	9	10	12		178.0733	3.735474	13.95377	178
4	26	26		9	7	12	14		232.7258	2.463123	6.066975	233
4	27	12		9	15	7	18		189.2702	2.984219	8.90556	189
4	26	15		9	15	9	12		178.4187	2.440922	5.958102	178
5	30	17		11	11	7	15		173.5731	2.563328	6.570652	173
3	31	19		12	6	11	11		179.9798	2.825535	7.983651	180
8	23	10		8	12	5	15		174.9109	4.479319	20.0643	174
3	25	12		7	16	12	14		206.4244	2.376984	5.650053	206
6	38	14		14	13	9	18		217.195	3.1062	9.648479	217
6	42	23		19	12	15	16		182.5212	4.257125	18.12311	182
3	36	17		14	6	12	11		178.8838	2.659987	7.075532	179
3	42	15		16	6	10	13		168.3822	2.415705	5.83563	168
2	43	18		14	7	11	12		200.0448	2.945469	8.675787	200
7	23	10		9	12	5	18		177.8548	2.542588	6.464753	178
5	30	16		10	8	8	17		169.0196	2.310644	5.339077	169
3	33	23		15	6	17	9		177.8646	2.756166	7.596453	178
4	38	16		12	7	9	15		224.243	2.403534	5.776975	224
5	33	16		12	12	12	19		224.0724	2.492829	6.214198	224
6	33	14		12	9	10	14		212.2108	4.295935	18.45506	212
2	44	15		16	6	14	11		171.1312	2.333968	5.447405	171
6	41	23		17	7	11	16		183.2451	2.477703	6.139013	183
3	41	18		17	5	18	15		616.4878	9.773267	95.51674	617
5	23	11		8	15	5	17		190.8471	2.747619	7.54941	191
5	35	18		12	10	15	12		208.4328	3.576342	12.79022	208
6	18	11		5	15	6	15		177.3061	6.234632	38.87063	177
4	28	18		10	14	6	17		188.4606	3.135319	9.830228	188
4	29	23		12	8	12	13		182.2818	2.853068	8.139997	182
5	26	11		12	14	10	16		187.2192	2.68832	7.227063	187
6	34	11		14	13	6	16		187.7403	4.73654	22.43482	187
3	36	37		12	18	8	13		177.2177	2.412619	5.820728	177
4	32	17		14	8	12	10		175.5179	3.250396	10.56508	175
4	26	13		13	7	7	16		209.1824	2.598241	6.750856	209
4	30	12		14	15	7	17		173.3419	2.438028	5.943979	173
6	19	17		8	19	6	19		194.5121	2.709343	7.340538	194

2EMGn_(S(	2EMGa_M(	2EMGa_st(	2EMGa_Va	2EMGa_M(	2EMGa_(S(	2ECG_Mea	2ECG_std	2ECG_Var	2ECG_Med	2ECG_RR n	2ECG_RR s	2ECG_HR n
	200.74	3.607037	13.01072	200		1.800811	0.173905	0.030243	1.77908	883	95	69
	214.3649	34.69471	1203.723	189		1.778239	0.344744	0.118849	1.79374	730	51	83
	199.0855	2.514406	6.322237	199		1.818187	0.309935	0.09606	1.77908	749	40	81
	195.3312	2.447294	5.989247	195		1.802019	0.456594	0.208478	1.74487	789	33	77
	185.9764	9.743795	94.94153	183		1.807828	0.212424	0.045124	1.77908	683	61	89
	222.7249	2.347721	5.511793	223		1.81813	0.227263	0.051648	1.77908	642	27	94
	213.1185	2.640751	6.973564	213		1.815167	0.319359	0.10199	1.73998	744	77	83
	203.1776	3.734047	13.94311	203		1.820826	0.45735	0.209169	1.73021	701	23	86
	218.3271	3.07125	9.432578	218		1.814919	0.423949	0.179733	1.73021	676	24	89
	211.7536	7.74396	59.96892	211		1.851585	0.307383	0.094485	1.80352	669	51	91
	207.5446	4.9601	24.60259	207		1.817896	0.319042	0.101788	1.77419	692	47	88
	202.8386	3.723001	13.86074	203		1.817237	0.287376	0.082585	1.78886	873	43	69
	218.3734	2.919529	8.523648	218		1.804388	0.435792	0.189915	1.82307	732	47	83
	198.6072	2.888291	8.342227	198		1.81382	0.394422	0.155569	1.76931	952	64	64
	199.2256	2.323796	5.400026	199		1.814079	0.45576	0.207717	1.74487	806	37	75
	214.0188	2.348353	5.514761	214		1.823188	0.420876	0.177137	1.79374	852	37	71
	247.8886	2.380136	5.665047	248		1.802162	0.257462	0.066286	1.78397	908	140	69
	248.2429	3.703666	13.71714	247		1.868854	0.609425	0.371398	1.79863	1000	160	65
	186.8809	3.005263	9.031605	186		1.812868	0.480893	0.231258	1.79863	653	21	92
	192.923	3.299519	10.88683	192		4.994429	0.01	0.0001	5			
	191.9274	3.145649	9.895109	191		1.817585	0.285574	0.081552	1.76931	647	120	97
	168.2691	6.6873	44.71998	168		1.823753	0.29193	0.085223	1.77908	903	48	67
	204.7113	3.764773	14.17351	204		1.816188	0.1857	0.034484	1.79863	825	34	73
	200.6236	3.321386	11.03161	200		1.83963	0.180119	0.032443	1.80841	783	43	77
	201.1748	3.634207	13.20746	201		1.803924	0.188938	0.035697	1.78886	963	50	63
	210.1144	9.823168	96.49462	207		1.823876	0.231025	0.053373	1.79374	653	29	94
	209.8332	4.622267	21.36535	209		1.812651	0.245068	0.060059	1.80352	732	38	83
	205.1535	3.155795	9.959042	205		1.781258	0.573391	0.328777	1.77908			
	194.8965	6.075883	36.91635	194		1.826053	0.163033	0.02658	1.82796	695	54	87
	189.5815	2.549925	6.502119	189		1.820267	0.242181	0.058652	1.79863	826	39	73
	189.2421	2.508697	6.29356	189		1.818366	0.165727	0.027465	1.77908	748	72	82
	201.2246	2.677877	7.171027	201		1.818414	0.314719	0.099048	1.77908	582	25	100
	189.0726	2.432685	5.917958	189		1.799065	0.572871	0.328181	1.80352			
	184.0355	2.726807	7.435476	184		1.808256	0.198336	0.039337	1.77908	636	37	95

2ECG_HR s	2ECG_RMS	2ECG_NN5	2ECG_PNN	2ECG_RR t	2ECG_TINN	2ECG_VLF	2ECG_LF p	2ECG_HF p	2ECG_VLF	2ECG_LF	2ECG_HF	2ECG_LF n
8.9	91	148	45	22	374.8	200	2000	1350	6.1	62	32	60.7
6	38	42	10	10.3	147.1	270	920	133	20	70	10	83.6
6.2	28	23	6.1	9.9	144.6	290	820	187	23	63	14	79.5
9.3	33	45	12	8.8	134.4	38	180	190	9.4	44	47	44.7
12	52	48	11	10.6	156.2	490	450	251	41	38	21	57
6.1	24	2	0.43	6	93.6	32	190	20.3	13	78	8.5	89.1
15	120	34	8.5	7.5	87.1	100	880	912	5.3	46	48	23.8
3.6	18	4	0.92	5.6	84.4	42	170	109	13	53	34	58
5.6	19	6	1.3	5.7	85.1	61	190	45.1	21	64	15	76.4
15	43	25	5.6	10.8	165.1	210	580	410	17	48	34	43.8
10	33	37	8.6	9.3	127.6	79	660	332	7.4	62	31	65.6
7	45	91	26	9.4	142.5	66	390	257	9.3	55	36	51.6
10	43	53	13	11.5	178.6	140	380	250	18	49	33	57.1
9.8	61	120	39	13.5	229.5	240	1300	641	11	60	30	63.5
5.2	36	75	20	9.6	153.2	160	340	223	22	47	31	56
3.7	32	38	11	9.3	139.3	110	330	103	20	61	19	67.9
15	86	151	46	23.3	410.1	5200	1600	587	70	22	8	61.3
26	200	147	50	20.9	309.3	3100	1700	1260	52	27	21	45.2
3.6	12	1	0.22	6.2	94.6	13	120	20.2	8.4	79	13	82.7
21	190	266	58	11.7	58.7	440	970	5170	6.7	15	79	9.96
4.9	41	42	13	12.7	198	210	740	128	20	69	12	83.5
7	30	35	9.7	9.4	147.4	120	420	151	17	61	22	70.9
11	42	28	7.3	10.1	149.2	120	290	158	21	51	28	61.1
8	48	89	29	11.8	183.3	110	500	409	11	49	40	52.8
14	17	7	1.5	7.2	101.4	29	310	73.2	7	75	18	79.2
9.6	30	12	2.9	7.4	109.3	28	260	128	6.6	63	31	62.2
9.1	29	30	7	13.9	215.6	470	1400	157	23	70	7.6	88.2
3.9	42	81	23	10.8	166.6	87	410	363	10	48	42	50.7
15	58	72	18	15.5	244.9	520	630	319	35	43	22	60.4
7.1	17	8	1.6	7.8	113.2	33	130	88.3	13	52	35	57.5
9.8	30	48	10	10.3	162.2	53	360	304	7.4	50	43	50.9

2ECG_HF n	2ECG_LF/H	2ECG_(Seri	2Lean_mea	2Lean_std.	2Lean_Var	2Lean_Me	2Lean_(Ser	2Chair_lea	2Chair_std	2Chair_Var	2Chair_me	2Chair_NO
31.5	1.9		21.06198	9.392937	88.22726	20.3		3.997245				3
12.2	6.9		19.60483	2.370361	5.618609	19.4		4				1
18.2	4.4		23.2114	1.956161	3.826566	23.1		3.866212				62
47.6	0.94		14.81842	2.144246	4.597791	14.8		3				1
31.5	1.8		17.94457	2.319273	5.379025	17.8		2.960355				17
9.73	9.2		14.16441	1.759214	3.094833	14.1		3.969945				36
24.6	0.96		15.66623	2.405884	5.788279	15.7		2.59877				43
37.4	1.6		13.3071	2.157305	4.653966	13.3		3.290984				96
18.1	4.2		14.31914	1.728425	2.987453	14.2		2.90499				43
30.9	1.4		24.84593	4.00075	16.006	24.6		2.262474				100
32.8	2		18.74856	4.19624	17.60843	18.7		2.754121				20
33.5	1.5		29.01701	3.137886	9.846329	29.15		2.999311				3
37.9	1.5		14.27031	3.968775	15.75117	14.2		3.755326				18
31.6	2		19.42315	2.31372	5.3533	19.4		2.131507				113
36.7	1.5		18.84177	2.489284	6.196534	18.8		2.010974				112
21.4	3.2		13.08584	2.006443	4.025812	12.95		1.943228				41
22.2	2.8		27.80389	2.665821	7.106604	28		2.368852				87
34.1	1.3		15.70549	19.07123	363.7118	13.1		2.921125				9
13.8	6		19.76277	1.973624	3.895191	19.7		2.066393				57
			19.54577	2.174871	4.730064	19.5		2.002732				3
53	0.19		20.06671	2.156788	4.651735	20.3		3.995899				38
14.4	5.8		17.86961	2.177209	4.74024	17.9		3.085558				51
25.4	2.8		11.91128	2.338972	5.470789	11.8		4				1
33.8	1.8		6.500616	1.730792	2.995643	6.5		3.705882				56
43.4	1.2		20.82726	2.525225	6.376764	20.8		3				1
19	4.2		13.74403	3.64097	13.25666	13.5		3.877816				95
30.3	2.1		22.3376	2.5281	6.391292	22.4		2.441781				12
			13.3286	3.099106	9.604458	13.4		2.28942				49
9.62	9.2		26.87529	5.044371	25.44568	27.1		2.661191				150
45.1	1.1		22.07447	3.011143	9.06698	22		2				1
30.4	2		19.90144	3.200959	10.24614	19.8		2.843943				61
38.7	1.5		13.62005	2.102839	4.421932	13.7		2.125257				36
			24.24668	2.279499	5.196115	24.3		2				186
43.4	1.2		17.75299	2.459736	6.050302	17.8		2.934111				178

2Chair_(Se	2Grid_x	2Grid_y	2PANAS_P	2PANAS_N	2PANAS_(S	2ADACL_A	2ADACL_A	2ADACL_B	2ADACL_B	2ADACL_(S	3EMGn_M	3EMGn_std
	5	1	34	34		15	5	17	7		195.2706	8.492629
	6	2	35	24		16	7	12	7		227.0746	2.375027
	3	1	41	29		20	5	19	5		185.0379	2.915785
	4	3	25	25		11	8	12	7		178.7595	2.369681
	3	3	26	26		13	13	13	8		178.9797	2.403896
	2	4	29	32		12	7	13	7		180.7468	4.031017
	4	2	37	17		14	7	11	5		173.9458	2.403479
	3	2	35	20		15	5	13	5		206.0375	2.363066
	3	3	25	28		15	7	15	7		218.1449	3.19131
	5	2	37	37		16	5	18	5		186.3913	3.985794
	1	2	27	29		15	5	17	7		175.124	2.38272
	3	3	26	26		15	6	15	6		167.7196	2.399986
	4	3	41	17		16	12	14	8		199.6938	2.545065
	6	3	33	10		14	13	6	14		179.2725	2.463371
	2	4	18	36		11	6	17	9		171.9926	2.783399
	5	1	33	35		17	5	17	6		178.157	2.440382
	4	3	28	23		13	6	15	7		224.0507	2.366809
	2	5	23	27		8	10	17	6		224.0448	2.406682
	2	3	32	28		14	6	17	6		212.926	3.392408
	2	1	19	18		12	6	13	10		171.8427	2.404402
	3	1	37	31		13	9	15	7		184.7937	2.49719
	4	1	48	31		19	6	18	5		667.935	9.810385
	2	3	33	25		15	11	17	7		190.0054	2.886415
	3	1	28	36		12	7	18	7		215.8209	3.426717
	3	3	26	29		10	7	15	5		176.0158	2.441747
	5	2	33	25		16	6	13	7		187.6794	3.115237
	8	2	35	31		17	6	15	8		182.0833	3.087749
	3	3	24	26		16	9	14	9		179.7302	10.19142
	3	2	32	16		15	5	12	5		184.8487	2.758289
	2	6	34	28		14	16	15	11		177.6022	2.379929
	3	3	21	26		12	10	17	5		174.1984	2.398496
	3	3	23	21		10	7	14	6		208.953	2.521688
	4	1	33	20		16	6	13	11		173.4142	2.38908
	4	4	33	27		16	13	16	13		196.02	2.757283

3EMGn_Va	3EMGn_M	3EMGn_(S	3EMGa_M	3EMGa_std	3EMGa_Va	3EMGa_M	3EMGa_(S	3ECG_Mea	3ECG_std	3ECG_Var	3ECG_Med	3ECG_RR n
72.12474	197		194.4515	3.247338	10.5452	194		1.801947	0.187734	0.035244	1.77908	923
5.640752	227		237.6041	2.498106	6.240532	237		1.80173	0.237569	0.056439	1.77908	719
8.501804	185		196.9247	2.296518	5.273997	197		1.805272	0.248746	0.061875	1.76931	825
5.615389	179		192.5975	2.229417	4.970299	192		1.803575	0.463038	0.214404	1.73998	752
5.778717	179		233.8733	2.973344	8.840775	234		1.809227	0.20492	0.041992	1.77908	753
16.24909	180		221.9962	2.38476	5.687079	222		1.814307	0.223374	0.049896	1.77419	677
5.77671	174		214.8659	2.614158	6.833823	215		1.814358	0.322324	0.103893	1.74487	800
5.584081	206		205.2285	2.642002	6.980176	205		1.822765	0.451249	0.203626	1.72532	729
10.18446	218		223.7797	3.323112	11.04307	223		1.813143	0.425533	0.181079	1.73021	702
15.88655	186		212.0122	4.383452	19.21465	211		1.828805	0.298736	0.089243	1.76931	641
5.677354	175		204.4544	2.806603	7.877021	204		1.814467	0.312309	0.097537	1.76931	651
5.759931	168		200.3423	4.60357	21.19286	199		1.81783	0.295549	0.087349	1.78397	913
6.477357	200		218.1087	2.543337	6.468563	218		1.827521	0.283369	0.080298	1.80352	770
6.068195	179		197.1227	2.716175	7.377606	197		1.815025	0.393523	0.15486	1.77419	972
7.747308	172		198.0022	2.315492	5.361505	198		1.810932	0.463704	0.215022	1.74487	815
5.955467	178		213.4894	2.422185	5.866978	213		1.815989	0.400663	0.160531	1.77908	842
5.601785	224		245.7692	2.333843	5.446821	246		1.819609	0.224622	0.050455	1.81329	1080
5.79212	224		247.9817	3.623975	13.1332	247		1.811256	0.135496	0.018359	1.80841	1000
11.50843	213		188.116	3.134419	9.824584	187		1.813932	0.458951	0.210636	1.77908	643
5.78115	172		225.056	3.786023	14.33397	225		4.627688	1.147998	1.317899	5	
6.235958	185		192.0769	3.157457	9.969534	191		1.81528	0.275667	0.075992	1.75953	740
96.24366	668		203.3563	6.589503	43.42155	203		1.816421	0.276871	0.076658	1.77419	888
8.331389	190		201.2553	3.312831	10.97485	200		1.821177	0.184589	0.034073	1.80841	828
11.74239	216		201.6696	3.297754	10.87518	201		1.827342	0.169988	0.028896	1.79863	727
5.962128	176		202.6757	3.328411	11.07832	202		1.80319	0.175423	0.030773	1.78397	939
9.704705	187		204.562	3.757526	14.119	204		1.811419	0.213381	0.045531	1.78397	738
9.534193	182		209.2887	4.175677	17.43628	208		1.814898	0.231272	0.053487	1.79863	752
103.865	178		203.5511	2.657569	7.062674	203		1.826843	0.160831	0.025867	1.80352	725
7.608158	185		194.2967	4.121494	16.98671	194		1.829065	0.135791	0.018439	1.81329	683
5.66406	177		189.8013	2.488676	6.193509	190		1.818139	0.243782	0.05943	1.79374	794
5.752785	174		192.3178	2.553156	6.518605	192		1.812652	0.170502	0.029071	1.77419	786
6.358913	209		199.5341	2.774668	7.698785	199		1.819859	0.318989	0.101754	1.77908	603
5.707705	173		188.624	2.438269	5.945156	188		1.817374	0.213096	0.04541	1.78886	893
7.602609	196		184.1234	2.486958	6.184962	184		1.807834	0.183123	0.033534	1.76931	741

3ECG_RR s	3ECG_HR n	3ECG_HR s	3ECG_RMS	3ECG_NN5	3ECG_PNN	3ECG_RR t	3ECG_TINN	3ECG_VLF	3ECG_LF p	3ECG_HF p	3ECG_VLF	3ECG_LF
110	66	11	140	153	48	17.7	264.5	660	2500	1750	13	51
40	84	5.2	26	26	6.3	9.6	143.3	70	670	118	8.1	78
34	73	3.5	31	37	10	8.2	122	110	310	259	16	45
32	80	6.5	36	61	16	8.7	137.9	21	210	126	5.9	59
44	80	5.8	38	67	17	10.7	157.7	310	370	136	38	45
33	89	4.2	41	2	0.45	5.5	81.4	44	100	40.6	23	55
100	77	15	160	52	14	9.9	145	190	1000	1710	6.5	35
19	82	2.5	17	0	0	4.8	74.9	22	120	41.4	12	66
31	86	6.4	23	11	2.6	8.1	120.4	54	360	141	9.7	65
28	94	7	15	1	0.22	9	132.8	68	200	51.1	22	62
24	93	6.1	17	3	0.66	5.8	86	9.1	240	84.7	2.7	72
46	66	5.5	47	100	31	11.7	184.9	410	310	242	43	32
32	78	5	28	27	7	9.6	135.1	120	190	153	27	40
60	63	8.4	58	94	30	15.5	228.7	670	1000	355	33	49
33	74	3.3	33	39	11	7.9	119.9	40	360	172	7	63
35	71	4.1	31	21	6	10.1	153.9	540	220	51.9	66	27
72	56	5.9	62	108	39	12.5	179.1	1300	1900	294	37	55
64	60	5.3	46	78	27	14	205.4	280	1300	287	15	70
15	94	3.9	10	0	0	4.3	65.4	8.9	100	21	6.8	77
43	82	8.3	35	43	11	9.1	141.7	160	600	225	16	61
50	68	7	32	35	11	11.4	164.8	460	1000	120	29	64
49	73	8.4	48	59	17	11.9	186.2	98	490	242	12	59
31	83	7.4	25	15	3.7	9.4	139.9	160	180	114	35	40
58	64	4.5	50	106	34	13.2	203.9	370	880	397	22	54
40	82	8	25	13	3.3	12.1	181.4	170	540	92.8	22	67
49	80	7.4	63	23	5.7	7.7	113.3	84	270	95.7	18	60
20	83	4.4	15	0	0	6	89.2	140	140	12.7	47	48
24	88	3.9	18	3	0.69	6.2	91.4	35	170	19.7	16	76
35	76	3.5	31	41	11	9.3	147	51	400	176	8.2	64
38	77	5.5	33	41	11	10.9	164.3	30	390	145	5.4	69
28	100	8.2	17	4	0.82	6.5	95.2	36	270	89.6	9.1	68
44	67	3.8	56	151	46	9.7	162.4	100	230	489	13	28
50	82	7.3	51	131	32	13.5	212.8	280	370	993	17	23

3ECG_HF	3ECG_LF n	3ECG_HF n	3ECG_LF/H	3ECG_(Seri	3Lean_mea	3Lean_std.	3Lean_Var	3Lean_Me	3Lean_(Ser	3Chair_lea	3Chair_std	3Chair_Var
36	48.1	34	1.4		26.66535	8.949685	80.09687	31.2		3.999321		
14	82.3	14.5	5.7		20.46549	1.932119	3.733084	20.5		4		
38	52.3	44.3	1.2		21.21522	1.744071	3.041782	21.3		3.448464		
35	52.8	31.2	1.7		14.43814	2.010372	4.041594	14.3		3		
17	61.4	22.7	2.7		16.88257	2.321489	5.389313	16.8		2.989233		
21	70.5	27.4	2.6		12.53667	1.979994	3.920377	12.5		3.983673		
59	19.8	33.6	0.59		15.09097	1.955907	3.825572	15.2		2.361396		
22	69.8	23.8	2.9		13.48141	2.028042	4.112956	13.5		2.77512		
25	69.6	27	2.6		16.28463	1.87482	3.514948	16.3		2.290301		
16	78.5	20.5	3.8		32.0565	7.068966	49.97028	28.4		2.696306		
25	72.1	25.2	2.9		19.62826	4.133053	17.08212	19.7		2.100823		
25	49	37.8	1.3		26.15821	3.349064	11.21623	26.1		3		
33	52.8	43.1	1.2		18.0345	3.651846	13.33598	17.8		3.317558		
17	68.8	24.3	2.8		18.49966	2.229234	4.969482	18.5		3.377688		
30	63.6	30.7	2.1		23.72454	3.888145	15.11767	23.7		4		
6.4	68	16.1	4.2		14.91394	1.963741	3.856279	14.7		2.064252		
8.5	81.3	12.6	6.4		27.76658	2.509568	6.297933	28.1		2.005468		
15	81.5	17.5	4.7		17.49774	3.61761	13.0871	17.5		2.455731		
16	80.1	16.6	4.8		19.48089	2.117002	4.481696	19.5		2		
					19.4224	2.22517	4.951382	19.45		2		
23	67.9	25.6	2.6		21.19381	2.05858	4.237752	21.4		3.806934		
7.5	88.5	10.4	8.5		19.69932	2.31556	5.361818	19.6		1.695147		
29	63.7	31.3	2		12.08317	2.333393	5.444724	12.1		4		
25	58.5	37.2	1.6		12.17591	2.585413	6.684361	12.1		3		
24	66.9	30.1	2.2		22.1987	2.362571	5.581741	22.1		3		
12	83.8	14.4	5.8		10.55915	3.00338	9.020292	10.6		3.391364		
21	58	20.3	2.9		23.80958	2.532123	6.411646	23.7		2.890561		
4.3	79.8	7.04	11		16.36048	2.986659	8.92013	16.4		2.325342		
8.7	79	9.13	8.7		29.9132	6.050189	36.60478	30.1		1.995896		
28	67.4	29.6	2.3		22.66123	2.225045	4.950826	22.7		2.00137		
26	67	25	2.7		20.16587	2.727492	7.439211	20.1		2.999316		
23	72.5	24.2	3		13.12428	2.018117	4.072796	13.3		1.088235		
59	29.8	63	0.47		23.66932	2.61321	6.828867	23.8		2		
60	26.3	69.9	0.38		17.49849	3.202185	10.25399	17.4		2.995885		

3Chair_me	3Chair_NO	3Chair_(Se	3Grid_x	3Grid_y	3PANAS_P/	3PANAS_N	3PANAS_(S	3ADACL_A:	3ADACL_A:	3ADACL_B:	3ADACL_B:	3ADACL_(S
	1		8	1	41	16		14	7	12	8	
	1		6	3	31	15		13	7	11	13	
	22		7	2	37	15		16	7	12	12	
	1		6	3	28	14		12	8	8	10	
	3		6	4	30	24		13	11	13	9	
	5		6	4	35	26		13	6	11	12	
	61		6	4	24	10		12	7	7	8	
	187		7	3	36	12		16	6	11	5	
	53		6	3	36	19		17	6	15	7	
	106		8	2	47	21		18	5	15	11	
	27		7	3	34	14		14	6	14	10	
	25		7	3	39	19		16	5	15	10	
	1		7	3	47	16		17	7	14	9	
	3		5	3	35	12		13	14	6	15	
	3		6	3	38	13		13	7	7	14	
	88		8	3	46	18		19	5	8	9	
	22		7	3	39	16		15	7	15	8	
	7		7	3	35	13		12	7	12	13	
	129		8	2	41	19		16	5	15	6	
	1		9	1	42	11		17	5	13	6	
	69		7	2	40	22		16	8	14	10	
	36		9	2	48	10		19	6	11	7	
	3		5	3	37	16		15	9	14	9	
	4		8	2	40	20		16	7	16	8	
	1		6	4	30	11		14	7	5	9	
	34		8	2	34	15		17	6	11	6	
	7		8	4	40	24		18	8	14	9	
	37		5	4	25	16		15	9	12	12	
	3		7	3	35	12		16	5	11	7	
	3		7	3	40	25		14	13	14	10	
	23		6	3	33	21		14	10	13	6	
	19		4	2	31	22		15	7	15	5	
	28		8	1	39	16		19	7	15	8	
	5		8	4	39	13		16	7	11	16	

EMGnMax	EMGnMin	1EMGnNor	2EMGnNor	3EMGnNor	EMGaMax	EMGaMin	1EMGaNor	2EMGaNor	3EMGaNor	1Grid_xNo	2Grid_xNo	3Grid_xNo
267	163	0.119799	0.144936	0.310294	239	185	0.422707	0.291481	0.175028	0.875	0.5	0.875
246	214	0.316427	0.585181	0.408581	275	172	0.554668	0.411309	0.636933	0.375	0.625	0.625
212	175	0.555643	0.385681	0.271293	217	189	0.281788	0.360196	0.283026	0.875	0.25	0.75
193	167	0.350341	0.43918	0.452288	211	183	0.318305	0.440398	0.342768	0.625	0.375	0.625
192	163	0.346858	0.364591	0.551024	251	175	0.207781	0.144426	0.774648	0.875	0.25	0.625
239	167	0.225541	0.180276	0.190928	243	214	0.497365	0.30086	0.275732	0.75	0.125	0.625
257	165	0.095317	0.107727	0.097237	231	203	0.355726	0.361373	0.423782	0.875	0.375	0.625
230	199	0.28397	0.239497	0.227015	244	191	0.240037	0.229767	0.268463	0.75	0.25	0.75
239	203	0.322622	0.394306	0.42069	257	207	0.168483	0.226543	0.335593	0.875	0.25	0.625
219	171	0.228538	0.240025	0.320652	230	198	0.437463	0.4298	0.437881	0.875	0.5	0.875
198	167	0.333526	0.383347	0.262065	241	190	0.288429	0.344011	0.283419	0.25	0	0.75
185	159	0.353606	0.360852	0.335369	240	189	0.194338	0.271346	0.222398	0.75	0.25	0.75
224	190	0.327121	0.295434	0.285113	236	206	0.258089	0.412447	0.403624	0.75	0.375	0.75
202	167	0.405501	0.310138	0.350644	231	187	0.169681	0.2638	0.230061	0.75	0.625	0.5
199	159	0.220217	0.250489	0.324814	222	189	0.210478	0.309866	0.272793	0.875	0.125	0.625
230	167	0.158396	0.172454	0.177095	239	202	0.279503	0.324833	0.310525	0.625	0.5	0.875
242	215	0.345777	0.342332	0.335212	265	239	0.367812	0.341871	0.260356	0.75	0.375	0.75
239	215	0.384487	0.378015	0.376868	267	238	0.37591	0.353205	0.344196	0.75	0.125	0.75
242	198	0.361408	0.322974	0.339226	207	178	0.373074	0.306238	0.348828	1	0.125	0.875
195	163	0.311693	0.254098	0.276335	243	183	0.192614	0.165383	0.700933	0.875	0.125	1
213	175	0.22458	0.216975	0.25773	213	180	0.33918	0.361437	0.365967	0.875	0.25	0.75
708	5.6911	0.81249	0.869698	0.942952	577	102	0.085556	0.139514	0.213382	0.75	0.375	1
206	179	0.391231	0.43878	0.407607	224	191	0.346116	0.415495	0.310767	0.75	0.125	0.5
233	191	0.276949	0.415066	0.590974	220	191	0.333306	0.331849	0.367917	0.875	0.25	0.875
214	166	0.212388	0.235544	0.208662	240	189	0.271104	0.238721	0.26815	0.75	0.25	0.625
220	175	0.230296	0.299125	0.281764	247	194	0.245673	0.304045	0.199283	0.75	0.5	0.875
204	171	0.25153	0.341872	0.335857	261	197	0.159282	0.200519	0.192011	0.75	0.875	0.875
233	163	0.368355	0.345988	0.239003	223	193	0.339476	0.405117	0.351703	0.625	0.25	0.5
231	172	0.26621	0.266784	0.217774	278	183	0.10754	0.125226	0.118913	0.625	0.25	0.75
193	167	0.378296	0.392988	0.407777	208	182	0.336529	0.291596	0.300049	0.625	0.125	0.75
230	163	0.122756	0.186834	0.16714	208	181	0.344185	0.305263	0.419179	0.625	0.25	0.625
225	199	0.398032	0.391631	0.382809	216	190	0.304418	0.431715	0.366697	0.75	0.25	0.375
191	165	0.356339	0.320844	0.323624	207	181	0.387709	0.310483	0.293231	0.75	0.375	0.875
213	183	0.40885	0.383736	0.433999	203	175	0.412274	0.322697	0.325837	0.75	0.375	0.875

1ADACL_tc	2ADACL_tc	3ADACL_tot
2	20	11
0	14	4
-17	29	9
-9	8	2
-8	5	6
6	11	6
-14	13	4
-11	18	16
-8	16	19
6	24	17
9	20	12
7	18	16
6	10	15
-16	-7	-10
-7	13	-1
17	23	13
-1	15	15
-7	9	4
-1	19	20
13	9	19
5	12	12
15	26	17
-19	14	11
5	16	17
-19	13	3
-15	16	16
3	18	15
-8	12	6
-9	17	15
-11	2	5
8	14	11
-3	11	18
-11	12	19
-24	6	4

Appendix B - TETRIS CODE

TETRIS SCENARIO 1

```
#!/usr/bin/env python

# PyTris (c) Lukasz Grzegorz Maciak
# Licensed under GNU General Public License Version 3

import sys, pygame, random
from pygame.locals import *
from pygame import K_q
import time

img = pygame.image.load('C:/Users/erikm/Google Drive/My
   Documents/Skoleting/Master/Scenarios/1st Prototype 27
   apr/Experiment/Pytris-master/cat.png')
pygame.mixer.pre_init(44100,-16,2,2048)
pygame.mixer.init()
pygame.init()

pygame.mixer.music.load('C:/Users/erikm/Google Drive/My
   Documents/Skoleting/Master/Scenarios/1st Prototype 27
   apr/Experiment/Sound/Relaxing.mp3')
positivesound = pygame.mixer.Sound('C:/Users/erikm/Google
   Drive/My Documents/Skoleting/Master/Scenarios/1st
   Prototype 27 apr/Experiment/Sound/levelup.wav')
positivesound.set_volume(0.05)
pygame.mixer.music.play()
pygame.mixer.music.set_volume(0.3)

# useful constants
size = width, height = 800, 600
lines_per_level = 200           #increas level after x
   lines
delay = 800                     #speed
mode = False                            #for switching
   directions
counter = 0             #feedback bar counter
st = 1
bitnumber = 0
lastlinecount = 0

# define colors
black = 0, 0, 0
red = 255, 0, 0
green = 0, 255, 0
```

```
blue = 0, 0, 255
white = 255, 255, 255
yellow = 255, 255, 0
purple = 160, 32, 240
cyan = 0, 255, 255
orange = 255, 165, 0
gray =  45, 45, 45

currentcolor = red      #for feedback bar
myfont = pygame.font.Font(None, 24)
timeFont = pygame.font.Font(None, 24)
perf = myfont.render("PERFORMANCE:", 1, white)
myfont2 = pygame.font.Font(None, 14)
timeshow = timeFont.render("TIME:", 1, white)
startTime = time.time()
lastTime = time.time()
countdown = 300

block_size = 15         # each piece is 4 blocks
block_gap = 1           # distance between blocks

offset = block_size + block_gap

twidth = 10 * offset
right_edge = 125+twidth - block_size

theight = 24 * offset
bottom_edge = 100+theight - block_size

start_point = 125+twidth/2 - block_size, 100
next_point = 125+twidth+125, 100+100

# Block object exists mostly to allow us to break a piece
   into individual
# components once it is locked in place. This is why the
   color information
# is redundant
class Block(object):
        """ Represents a rectangular Tetris block """

        def __init__(self, x, y, color):

                self.x = x
                self.y = y
                self.color = color
```

```
        def draw(self):

                pygame.draw.rect(screen, self.color, self.
                    rect)

        @property
        def rect(self):
                return (self.x, self.y, block_size,
                    block_size)

class Piece(object):
        """ A Tetris Piece - composed of 4 blocks """

        def __init__(self,x,y):

                self.x = x
                self.y = y
                self.mobile = True

                self.rotation = 0 # defines which of the
                    members of self.positions to use for
                    this object

                # set of offsets that is applied to
                    coordinates of each block in this piece
                self.positions = None


        def get_blocks(self):
                """ Returns an array of 4 blocks which make
                     up this piece; each block has it's own
                    coordinates and draw function """

                blocks = []

                for i in range(4):
                        blocks.append( Block(self.x + self.
                            positions[self.rotation][i][0],
                            self.y + self.positions[self.
                            rotation][i][1], self.color) )

                return blocks

        def draw(self):
```

```
		""" Draw all the blocks of this piece to
		    the screen """

		blocks = self.get_blocks()

		for b in blocks:
			pygame.draw.rect(screen, self.color
			    , b.rect)

	# calling flip repeatedly cycles through all
	   available positions and goes back to the first
	   one
	def flip(self):
		if(self.rotation < len(self.positions) -1):
			self.rotation += 1
		else:
			self.rotation = 0

	def set_point(self, x, y):
		self.x = x
		self.y = y


# Define different piece types: Z, S, O, T, I, L, J

class ZPiece(Piece):

	def __init__(self,x,y):

		super(ZPiece, self).__init__(x, y)

		self.color = red

		self.positions =	[
						( (0,0), (
						   offset,
						   0), (
						   offset,
						   offset),
						    (2*
						   offset,
						   offset)
						       ),
						( (0,0),
						    (0,
```

```
                                                    offset),
                                                     (-
                                                    offset,
                                                    offset),
                                                     (-
                                                    offset,
                                                    2*offset
                                                    )    )
                                           ]
class SPiece(Piece):

        def __init__(self,x,y):

                super(SPiece, self).__init__(x, y)

                self.color = green

                self.positions =        [
                                                 ( (0,0), (-
                                                    offset,
                                                    0), (-
                                                    offset,
                                                    offset),
                                                     (-2*
                                                    offset,
                                                    offset)
                                                      ),
                                                 ( (0,0),
                                                     (0,
                                                    offset),
                                                     (offset
                                                    , offset
                                                    ), (
                                                    offset,
                                                    2*offset
                                                    )      )
                                           ]


class OPiece(Piece):

        def __init__(self, x, y):

                super(OPiece, self).__init__(x,y)
```

```
        self.color = yellow

        self.positions =        [
                                        ( (0,0), (
                                          offset,
                                          0), (
                                          offset,
                                          offset),
                                           (0,
                                          offset)
                                              )
                                ]


class TPiece(Piece):

    def __init__(self,x,y):

        super(TPiece, self).__init__(x, y)

        self.color = purple

        self.positions =        [
                                        ( (0,0),
                                           (0,
                                          offset),
                                           (-
                                          offset,
                                          offset),
                                           (offset
                                          , offset
                                          )
                                          ),
                                        ( (0,0),
                                           (0,
                                          offset),
                                           (offset
                                          , offset
                                          ), (0,
                                          2*offset
                                          )   ),
                                        ( (0,0), (-
                                          offset,
                                          0), (
                                          offset,
```

```
                                                      0), (0,
                                                      offset)

                                                      ),
                                                   ( (0,0),
                                                      (0,
                                                      offset),
                                                       (-
                                                      offset,
                                                      offset),
                                                       (0, 2*
                                                      offset)
                                                       ),
                                        ]


class IPiece(Piece):

        def __init__(self,x,y):

                super(IPiece, self).__init__(x, y)

                self.color = cyan

                self.positions =        [
                                                   ( (0,0),
                                                      (0,
                                                      offset),
                                                       (0, 2*
                                                      offset),
                                                       (0, 3*
                                                      offset)
                                                           ),
                                                   ( (0,0), (
                                                      offset,
                                                      0), (2*
                                                      offset,
                                                      0), (3*
                                                      offset,
                                                      0)
                                                      )
                                        ]


class LPiece(Piece):
```

```
def __init__(self,x,y):

        super(LPiece, self).__init__(x, y)

        self.color = orange

        self.positions =        [
                                        ( (0,0),
                                           (0,
                                          offset),
                                            (0, 2*
                                          offset),
                                            (offset
                                          , 2*
                                          offset)

                                          ),
                                        ( (0,0),
                                           (0,
                                          offset),
                                            (offset
                                          , 0),
                                           (2*
                                          offset,
                                          0)
                                                 )
                                          ,
                                        ( (0,0), (-
                                          offset,
                                          0), (0,
                                          offset),
                                            (0, 2*
                                          offset)
                                                ),
                                        ( (0,0),
                                           (0,
                                          offset),
                                            (-
                                          offset,
                                          offset),
                                            (-2*
                                          offset,
                                          offset)
                                             ),
```

```
]

class JPiece(Piece):

        def __init__(self,x,y):

                super(JPiece, self).__init__(x, y)

                self.color = blue

                self.positions =        [
                                                ( (0,0),
                                                   (0,
                                                  offset),
                                                   (0, 2*
                                                  offset),
                                                   (-
                                                  offset,
                                                  2*offset
                                                  )
                                                        )
                                                  ,
                                                ( (0,0), (
                                                  offset,
                                                  0), (2*
                                                  offset,
                                                  0), (2*
                                                  offset,
                                                  offset)

                                                  ),
                                                ( (0,0), (
                                                  offset,
                                                  0), (0,
                                                  offset),
                                                   (0, 2*
                                                  offset)
                                                        )
                                                  ,
                                                ( (0,0),
                                                   (0,
                                                  offset),
                                                   (offset
                                                  , offset
                                                  ), (2*
```

```
                                                    offset,
                                                    offset)
                                                        ),
                                         ]



class Grid(object):

        def __init__(self):

                self.current = None
                self.next = None
                self.blocks = []

                self.next_piece()

                self.total_cleared_lines = 0
                self.cleared_lines = 0
                self.level = 1
                self.score = 0

                self.lines_til_next_level = lines_per_level

                self.delay = delay

                self.game_over = False

                self.next_rect = pygame.Rect(150+125,
                   90+100, 300, 300)

                # this is for multiplying scores
                self.multiplier =       {
                                                0 : 0,

                                                   # no
                                                   lines
                                                   cleared
                                                1 : 40,

                                                   # single
                                                    line
                                                   cleared
                                                2 : 100,
```

```
                                                        # two
                                                        lines
                                                        cleared
                                                    3 : 300,

                                                        # three
                                                        lines
                                                        cleared
                                                    4 : 1200

                                                        # TETRIS
                                            }

        self.nfont = pygame.font.Font(None, 24)
        self.largefont = pygame.font.Font(None, 40)
        self.smallfont = pygame.font.Font(None, 14)


    def random_piece(self):
        rekkefolge
            =[0,3,1,2,0,0,1,1,1,0,0,3,0,0,2,3,0,0,2,1,1,2,3,1,2,

        pcs =   {
                        #0 : ZPiece(*next_point),
                        0 : OPiece(*next_point),
                        #2 : SPiece(*next_point),
                        #1 : TPiece(*next_point),
                        1 : IPiece(*next_point),
                        2 : LPiece(*next_point),
                        3 : JPiece(*next_point)
                }
        global bitnumber
        c = rekkefolge[bitnumber]
        bitnumber+=1
        return pcs[c]


    def next_piece(self):

        if not self.next:
                self.next = self.random_piece()
                self.current = self.random_piece()
        else:
                self.current = self.next
                self.next = self.random_piece()
```

```
        self.current.set_point(*start_point)

def move_down(self):

        if(self.current.y < theight+100):
                self.current.y += offset

                if self.has_overlap():
                        self.current.y -= offset
                        self.current.mobile = False
        else:
                self.current.mobile = False

        if self.current.mobile == False and self.
           current.y == 100:
                self.game_over = True

# this is a hard drop - just go all the way down
   until you hit something
def drop_down(self):

        while(self.current.mobile): self.move_down
            ()

def move_right(self):
        if(self.current.mobile): self.current.x +=
           offset

        if self.has_overlap(): self.current.x -=
           offset

def move_left(self):
        if(self.current.mobile): self.current.x -=
           offset
        if self.has_overlap(): self.current.x +=
           offset

# TODO: wall kick logic needed
def rotate(self):
        self.current.flip()

        while self.has_overlap(): self.current.flip
            ()
```

```
# Saves individual blocks of the current piece -
   they become part of the grid
# Automatically check for lines
def remember_block_positions(self):

        self.blocks.extend(self.current.get_blocks
           ())
        self.next_piece()

        self.blocks = sorted(self.blocks, key=
           lambda block: block.y)

        self.check_for_lines()

        #for b in self.blocks: print str(b.y)+", "

def draw_blocks(self):

        for b in self.blocks:
                b.draw()

# check if current piece overlaps with walls or
   with other pieces
def has_overlap(self):

        blocks = self.current.get_blocks()

        for b in blocks:
                if b.x < 125 or b.x > right_edge
                   or b.y < 100 or b.y >
                   bottom_edge: return True

        # TODO: optimize the shit out of this
        for b in self.blocks:
                for c in self.current.get_blocks():
                        if b.x == c.x and b.y == c.
                           y: return True

        return False

# TODO: there ought to be a better way to do this
def check_for_lines(self):

        lines = {}
```

```
        for b in self.blocks:

                if b.y in lines:
                        lines[b.y] +=1
                else:
                        lines[b.y] = 1

        lines_to_be_destroyed = []

        for ln in lines:
                if lines[ln] == 10:
                        self.total_cleared_lines +=
                            1
                        self.cleared_lines += 1
                        lines_to_be_destroyed.
                           append(ln)

        for l in lines_to_be_destroyed:
                self.destroy_line(l)

        copy_of_blocks = self.blocks[:]

        if len(lines_to_be_destroyed) > 0:
                self.collapse_hovering_blocks(min(
                   lines_to_be_destroyed), len(
                   lines_to_be_destroyed))

                #for ln in lines_to_be_destroyed:
                        #self.
                           collapse_hovering_blocks
                           (ln, copy_of_blocks)

        self.calculate_score()


def destroy_line(self, ln):

        # note the slice notation - I'm iterating
           over a copy of self.blocks but removing
        # from the original
        for b in self.blocks[:]:
                if b.y == ln: self.blocks.remove(b)
```

```
def collapse_hovering_blocks(self, ln, total):
        """ drop down all the blocks that are
            hovering """

        for i,b in enumerate(self.blocks):
                if b.y < ln:
                        self.blocks[i].y += offset
                            * total



def block_overlaps(self, block, block_list):


        for b in block_list:
                if block.y == b.y and block.x == b.
                    x:
                        return True

        return False



def calculate_score(self):

        # score is calculated like so: M * ( N + 1
            ) where:
        # M is multiplier (see self.multiplier)
        # N is level (self.level)

        self.score += self.multiplier[self.
            cleared_lines] * (self.level + 1)

        tmp = self.lines_til_next_level - self.
            cleared_lines

        if tmp <= 0:
                self.level +=1
                self.lines_til_next_level =
                    lines_per_level + tmp

                self.delay -= self.level * 30

                if self.delay < 10: self.delay = 10
```

```
        else:
                self.lines_til_next_level = tmp

        self.cleared_lines = 0
        self.draw_text()


# redraws the whole UI - we should really be
   bliting this shit
def draw_ui(self):

        pygame.draw.rect(screen, gray, (125,100,
           right_edge+offset-125, bottom_edge+
           offset-100))

        for i in xrange(10): pygame.draw.line(
           screen, black, (i*offset+125, 100),
           (125+i*offset, theight+offset+100))
        for i in xrange(24): pygame.draw.line(
           screen, black, (125, i*offset+100), (
           twidth+125, i*offset+100))

        self.draw_text()


# I really dislike the default font in pygame
def draw_text(self):

        msg = self.nfont.render("NEXT PIECE:", 1,
           white)
        screen.blit(msg, (250+125, 50+100))

        pygame.draw.rect(screen, black, (250+125,
           100+200, 200, 100))

        #sc = self.nfont.render("SCORE: " + str(
           self.score), 1, white)
        #screen.blit(sc, (250+125, 200+100))

        #sc = self.nfont.render("LINES: " + str(
           self.total_cleared_lines), 1, white)
        #screen.blit(sc, (250+125, 230+100))

        #sc = self.nfont.render("LEVEL: " + str(
           self.level), 1, white)
```

```
            #screen.blit(sc, (250, 260))



            #msg = self.smallfont.render("Left, Right
               Arrow to move", 1, white)
            #screen.blit(msg, (230, 290))

            #msg = self.smallfont.render("Up Arrow to
               flip", 1, white)
            #screen.blit(msg, (230, 300))

            #msg = self.smallfont.render("Down Arrow to
                move downw", 1, white)
            #screen.blit(msg, (230, 310))

            #msg = self.smallfont.render("Space or
               Enter to drop down", 1, white)
            #screen.blit(msg, (230, 320))

            #msg = self.smallfont.render("Esc to pause,
                F1 for new game", 1, white)
            #screen.blit(msg, (230, 330))

            #screen.blit(self.smallfont.render("ver
               0.2", 1, white), (230, 350))

        def draw_game_over(self):

            msg = self.largefont.render("GAME OVER", 1,
                red)
            screen.blit(msg, (195+125,95+100))

            #msg2 = self.nfont.render("PRESS F1 TO PLAY
                AGAIN", 1, red)
            #screen.blit(msg2, (175, 130))




# magic
screen = pygame.display.set_mode(size)
pygame.key.set_repeat(100, 150)

#pygame.time.set_timer(USEREVENT+1, delay)
```

```
grid = Grid()
clock = pygame.time.Clock()

paused = False

time_elapsed = 0

while st == 1:
        currentTime = time.time()
        if grid.total_cleared_lines>lastlinecount:
                positivesound.play()
                lastlinecount=grid.total_cleared_lines
                counter+=1
        if not paused and not grid.game_over:

                time_elapsed += clock.tick()

                #print str(grid.delay)

                # timed block drop
                if time_elapsed > grid.delay:
                        time_elapsed = 0
                        grid.move_down()

                # check if a line was created and remove it
                grid.check_for_lines()

                # clear the next piece area
                pygame.draw.rect(screen, black, grid.
                   next_rect)

                # draw the gridlines
                grid.draw_ui()

                # if the current piece is locked in place
                   memorize it's position
                if not grid.current.mobile:
                        grid.remember_block_positions()

                # draw the memorized blocks
                grid.draw_blocks()


        for event in pygame.event.get():
```

```
if event.type == pygame.QUIT: sys.exit()


if event.type == KEYDOWN:

        if event.key == K_ESCAPE: paused =
           not paused

        if event.key == K_F1:
                grid = Grid()
                clock = pygame.time.Clock()
                counter = 0
                paused = False
                time_elapsed = 0

        if mode == True:
                if not paused and not grid.
                   game_over:
                        if event.key == K_d
                           : grid.move_left
                           ()
                        if event.key == K_a
                           : grid.
                           move_right()
                        if event.key == K_s
                           : grid.move_down
                           ()
                        if event.key == K_w
                           : grid.rotate()
                        if event.key ==
                           K_SPACE: grid.
                           drop_down()
                        if event.key ==
                           K_RETURN: grid.
                           drop_down()
                        if event.key == K_q
                           : mode = False
                        if event.key == K_1
                           : grid.delay =
                           100
                        if event.key == K_2
                           : grid.delay =
                           200
                        if event.key == K_3
```

```
                                        : grid.delay =
                                        300
                                    if event.key == K_4
                                        : grid.delay =
                                        400
                                    if event.key == K_5
                                        : grid.delay =
                                        500
                                    if event.key == K_6
                                        : grid.delay =
                                        600
                                    if event.key == K_7
                                        : grid.delay =
                                        700
                                    if event.key == K_8
                                        : grid.delay =
                                        800
                                    if event.key == K_9
                                        : grid.delay =
                                        900
                                    if event.key == K_0
                                        : grid.delay =
                                        1000
                                    if event.key ==
                                        K_UP: counter
                                        +=1;
                                        positivesound.
                                        play()
                                    if event.key ==
                                        K_DOWN: counter
                                        -=1.5
                                    if event.key == K_m
                                        : counter-=4
                                    if event.key == K_p
                                        : st=0
                if mode == False:
                        if not paused and not grid.
                            game_over:
                                    if event.key == K_d
                                        : grid.
                                        move_right()
                                    if event.key == K_a
                                        : grid.move_left
                                        ()
                                    if event.key == K_s
```

```
        : grid.move_down
        ()
    if event.key == K_w
        : grid.rotate()
    if event.key ==
        K_SPACE: grid.
        drop_down()
    if event.key ==
        K_RETURN: grid.
        drop_down()
    if event.key == K_e
        : mode = True
    if event.key == K_1
        : grid.delay =
        100
    if event.key == K_2
        : grid.delay =
        200
    if event.key == K_3
        : grid.delay =
        300
    if event.key == K_4
        : grid.delay =
        400
    if event.key == K_5
        : grid.delay =
        500
    if event.key == K_6
        : grid.delay =
        600
    if event.key == K_7
        : grid.delay =
        700
    if event.key == K_8
        : grid.delay =
        800
    if event.key == K_9
        : grid.delay =
        900
    if event.key == K_0
        : grid.delay =
        1000
    if event.key ==
        K_UP: counter
        +=1;
```

```
                                        positivesound.
                                        play()
                                    if event.key ==
                                        K_DOWN: counter
                                        -=1.5
                                    if event.key == K_m
                                        : counter-=4
                                    if event.key == K_p
                                        : st=0


    if not paused and not grid.game_over:

            grid.current.draw()      # draw current
                piece
            grid.next.draw()         # draw the next
                piece
            if counter < 0:
                    currentcolor = red
            else:
                    currentcolor = green
            if counter >=11: counter = 11
            if counter <=-10: counter = -10
            pygame.draw.rect(screen, black,
                (370+125,100,200,1000), 0)
            screen.blit(perf, (405+125, 100+50))
            #screen.blit(timeshow, (250, 300))
            if currentTime - lastTime > 1:
                    countdown-=1
                    if countdown <= 0:
                            countdown = 0
                    lastTime = currentTime
            m, s = divmod(countdown, 60)
            #if m >= 1:
            #       screen.blit(timeFont.render(str(m)
                ,1,white), (305, 300))
            #       screen.blit(timeFont.render(":",1,
                white), (313, 300))
            #       screen.blit(timeFont.render(str(s)
                ,1,white), (320, 300))
            #else:
            #       screen.blit(timeFont.render(str(s)
                ,1,white), (320, 300))
            pygame.draw.rect(screen, currentcolor,
                (370+125,100+230,200,-counter*15), 0)
```

```
            screen.blit(img,(180+125,100+250))
            pygame.display.flip()

    # draw game over message
    if grid.game_over:
            #grid.draw_game_over()
            counter = 0
            if currentTime - lastTime > 1:
                    countdown-=1
                    if countdown <= 0:
                            countdown = 0
                    lastTime = currentTime
            #pygame.display.flip()
            grid = Grid()
            clock = pygame.time.Clock()
            counter = 0
            paused = False
            time_elapsed = 0
```

TETRIS SCENARIO 2

```
#!/usr/bin/env python

# PyTris (c) Lukasz Grzegorz Maciak
# Licensed under GNU General Public License Version 3

import sys, pygame, random
from pygame.locals import *
from pygame import K_q
import time
from time import sleep

pygame.mixer.pre_init(44100,-16,2,2048)
pygame.mixer.init()
pygame.init()

pygame.mixer.music.load('C:/Users/erikm/Google Drive/My
   Documents/Skoleting/Master/Scenarios/1st Prototype 27
   apr/Experiment/Sound/War Warrior.mp3')
positivesound = pygame.mixer.Sound('C:/Users/erikm/Google
   Drive/My Documents/Skoleting/Master/Scenarios/1st
   Prototype 27 apr/Experiment/Sound/levelup.wav')
positivesound.set_volume(0.7)
negativesound = pygame.mixer.Sound('C:/Users/erikm/Google
   Drive/My Documents/Skoleting/Master/Scenarios/1st
   Prototype 27 apr/Experiment/Sound/Wrongbuzz.wav')
negativesound.set_volume(0.6)
pygame.mixer.music.play()

# useful constants
size = width, height = 800, 600
lines_per_level = 200           #increas level after x
   lines
delay = 200                     #speed
mode = True                             #for switching
   directions
counter = 0             #feedback bar counter
st = 1
bitnumber = 0

# define colors
black = 0, 0, 0
red = 255, 0, 0
green = 0, 255, 0
```

```
blue = 0, 0, 255
white = 255, 255, 255
yellow = 255, 255, 0
purple = 160, 32, 240
cyan = 0, 255, 255
orange = 255, 165, 0
gray =  45, 45, 45

currentcolor = red      #for feedback bar
myfont = pygame.font.Font(None, 24)
timeFont = pygame.font.Font(None, 50)
perf = myfont.render("PERFORMANCE:", 1, white)
myfont2 = pygame.font.Font(None, 14)
timeshow = timeFont.render("TIME:", 1, white)
startTime = time.time()
lastTime = time.time()
lastTime1 = time.time()
countdown = 270

block_size = 15         # each piece is 4 blocks
block_gap = 1           # distance between blocks

offset = block_size + block_gap

twidth = 10 * offset
right_edge = 125+twidth - block_size

theight = 24 * offset
bottom_edge = 100+theight - block_size

start_point = 125 + twidth/2 - block_size, 100
next_point = 125 + twidth+125, 100+100

# Block object exists mostly to allow us to break a piece
   into individual
# components once it is locked in place. This is why the
   color information
# is redundant
class Block(object):
        """ Represents a rectangular Tetris block """

        def __init__(self, x, y, color):

                self.x = x
                self.y = y
```

```
        self.color = color

    def draw(self):

        pygame.draw.rect(screen, self.color, self.
            rect)

    @property
    def rect(self):
        return (self.x, self.y, block_size,
            block_size)

class Piece(object):
    """ A Tetris Piece - composed of 4 blocks """

    def __init__(self,x,y):

        self.x = x
        self.y = y
        self.mobile = True

        self.rotation = 0 # defines which of the
            members of self.positions to use for
            this object

        # set of offsets that is applied to
            coordinates of each block in this piece
        self.positions = None


    def get_blocks(self):
        """ Returns an array of 4 blocks which make
             up this piece; each block has it’s own
            coordinates and draw function """

        blocks = []

        for i in range(4):
            blocks.append( Block(self.x + self.
                positions[self.rotation][i][0],
                self.y + self.positions[self.
                rotation][i][1], self.color) )

        return blocks
```

```
	def draw(self):
		""" Draw all the blocks of this piece to
		    the screen """

		blocks = self.get_blocks()

		for b in blocks:
			pygame.draw.rect(screen, self.color
			    , b.rect)

	# calling flip repeatedly cycles through all
	   available positions and goes back to the first
	   one
	def flip(self):
		if(self.rotation < len(self.positions) -1):
			self.rotation += 1
		else:
			self.rotation = 0

	def set_point(self, x, y):
		self.x = x
		self.y = y


# Define different piece types: Z, S, O, T, I, L, J

class ZPiece(Piece):

	def __init__(self,x,y):

		super(ZPiece, self).__init__(x, y)

		self.color = red

		self.positions =	[
					( (0,0), (
					   offset,
					   0), (
					   offset,
					   offset),
					    (2*
					   offset,
					   offset)
					        ),
					( (0,0),
```

```
                                                     (0,
                                                     offset),
                                                      (-
                                                     offset,
                                                     offset),
                                                      (-
                                                     offset,
                                                     2*offset
                                                     )    )
                                          ]
class SPiece(Piece):

        def __init__(self,x,y):

                super(SPiece, self).__init__(x, y)

                self.color = green

                self.positions =        [
                                                  ( (0,0), (-
                                                     offset,
                                                     0), (-
                                                     offset,
                                                     offset),
                                                      (-2*
                                                     offset,
                                                     offset)
                                                       ),
                                                  ( (0,0),
                                                     (0,
                                                     offset),
                                                      (offset
                                                     , offset
                                                     ), (
                                                     offset,
                                                     2*offset
                                                     )      )
                                          ]


class OPiece(Piece):

        def __init__(self, x, y):

                super(OPiece, self).__init__(x,y)
```

```
            self.color = yellow

            self.positions =        [
                                          ( (0,0), (
                                            offset,
                                            0), (
                                            offset,
                                            offset),
                                             (0,
                                            offset)
                                                )
                                    ]


class TPiece(Piece):

      def __init__(self,x,y):

            super(TPiece, self).__init__(x, y)

            self.color = purple

            self.positions =        [
                                          ( (0,0),
                                             (0,
                                            offset),
                                             (-
                                            offset,
                                            offset),
                                             (offset
                                            , offset
                                            )
                                            ),
                                          ( (0,0),
                                             (0,
                                            offset),
                                             (offset
                                            , offset
                                            ), (0,
                                            2*offset
                                            )   ),
                                          ( (0,0), (-
                                            offset,
                                            0), (
```

```
                                                         offset,
                                                         0), (0,
                                                         offset)

                                                         ),
                                                      ( (0,0),
                                                         (0,
                                                         offset),
                                                          (-
                                                         offset,
                                                         offset),
                                                          (0, 2*
                                                         offset)
                                                          ),
                                              ]


class IPiece(Piece):

        def __init__(self,x,y):

                super(IPiece, self).__init__(x, y)

                self.color = cyan

                self.positions =        [
                                                      ( (0,0),
                                                         (0,
                                                         offset),
                                                          (0, 2*
                                                         offset),
                                                          (0, 3*
                                                         offset)
                                                              ),
                                                      ( (0,0), (
                                                         offset,
                                                         0), (2*
                                                         offset,
                                                         0), (3*
                                                         offset,
                                                         0)
                                                         )
                                              ]
```

```
class LPiece(Piece):

        def __init__(self,x,y):

                super(LPiece, self).__init__(x, y)

                self.color = orange

                self.positions =        [
                                                ( (0,0),
                                                   (0,
                                                  offset),
                                                   (0, 2*
                                                  offset),
                                                   (offset
                                                  , 2*
                                                  offset)

                                                  ),
                                                ( (0,0),
                                                   (0,
                                                  offset),
                                                   (offset
                                                  , 0),
                                                  (2*
                                                  offset,
                                                  0)
                                                        )
                                                  ,
                                                ( (0,0), (-
                                                  offset,
                                                  0), (0,
                                                  offset),
                                                   (0, 2*
                                                  offset)
                                                        ),
                                                ( (0,0),
                                                   (0,
                                                  offset),
                                                   (-
                                                  offset,
                                                  offset),
                                                   (-2*
                                                  offset,
                                                  offset)
```

```
                                                  ),
                                    ]

class JPiece(Piece):

        def __init__(self,x,y):

                super(JPiece, self).__init__(x, y)

                self.color = blue

                self.positions =        [
                                                ( (0,0),
                                                   (0,
                                                   offset),
                                                    (0, 2*
                                                   offset),
                                                    (-
                                                   offset,
                                                   2*offset
                                                   )
                                                          )
                                                   ,
                                                ( (0,0), (
                                                   offset,
                                                   0), (2*
                                                   offset,
                                                   0), (2*
                                                   offset,
                                                   offset)

                                                   ),
                                                ( (0,0), (
                                                   offset,
                                                   0), (0,
                                                   offset),
                                                    (0, 2*
                                                   offset)
                                                          )
                                                   ,
                                                ( (0,0),
                                                   (0,
                                                   offset),
                                                    (offset
                                                   , offset
```

```
                                                    ), (2*
                                                    offset,
                                                    offset)
                                                        ),
                                         ]



class Grid(object):

        def __init__(self):

                self.current = None
                self.next = None
                self.blocks = []

                self.next_piece()

                self.total_cleared_lines = 0
                self.cleared_lines = 0
                self.level = 1
                self.score = 0

                self.lines_til_next_level = lines_per_level

                self.delay = delay

                self.game_over = False

                self.next_rect = pygame.Rect(150+125,
                   90+100, 300, 300)

                # this is for multiplying scores
                self.multiplier =       {
                                                0 : 0,

                                                   # no
                                                   lines
                                                   cleared
                                                1 : 40,

                                                   # single
                                                    line
                                                   cleared
                                                2 : 100,
```

```
                                                    # two
                                                    lines
                                                    cleared
                                                3 : 300,

                                                    # three
                                                    lines
                                                    cleared
                                                4 : 1200

                                                    # TETRIS
                                    }

        self.nfont = pygame.font.Font(None, 24)
        self.largefont = pygame.font.Font(None, 40)
        self.smallfont = pygame.font.Font(None, 14)


    def random_piece(self):
        rekkefolge =
            [3,5,3,5,4,0,1,5,5,1,1,2,1,2,5,2,3,0,2,2,0,3,6,0,4,5,

        pcs =   {
                        0 : ZPiece(*next_point),
                        1 : OPiece(*next_point),
                        2 : SPiece(*next_point),
                        3 : TPiece(*next_point),
                        4 : IPiece(*next_point),
                        5 : LPiece(*next_point),
                        6 : JPiece(*next_point)
                }

        global bitnumber
        c = rekkefolge[bitnumber]
        bitnumber+=1
        return pcs[c]


    def next_piece(self):

        if not self.next:
                self.next = self.random_piece()
                self.current = self.random_piece()
        else:
```

```
                self.current = self.next
                self.next = self.random_piece()

        self.current.set_point(*start_point)

def move_down(self):

        if(self.current.y < theight+100):
                self.current.y += offset

                if self.has_overlap():
                        self.current.y -= offset
                        self.current.mobile = False
        else:
                self.current.mobile = False

        if self.current.mobile == False and self.
            current.y == 100:
                self.game_over = True

# this is a hard drop - just go all the way down
    until you hit something
def drop_down(self):

        while(self.current.mobile): self.move_down
            ()

def move_right(self):
        if(self.current.mobile): self.current.x +=
            offset

        if self.has_overlap(): self.current.x -=
            offset

def move_left(self):
        if(self.current.mobile): self.current.x -=
            offset
        if self.has_overlap(): self.current.x +=
            offset

# TODO: wall kick logic needed
def rotate(self):
        self.current.flip()

        while self.has_overlap(): self.current.flip
```

```
        ()

# Saves individual blocks of the current piece -
    they become part of the grid
# Automatically check for lines
def remember_block_positions(self):

        self.blocks.extend(self.current.get_blocks
            ())
        self.next_piece()

        self.blocks = sorted(self.blocks, key=
            lambda block: block.y)

        self.check_for_lines()

        #for b in self.blocks: print str(b.y)+", "

def draw_blocks(self):

        for b in self.blocks:
                b.draw()

# check if current piece overlaps with walls or
    with other pieces
def has_overlap(self):

        blocks = self.current.get_blocks()

        for b in blocks:
                if b.x < 125 or b.x > right_edge
                    or b.y < 100 or b.y >
                    bottom_edge: return True

        # TODO: optimize the shit out of this
        for b in self.blocks:
                for c in self.current.get_blocks():
                        if b.x == c.x and b.y == c.
                            y: return True

        return False

# TODO: there ought to be a better way to do this
def check_for_lines(self):
```

```
        lines = {}

        for b in self.blocks:

                if b.y in lines:
                        lines[b.y] +=1
                else:
                        lines[b.y] = 1

        lines_to_be_destroyed = []

        for ln in lines:
                if lines[ln] == 10:
                        self.total_cleared_lines +=
                             1
                        self.cleared_lines += 1
                        lines_to_be_destroyed.
                           append(ln)

        for l in lines_to_be_destroyed:
                self.destroy_line(l)

        copy_of_blocks = self.blocks[:]

        if len(lines_to_be_destroyed) > 0:
                self.collapse_hovering_blocks(min(
                   lines_to_be_destroyed), len(
                   lines_to_be_destroyed))

                #for ln in lines_to_be_destroyed:
                        #self.
                           collapse_hovering_blocks
                           (ln, copy_of_blocks)

        self.calculate_score()


def destroy_line(self, ln):

        # note the slice notation - I'm iterating
           over a copy of self.blocks but removing
        # from the original
        for b in self.blocks[:]:
                if b.y == ln: self.blocks.remove(b)
```

```
def collapse_hovering_blocks(self, ln, total):
        """ drop down all the blocks that are
           hovering """

        for i,b in enumerate(self.blocks):
                if b.y < ln:
                        self.blocks[i].y += offset
                           * total



def block_overlaps(self, block, block_list):


        for b in block_list:
                if block.y == b.y and block.x == b.
                   x:
                        return True

        return False



def calculate_score(self):

        # score is calculated like so: M * ( N + 1
           ) where:
        # M is multiplier (see self.multiplier)
        # N is level (self.level)

        self.score += self.multiplier[self.
           cleared_lines] * (self.level + 1)

        tmp = self.lines_til_next_level - self.
           cleared_lines

        if tmp <= 0:
                self.level +=1
                self.lines_til_next_level =
                   lines_per_level + tmp

                self.delay -= self.level * 30
```

```
                if self.delay < 10: self.delay = 10

        else:
                self.lines_til_next_level = tmp

        self.cleared_lines = 0
        self.draw_text()


# redraws the whole UI - we should really be
    bliting this shit
def draw_ui(self):

        pygame.draw.rect(screen, gray, (125,100,
            right_edge+offset-125, bottom_edge+
            offset-100))

        for i in xrange(10): pygame.draw.line(
            screen, black, (i*offset+125, 100),
            (125+i*offset, theight+offset+100))
        for i in xrange(24): pygame.draw.line(
            screen, black, (125, i*offset+100), (
            twidth+125, i*offset+100))

        self.draw_text()


# I really dislike the default font in pygame
def draw_text(self):

        msg = self.nfont.render("NEXT PIECE:", 1,
            white)
        screen.blit(msg, (250+125, 50+100))

        pygame.draw.rect(screen, black, (250+125,
            200+100, 200, 100))

        sc = self.nfont.render("YOUR SCORE: ", 1,
            white)
        screen.blit(sc, (240+125, 220+100))
        sc = self.largefont.render(str(self.score)
            ,1,white)
        screen.blit(sc, (270+125, 240+100))

        #sc = self.nfont.render("LINES: " + str(
```

```
            self.total_cleared_lines), 1, white)
        #screen.blit(sc, (250, 230))

        #sc = self.nfont.render("LEVEL: " + str(
            self.level), 1, white)
        #screen.blit(sc, (250, 260))



        #msg = self.smallfont.render("Left, Right
            Arrow to move", 1, white)
        #screen.blit(msg, (230, 290))

        #msg = self.smallfont.render("Up Arrow to
            flip", 1, white)
        #screen.blit(msg, (230, 300))

        #msg = self.smallfont.render("Down Arrow to
             move downw", 1, white)
        #screen.blit(msg, (230, 310))

        #msg = self.smallfont.render("Space or
            Enter to drop down", 1, white)
        #screen.blit(msg, (230, 320))

        #msg = self.smallfont.render("Esc to pause,
             F1 for new game", 1, white)
        #screen.blit(msg, (230, 330))

        #screen.blit(self.smallfont.render("ver
            0.2", 1, white), (230, 350))

def draw_game_over(self):

        pygame.draw.rect(screen, black,
            (125,0,800,800),0)

        msg = self.largefont.render("GAME OVER", 1,
             red)
        screen.blit(msg, (195+125,95+100))

        #msg2 = self.nfont.render("PRESS F1 TO PLAY
             AGAIN", 1, red)
        #screen.blit(msg2, (175, 130))
```

```
# magic
screen = pygame.display.set_mode(size)
pygame.key.set_repeat(100, 150)

#pygame.time.set_timer(USEREVENT+1, delay)

grid = Grid()
clock = pygame.time.Clock()

paused = False

time_elapsed = 0

while st == 1:
        currentTime = time.time()
        if (currentTime-lastTime1>150):
                mode = False
                grid.delay = 170
        if (currentTime-lastTime1>240):
                mode = True
                grid.delay = 150
        if not paused and not grid.game_over:

                time_elapsed += clock.tick()

                #print str(grid.delay)

                # timed block drop
                if time_elapsed > grid.delay:
                        time_elapsed = 0
                        grid.move_down()

                # check if a line was created and remove it
                grid.check_for_lines()

                # clear the next piece area
                pygame.draw.rect(screen, black, grid.
                   next_rect)

                # draw the gridlines
                grid.draw_ui()
```

```
        # if the current piece is locked in place
           memorize it's position
        if not grid.current.mobile:
                grid.remember_block_positions()

        # draw the memorized blocks
        grid.draw_blocks()


for event in pygame.event.get():

        if event.type == pygame.QUIT: sys.exit()


        if event.type == KEYDOWN:

                if event.key == K_ESCAPE: paused =
                   not paused

                if event.key == K_F1:
                        grid = Grid()
                        clock = pygame.time.Clock()
                        counter = 0
                        #countdown = 300
                        paused = False
                        time_elapsed = 0

                if mode == True:
                        if not paused and not grid.
                           game_over:
                                if event.key == K_d
                                   : grid.move_left
                                   ()
                                if event.key == K_a
                                   : grid.
                                   move_right()
                                if event.key == K_s
                                   : grid.move_down
                                   ()
                                if event.key == K_w
                                   : grid.rotate()
                                if event.key ==
                                   K_SPACE: grid.
                                   drop_down()
                                if event.key ==
```

```
    K_RETURN: grid.
    drop_down()
if event.key == K_q
    : mode = False
if event.key == K_1
    : grid.delay =
    160
if event.key == K_2
    : grid.delay =
    200
if event.key == K_3
    : grid.delay =
    300
if event.key == K_4
    : grid.delay =
    400
if event.key == K_5
    : grid.delay =
    500
if event.key == K_6
    : grid.delay =
    600
if event.key == K_7
    : grid.delay =
    700
if event.key == K_8
    : grid.delay =
    800
if event.key == K_9
    : grid.delay =
    900
if event.key == K_0
    : grid.delay =
    1000
if event.key ==
    K_UP: counter
    +=1;
if event.key ==
    K_DOWN: counter
    -=1.5;
    negativesound.
    play()
if event.key == K_m
    : counter-=4
if event.key == K_p
```

```
                        : st=0
            if mode == False:
                    if not paused and not grid.
                        game_over:
                            if event.key == K_d
                                : grid.
                                move_right()
                            if event.key == K_a
                                : grid.move_left
                                ()
                            if event.key == K_s
                                : grid.move_down
                                ()
                            if event.key == K_w
                                : grid.rotate()
                            if event.key ==
                                K_SPACE: grid.
                                drop_down()
                            if event.key ==
                                K_RETURN: grid.
                                drop_down()
                            if event.key == K_e
                                : mode = True
                            if event.key == K_1
                                : grid.delay =
                                160
                            if event.key == K_2
                                : grid.delay =
                                200
                            if event.key == K_3
                                : grid.delay =
                                300
                            if event.key == K_4
                                : grid.delay =
                                400
                            if event.key == K_5
                                : grid.delay =
                                500
                            if event.key == K_6
                                : grid.delay =
                                600
                            if event.key == K_7
                                : grid.delay =
                                700
                            if event.key == K_8
```

```
                                                    : grid.delay =
                                                    800
                                                if event.key == K_9
                                                    : grid.delay =
                                                    900
                                                if event.key == K_0
                                                    : grid.delay =
                                                    1000
                                                if event.key ==
                                                    K_UP: counter
                                                    +=1;
                                                if event.key ==
                                                    K_DOWN: counter
                                                    -=1.5;
                                                    negativesound.
                                                    play()
                                                if event.key == K_m
                                                    : counter-=4
                                                if event.key == K_p
                                                    : st=0


        if not paused and not grid.game_over:

                grid.current.draw()      # draw current
                    piece
                grid.next.draw()         # draw the next
                    piece
                if counter < 0:
                        currentcolor = red
                else:
                        currentcolor = green
                if counter >=11: counter = 11
                if counter <=-10: counter = -10
                pygame.draw.rect(screen, black,
                    (370+125,100,200,1000), 0)
                screen.blit(perf, (405+125, 50+100))
                #screen.blit(timeshow, (250+125, 300+100))
                if currentTime - lastTime > 1:
                        countdown-=1
                        if countdown <= 0:
                                countdown = 30
                        lastTime = currentTime
                m, s = divmod(countdown, 60)
                if m >= 1:
```

```
                screen.blit(timeFont.render(str(m)
                    ,1,white), (305+75, 300+100))
                screen.blit(timeFont.render(":",1,
                    white), (313+85, 300+100))
                screen.blit(timeFont.render(str(s)
                    ,1,white), (320+95, 300+100))
        else:
                screen.blit(timeFont.render(str(s)
                    ,1,white), (320+95, 300+100))
        pygame.draw.rect(screen, currentcolor,
            (370+125,230+100,200,-counter*15), 0)
        pygame.display.flip()


# draw game over message
if grid.game_over:
        grid.draw_game_over()
        pygame.display.flip()
        sleep(1)
        counter = 0
        if currentTime - lastTime > 1:
                countdown-=1
                if countdown <= 0:
                        countdown = 30
                lastTime = currentTime
        grid = Grid()
        clock = pygame.time.Clock()
        counter = 0
        paused = False
        time_elapsed = 0
```

TETRIS SCENARIO 3

```
#!/usr/bin/env python

# PyTris (c) Lukasz Grzegorz Maciak
# Licensed under GNU General Public License Version 3

import sys, pygame, random
from pygame.locals import *
from pygame import K_q
import time
from time import sleep

img = pygame.image.load('cat.png')


pygame.mixer.init()
pygame.init()

pygame.mixer.music.load('C:/Users/erikm/Google Drive/My
   Documents/Skoleting/Master/Scenarios/1st Prototype 27
   apr/Experiment/Sound/Sandstorm.mp3')
positivesound = pygame.mixer.Sound('C:/Users/erikm/Google
   Drive/My Documents/Skoleting/Master/Scenarios/1st
   Prototype 27 apr/Experiment/Sound/levelup.wav')
positivesound.set_volume(0.7)
negativesound = pygame.mixer.Sound('C:/Users/erikm/Google
   Drive/My Documents/Skoleting/Master/Scenarios/1st
   Prototype 27 apr/Experiment/Sound/Wrongbuzz.wav')
negativesound.set_volume(0.3)
levelupsound = pygame.mixer.Sound('C:/Users/erikm/Google
   Drive/My Documents/Skoleting/Master/Scenarios/1st
   Prototype 27 apr/Experiment/Sound/Epic_win2.wav')
pygame.mixer.music.play()

# useful constants
size = width, height = 800, 600
lines_per_level = 200           #increas level after x
   lines
delay = 600                     #speed
mode = False                            #for switching
   directions
counter = 0             #feedback bar counter
highscore = 780
highscorecount = 3
```

```
st = 1
bitnumber = 0
lastlinecount = 0


# define colors
black = 0, 0, 0
red = 255, 0, 0
green = 0, 255, 0
blue = 0, 0, 255
white = 255, 255, 255
yellow = 255, 255, 0
purple = 160, 32, 240
cyan = 0, 255, 255
orange = 255, 165, 0
gray =  45, 45, 45

currentcolor = red      #for feedback bar
myfont = pygame.font.Font(None, 24)
timeFont = pygame.font.Font(None, 24)
perf = myfont.render("PERFORMANCE:", 1, white)
myfont2 = pygame.font.Font(None, 14)
timeshow = timeFont.render("TIME:", 1, white)
highscoreFont = timeFont.render("SCORE TO BEAT:",1,white)
myfont3 = pygame.font.Font(None, 35)
myfont4 = pygame.font.Font(None,50)
inthelead1 = myfont4.render("YOU ARE NOW IN THE LEAD!",1,
   green)
inthelead2 = myfont4.render("YOU ARE NOW IN THE LEAD!",1,
   blue)
thirdplace1 = myfont4.render("YOU ARE NOW IN 3rd PLACE!",1,
   green)
thirdplace2 = myfont4.render("YOU ARE NOW IN 3rd PLACE!",1,
   blue)
secondplace1 = myfont4.render("YOU ARE NOW IN 2rd PLACE
   !",1,green)
secondplace2 = myfont4.render("YOU ARE NOW IN 2rd PLACE
   !",1,blue)
startTime = time.time()
lastTime = time.time()
countdown = 300

block_size = 15         # each piece is 4 blocks
block_gap = 1           # distance between blocks
```

```
offset = block_size + block_gap

twidth = 10 * offset
right_edge = 125+twidth - block_size

theight = 24 * offset
bottom_edge = 100+theight - block_size

start_point = 125+twidth/2 - block_size, 100
next_point = 125+twidth+125, 100+100

# Block object exists mostly to allow us to break a piece
   into individual
# components once it is locked in place. This is why the
   color information
# is redundant

def bitcountfunction(arg):
        arg-=1

class Block(object):
        """ Represents a rectangular Tetris block """

        def __init__(self, x, y, color):

                self.x = x
                self.y = y
                self.color = color

        def draw(self):

                pygame.draw.rect(screen, self.color, self.
                   rect)

        @property
        def rect(self):
                return (self.x, self.y, block_size,
                   block_size)

class Piece(object):
        """ A Tetris Piece - composed of 4 blocks """

        def __init__(self,x,y):

                self.x = x
```

```
        self.y = y
        self.mobile = True

        self.rotation = 0 # defines which of the
           members of self.positions to use for
           this object

        # set of offsets that is applied to
           coordinates of each block in this piece
        self.positions = None


def get_blocks(self):
        """ Returns an array of 4 blocks which make
            up this piece; each block has it’s own
           coordinates and draw function """

        blocks = []

        for i in range(4):
                blocks.append( Block(self.x + self.
                   positions[self.rotation][i][0],
                   self.y + self.positions[self.
                   rotation][i][1], self.color) )

        return blocks

def draw(self):
        """ Draw all the blocks of this piece to
           the screen """

        blocks = self.get_blocks()

        for b in blocks:
                pygame.draw.rect(screen, self.color
                   , b.rect)

# calling flip repeatedly cycles through all
   available positions and goes back to the first
   one
def flip(self):
        if(self.rotation < len(self.positions) -1):
                self.rotation += 1
        else:
                self.rotation = 0
```

```
        def set_point(self, x, y):
                self.x = x
                self.y = y


# Define different piece types: Z, S, O, T, I, L, J

class ZPiece(Piece):

        def __init__(self,x,y):

                super(ZPiece, self).__init__(x, y)

                self.color = red

                self.positions =        [
                                                ( (0,0), (
                                                   offset,
                                                   0), (
                                                   offset,
                                                   offset),
                                                    (2*
                                                   offset,
                                                   offset)
                                                        ),
                                                ( (0,0),
                                                    (0,
                                                   offset),
                                                    (-
                                                   offset,
                                                   offset),
                                                    (-
                                                   offset,
                                                   2*offset
                                                   )    )
                                        ]
class SPiece(Piece):

        def __init__(self,x,y):

                super(SPiece, self).__init__(x, y)

                self.color = green
```

```
            self.positions =        [
                                          ( (0,0), (-
                                            offset,
                                            0), (-
                                            offset,
                                            offset),
                                             (-2*
                                            offset,
                                            offset)
                                              ),
                                          ( (0,0),
                                             (0,
                                            offset),
                                             (offset
                                            , offset
                                            ), (
                                            offset,
                                            2*offset
                                            )      )
                                    ]


class OPiece(Piece):

        def __init__(self, x, y):

                super(OPiece, self).__init__(x,y)

                self.color = yellow

                self.positions =        [
                                          ( (0,0), (
                                            offset,
                                            0), (
                                            offset,
                                            offset),
                                             (0,
                                            offset)
                                                )
                                    ]


class TPiece(Piece):

        def __init__(self,x,y):
```

```
            super(TPiece, self).__init__(x, y)

            self.color = purple

            self.positions =        [
                                            ( (0,0),
                                               (0,
                                              offset),
                                                (-
                                              offset,
                                              offset),
                                                (offset
                                              , offset
                                              )
                                              ),
                                            ( (0,0),
                                               (0,
                                              offset),
                                                (offset
                                              , offset
                                              ), (0,
                                              2*offset
                                              )   ),
                                            ( (0,0), (-
                                              offset,
                                              0), (
                                              offset,
                                              0), (0,
                                              offset)

                                              ),
                                            ( (0,0),
                                               (0,
                                              offset),
                                                (-
                                              offset,
                                              offset),
                                                (0, 2*
                                              offset)
                                                ),
                                    ]

class IPiece(Piece):
```

```
        def __init__(self,x,y):
                super(IPiece, self).__init__(x, y)
                self.color = cyan
                self.positions =        [
                                                ( (0,0),
                                                   (0,
                                                  offset),
                                                   (0, 2*
                                                  offset),
                                                   (0, 3*
                                                  offset)
                                                       ),
                                                ( (0,0), (
                                                  offset,
                                                  0), (2*
                                                  offset,
                                                  0), (3*
                                                  offset,
                                                  0)
                                                  )
                                        ]

class LPiece(Piece):
        def __init__(self,x,y):
                super(LPiece, self).__init__(x, y)
                self.color = orange
                self.positions =        [
                                                ( (0,0),
                                                   (0,
                                                  offset),
                                                   (0, 2*
                                                  offset),
                                                   (offset
                                                  , 2*
                                                  offset)
```

```
                                                            ),
                                                        ( (0,0),
                                                            (0,
                                                            offset),
                                                             (offset
                                                            , 0),
                                                            (2*
                                                            offset,
                                                            0)
                                                                   )
                                                            ,
                                                        ( (0,0), (-
                                                            offset,
                                                            0), (0,
                                                            offset),
                                                             (0, 2*
                                                            offset)
                                                                  ),
                                                        ( (0,0),
                                                            (0,
                                                            offset),
                                                             (-
                                                            offset,
                                                            offset),
                                                             (-2*
                                                            offset,
                                                            offset)
                                                               ),
                                                ]

class JPiece(Piece):

        def __init__(self,x,y):

                super(JPiece, self).__init__(x, y)

                self.color = blue

                self.positions =        [
                                                        ( (0,0),
                                                            (0,
                                                            offset),
                                                             (0, 2*
                                                            offset),
                                                             (-
```

```
                                                offset,
                                                2*offset
                                                )
                                                       )
                                                ,
                                             ( (0,0), (
                                                offset,
                                                0), (2*
                                                offset,
                                                0), (2*
                                                offset,
                                                offset)

                                                ),
                                             ( (0,0), (
                                                offset,
                                                0), (0,
                                                offset),
                                                 (0, 2*
                                                offset)
                                                       )
                                                ,
                                             ( (0,0),
                                                 (0,
                                                offset),
                                                 (offset
                                                , offset
                                                ), (2*
                                                offset,
                                                offset)
                                                     ),
                                    ]



class Grid(object):

        def __init__(self):

                self.current = None
                self.next = None
                self.blocks = []

                self.next_piece()
```

```
self.total_cleared_lines = 0
self.cleared_lines = 0
self.level = 1
self.score = 0

self.lines_til_next_level = lines_per_level

self.delay = delay

#self.bitcount = 0

self.game_over = False

self.next_rect = pygame.Rect(150+125,
   90+100, 300, 300)

# this is for multiplying scores
self.multiplier =       {
                                0 : 0,

                                   # no
                                   lines
                                   cleared
                                1 : 40,

                                   # single
                                    line
                                   cleared
                                2 : 100,

                                   # two
                                   lines
                                   cleared
                                3 : 300,

                                   # three
                                   lines
                                   cleared
                                4 : 400

                                   # TETRIS
                        }

self.nfont = pygame.font.Font(None, 24)
self.largefont = pygame.font.Font(None, 40)
```

```
        self.smallfont = pygame.font.Font(None, 14)


    def random_piece(self):
        rekkefolge =
           [3,5,3,5,4,0,1,5,5,1,1,2,1,2,5,2,3,0,2,2,0,3,6,0,4,5,

        pcs =   {
                        0 : ZPiece(*next_point),
                        1 : OPiece(*next_point),
                        2 : SPiece(*next_point),
                        3 : TPiece(*next_point),
                        4 : IPiece(*next_point),
                        5 : LPiece(*next_point),
                        6 : JPiece(*next_point)
                }

        global bitnumber
        c = rekkefolge[bitnumber]
        bitnumber+=1
        return pcs[c]


    def next_piece(self):

        if not self.next:
                self.next = self.random_piece()
                self.current = self.random_piece()
        else:
                self.current = self.next
                self.next = self.random_piece()

        self.current.set_point(*start_point)

    def move_down(self):

        if(self.current.y < theight+100):
                self.current.y += offset

                if self.has_overlap():
                        self.current.y -= offset
                        self.current.mobile = False
        else:
                self.current.mobile = False
```

```
        if self.current.mobile == False and self.
            current.y == 100:
                self.game_over = True

# this is a hard drop - just go all the way down
    until you hit something
def drop_down(self):

        while(self.current.mobile): self.move_down
            ()

def move_right(self):
        if(self.current.mobile): self.current.x +=
            offset

        if self.has_overlap(): self.current.x -=
            offset

def move_left(self):
        if(self.current.mobile): self.current.x -=
            offset
        if self.has_overlap(): self.current.x +=
            offset

# TODO: wall kick logic needed
def rotate(self):
        self.current.flip()

        while self.has_overlap(): self.current.flip
            ()

# Saves individual blocks of the current piece -
    they become part of the grid
# Automatically check for lines
def remember_block_positions(self):

        self.blocks.extend(self.current.get_blocks
            ())
        self.next_piece()

        self.blocks = sorted(self.blocks, key=
            lambda block: block.y)

        self.check_for_lines()
```

```
        #for b in self.blocks: print str(b.y)+", "

def draw_blocks(self):

        for b in self.blocks:
                b.draw()

# check if current piece overlaps with walls or
   with other pieces
def has_overlap(self):

        blocks = self.current.get_blocks()

        for b in blocks:
                if b.x < 125 or b.x > right_edge
                   or b.y < 100 or b.y >
                   bottom_edge: return True

        # TODO: optimize the shit out of this
        for b in self.blocks:
                for c in self.current.get_blocks():
                        if b.x == c.x and b.y == c.
                           y: return True

        return False

# TODO: there ought to be a better way to do this
def check_for_lines(self):

        lines = {}

        for b in self.blocks:

                if b.y in lines:
                        lines[b.y] +=1
                else:
                        lines[b.y] = 1

        lines_to_be_destroyed = []

        for ln in lines:
                if lines[ln] == 10:
                        self.total_cleared_lines +=
                            1
                        self.cleared_lines += 1
```

```
                                lines_to_be_destroyed.
                                   append(ln)

                for l in lines_to_be_destroyed:
                        self.destroy_line(l)

                copy_of_blocks = self.blocks[:]

                if len(lines_to_be_destroyed) > 0:
                        self.collapse_hovering_blocks(min(
                           lines_to_be_destroyed), len(
                           lines_to_be_destroyed))

                        #for ln in lines_to_be_destroyed:
                                #self.
                                   collapse_hovering_blocks
                                   (ln, copy_of_blocks)

                self.calculate_score()


        def destroy_line(self, ln):

                # note the slice notation - I'm iterating
                   over a copy of self.blocks but removing
                # from the original
                for b in self.blocks[:]:
                        if b.y == ln: self.blocks.remove(b)



        def collapse_hovering_blocks(self, ln, total):
                """ drop down all the blocks that are
                   hovering """

                for i,b in enumerate(self.blocks):
                        if b.y < ln:
                                self.blocks[i].y += offset
                                   * total



        def block_overlaps(self, block, block_list):
```

```
        for b in block_list:
                if block.y == b.y and block.x == b.
                    x:
                        return True

        return False



def calculate_score(self):

        # score is calculated like so: M * ( N + 1
            ) where:
        # M is multiplier (see self.multiplier)
        # N is level (self.level)

        self.score += self.multiplier[self.
            cleared_lines] * (self.level + 1)

        tmp = self.lines_til_next_level - self.
            cleared_lines

        if tmp <= 0:
                self.level +=1
                self.lines_til_next_level =
                    lines_per_level + tmp

                self.delay -= self.level * 30

                if self.delay < 10: self.delay = 10

        else:
                self.lines_til_next_level = tmp

        self.cleared_lines = 0
        self.draw_text()


# redraws the whole UI - we should really be
    bliting this shit
def draw_ui(self):

        pygame.draw.rect(screen, gray, (125,100,
            right_edge+offset-125, bottom_edge+
            offset-100))
```

```
        for i in xrange(10): pygame.draw.line(
            screen, black, (i*offset+125, 100),
            (125+i*offset, theight+offset+100))
        for i in xrange(24): pygame.draw.line(
            screen, black, (125, i*offset+100), (
            twidth+125, i*offset+100))

        self.draw_text()


# I really dislike the default font in pygame
def draw_text(self):

        msg = self.nfont.render("NEXT PIECE:", 1,
            white)
        screen.blit(msg, (250+125, 50+100))

        pygame.draw.rect(screen, black, (250+125,
            200+100, 200, 100))

        sc = self.nfont.render("YOUR SCORE: ", 1,
            white)
        screen.blit(sc, (240+125, 250+100))
        sc = self.largefont.render(str(self.score)
            ,1,white)
        screen.blit(sc, (270+125, 270+100))

        #sc = self.nfont.render("LINES: " + str(
            self.total_cleared_lines), 1, white)
        #screen.blit(sc, (250, 280))

        #sc = self.nfont.render("LEVEL: " + str(
            self.level), 1, white)
        #screen.blit(sc, (250, 260))



        #msg = self.smallfont.render("Left, Right
            Arrow to move", 1, white)
        #screen.blit(msg, (230, 290))

        #msg = self.smallfont.render("Up Arrow to
            flip", 1, white)
        #screen.blit(msg, (230, 300))
```

```
            #msg = self.smallfont.render("Down Arrow to
                move downw", 1, white)
            #screen.blit(msg, (230, 310))

            #msg = self.smallfont.render("Space or
               Enter to drop down", 1, white)
            #screen.blit(msg, (230, 320))

            #msg = self.smallfont.render("Esc to pause,
                F1 for new game", 1, white)
            #screen.blit(msg, (230, 330))

            #screen.blit(self.smallfont.render("ver
               0.2", 1, white), (230, 350))

        def draw_game_over(self):

            msg = self.largefont.render("GAME OVER", 1,
                red)
            screen.blit(msg, (195+125,95+100))

            #msg2 = self.nfont.render("PRESS F1 TO PLAY
                AGAIN", 1, red)
            #screen.blit(msg2, (175, 130))



# magic
screen = pygame.display.set_mode(size)
pygame.key.set_repeat(100, 150)

#pygame.time.set_timer(USEREVENT+1, delay)

grid = Grid()
clock = pygame.time.Clock()

paused = False

time_elapsed = 0

while st == 1:
        currentTime = time.time()
        if grid.total_cleared_lines>lastlinecount:
```

```
            positivesound.play()
            lastlinecount=grid.total_cleared_lines
            counter+=1
    if not paused and not grid.game_over:

            time_elapsed += clock.tick()

            #print str(grid.delay)

            # timed block drop
            if time_elapsed > grid.delay:
                    time_elapsed = 0
                    grid.move_down()

            # check if a line was created and remove it
            grid.check_for_lines()

            # clear the next piece area
            pygame.draw.rect(screen, black, grid.
               next_rect)

            # draw the gridlines
            grid.draw_ui()

            # if the current piece is locked in place
               memorize it's position
            if not grid.current.mobile:
                    grid.remember_block_positions()

            # draw the memorized blocks
            grid.draw_blocks()


    for event in pygame.event.get():

            if event.type == pygame.QUIT: sys.exit()


            if event.type == KEYDOWN:

                    if event.key == K_ESCAPE: paused =
                       not paused

                    if event.key == K_F1:
                            grid = Grid()
```

```
                clock = pygame.time.Clock()
                counter = 0
                paused = False
                time_elapsed = 0

        if mode == True:
                if not paused and not grid.
                    game_over:
                        if event.key == K_d
                            : grid.move_left
                            ()
                        if event.key == K_a
                            : grid.
                            move_right()
                        if event.key == K_s
                            : grid.move_down
                            ()
                        if event.key == K_w
                            : grid.rotate()
                        if event.key ==
                            K_SPACE: grid.
                            drop_down()
                        if event.key ==
                            K_RETURN: grid.
                            drop_down()
                        if event.key == K_q
                            : mode = False
                        if event.key == K_1
                            : grid.delay =
                            100
                        if event.key == K_2
                            : grid.delay =
                            200
                        if event.key == K_3
                            : grid.delay =
                            300
                        if event.key == K_4
                            : grid.delay =
                            400
                        if event.key == K_5
                            : grid.delay =
                            500
                        if event.key == K_6
                            : grid.delay =
                            600
```

```
                                if event.key == K_7
                                   : grid.delay =
                                   700
                                if event.key == K_8
                                   : grid.delay =
                                   800
                                if event.key == K_9
                                   : grid.delay =
                                   900
                                if event.key == K_0
                                   : grid.delay =
                                   1000
                                if event.key ==
                                   K_UP: counter
                                   +=1;
                                   positivesound.
                                   play()
                                if event.key ==
                                   K_DOWN: counter
                                   -=1.5;
                                if event.key == K_m
                                   : counter-=4
                                if event.key == K_p
                                   : st=0
                if mode == False:
                        if not paused and not grid.
                           game_over:
                                if event.key == K_d
                                   : grid.
                                   move_right()
                                if event.key == K_a
                                   : grid.move_left
                                   ()
                                if event.key == K_s
                                   : grid.move_down
                                   ()
                                if event.key == K_w
                                   : grid.rotate()
                                if event.key ==
                                   K_SPACE: grid.
                                   drop_down()
                                if event.key ==
                                   K_RETURN: grid.
                                   drop_down()
                                if event.key == K_e
```

```
    : mode = True
if event.key == K_1
    : grid.delay =
    100
if event.key == K_2
    : grid.delay =
    200
if event.key == K_3
    : grid.delay =
    300
if event.key == K_4
    : grid.delay =
    400
if event.key == K_5
    : grid.delay =
    500
if event.key == K_6
    : grid.delay =
    600
if event.key == K_7
    : grid.delay =
    700
if event.key == K_8
    : grid.delay =
    800
if event.key == K_9
    : grid.delay =
    900
if event.key == K_0
    : grid.delay =
    1000
if event.key ==
    K_UP: counter
    +=1;
    positivesound.
    play()
if event.key ==
    K_DOWN: counter
    -=1.5;
if event.key == K_m
    : counter-=4
if event.key == K_p
    : st=0
```

```
if not paused and not grid.game_over:
        grid.current.draw()     # draw current
           piece
        grid.next.draw()        # draw the next
           piece
        if counter < 0:
                currentcolor = red
        else:
                currentcolor = green
        if counter >=11: counter = 11
        if counter <=-10: counter = -10
        pygame.draw.rect(screen, black,
           (370+125,100,200,1000), 0)
        screen.blit(perf, (405+125, 100+50))
        #screen.blit(timeshow, (250, 300))
        if grid.score >= highscore and
           highscorecount >=0:
                highscore+=630
                grid.delay -=100
                highscorecount-=1
                paused = not paused
                pygame.mixer.music.set_volume(0.1)
                levelupsound.play()
                if highscorecount==2:
                        pygame.draw.rect(screen,
                           black, (0,0,800,600), 0)
                        screen.blit(thirdplace1,
                           (150, 200))
                        pygame.display.flip()
                        sleep(0.2)
                        pygame.draw.rect(screen,
                           black, (0,0,800,600), 0)
                        screen.blit(thirdplace2,
                           (150, 200))
                        pygame.display.flip()
                        sleep(0.2)
                        pygame.draw.rect(screen,
                           black, (0,0,800,600), 0)
                        screen.blit(thirdplace1,
                           (150, 200))
                        pygame.display.flip()
                        sleep(0.2)
                        pygame.draw.rect(screen,
                           black, (0,0,800,600), 0)
                        screen.blit(thirdplace2,
```

```
                    (150, 200))
                pygame.display.flip()
                sleep(0.2)
                pygame.draw.rect(screen,
                   black, (0,0,800,600), 0)
                screen.blit(thirdplace1,
                    (150, 200))
                pygame.display.flip()
                sleep(1)
                pygame.draw.rect(screen,
                   black, (0,0,800,600), 0)
                pygame.display.flip()
        if highscorecount==1:
                pygame.draw.rect(screen,
                   black, (0,0,800,600), 0)
                screen.blit(secondplace1,
                    (150, 200))
                pygame.display.flip()
                sleep(0.2)
                pygame.draw.rect(screen,
                   black, (0,0,800,600), 0)
                screen.blit(secondplace2,
                    (150, 200))
                pygame.display.flip()
                sleep(0.2)
                pygame.draw.rect(screen,
                   black, (0,0,800,600), 0)
                screen.blit(secondplace1,
                    (150, 200))
                pygame.display.flip()
                sleep(0.2)
                pygame.draw.rect(screen,
                   black, (0,0,800,600), 0)
                screen.blit(secondplace2,
                    (150, 200))
                pygame.display.flip()
                sleep(0.2)
                pygame.draw.rect(screen,
                   black, (0,0,800,600), 0)
                screen.blit(secondplace1,
                    (150, 200))
                pygame.display.flip()
                sleep(1)
                pygame.draw.rect(screen,
                   black, (0,0,800,600), 0)
```

```
                                        pygame.display.flip()
                                if highscorecount==0:
                                        pygame.draw.rect(screen,
                                           black, (0,0,800,600), 0)
                                        screen.blit(inthelead1,
                                           (150, 200))
                                        pygame.display.flip()
                                        sleep(0.2)
                                        pygame.draw.rect(screen,
                                           black, (0,0,800,600), 0)
                                        screen.blit(inthelead2,
                                           (150, 200))
                                        pygame.display.flip()
                                        sleep(0.2)
                                        pygame.draw.rect(screen,
                                           black, (0,0,800,600), 0)
                                        screen.blit(inthelead1,
                                           (150, 200))
                                        pygame.display.flip()
                                        sleep(0.2)
                                        pygame.draw.rect(screen,
                                           black, (0,0,800,600), 0)
                                        screen.blit(inthelead2,
                                           (150, 200))
                                        pygame.display.flip()
                                        sleep(0.2)
                                        pygame.draw.rect(screen,
                                           black, (0,0,800,600), 0)
                                        screen.blit(inthelead1,
                                           (150, 200))
                                        pygame.display.flip()
                                        sleep(1)
                                        pygame.draw.rect(screen,
                                           black, (0,0,800,600), 0)
                                        pygame.display.flip()

                                pygame.mixer.music.set_volume(1)
                                paused = not paused
                        if highscorecount == 3:
                                screen.blit(timeFont.render("3rd
                                   place score:",1,white),
                                   (235+125,100+180))
                                screen.blit(myfont3.render(str(
                                   highscore),1,white),
                                   (270+125,100+200))
```

```
elif highscorecount == 2:
        screen.blit(timeFont.render("2nd
           place score:",1,white),
           (235+125,100+180))
        screen.blit(myfont3.render(str(
           highscore),1,white),
           (270+125,100+200))
elif highscorecount == 1:
        screen.blit(timeFont.render("1st
           place score:",1,white),
           (235+125,100+180))
        screen.blit(myfont3.render(str(
           highscore),1,white),
           (270+125,100+200))
else:
        screen.blit(myfont3.render("YOU ARE
            NOW",1,white),
           (200+125,100+170))
        screen.blit(myfont3.render("IN THE
           LEAD!",1,white),
           (200+125,200+100))
#else:
#               screen.blit(highscoreFont,
   (220,180))
#               screen.blit(myfont3.render(
   str(highscore),1,white), (270,200))
if currentTime - lastTime > 1:
        countdown-=1
        if countdown <= 0:
                countdown = 0
        lastTime = currentTime
m, s = divmod(countdown, 60)
#if m >= 1:
#       screen.blit(timeFont.render(str(m)
   ,1,white), (305, 300))
#       screen.blit(timeFont.render(":",1,
   white), (313, 300))
#       screen.blit(timeFont.render(str(s)
   ,1,white), (320, 300))
#else:
#       screen.blit(timeFont.render(str(s)
   ,1,white), (320, 300))
pygame.draw.rect(screen, currentcolor,
   (400+125,100+230,150,-counter*15), 0)
#screen.blit(img,(180,250))
```

```
        pygame.display.flip()

# draw game over message
if grid.game_over:
        #grid.draw_game_over()
        counter = 0
        highscorecount=3
        highscore = 1410
        if currentTime - lastTime > 1:
                countdown-=1
                if countdown <= 0:
                        countdown = 0
                lastTime = currentTime
        #pygame.display.flip()
        grid = Grid()
        clock = pygame.time.Clock()
        counter = 0
        paused = False
        time_elapsed = 0
```

Appendix I - Machine Learning, Classification of Postions and Leaning

Claasification Algorithm

```
    # coding=utf-8
import pandas as pd
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn import neighbors, datasets
from sklearn import svm
from sklearn.metrics import accuracy_score
from time import time
from time import sleep
from collections import Counter
from sklearn.model_selection import train_test_split
import serial





#Definisjon av underfunksjonene, som kalles p   i
   hovedscriptet:

#Gj r datasettet klart til maskinl ringa
def preProcessing(dataset):
    trainValidation_dataset = pd.DataFrame.from_csv(dataset
       )

    train, validation = train_test_split(
       trainValidation_dataset, test_size=0.2)

    # labels
    labels_train = train['Label']
    labels_validation = validation['Label']

    # features
    features_train = train.drop('Label', 1)
    features_validation = validation.drop('Label', 1)


    return features_train, features_validation,
       labels_train, labels_validation

#Trener til K-Ball-tree, men du kan lett bytte ut med en
   annen sklearn-algoritme
```

```
def model_train(features_train, features_validation,
    labels_train, labels_validation):
    # KD-Tree
    t0 = time()
    clfKD = neighbors.KNeighborsClassifier(8, weights='
        distance', algorithm='kd_tree')
    clfKD.fit(features_train, labels_train)
    validation = clfKD.predict(features_validation)
    print 'KD Initialize speed:', round((time() - t0) *
        1000, 2), 'ms'
    t0 = time()
    print 'KD Marginal 1000:', round((time() - t0) * 1000,
        2), 'ms'
    print 'Validation accuracy\t', accuracy_score(
        labels_validation, validation)
    mistakeList = []


#print validation, type(validation)
#print features_validation[1], type(features_validation[1])
    temp=[]
    for x in range (0,len(labels_validation.keys())):
    #print labels_validation[labels_validation.keys()[x]]
    #print labels_validation.keys()[x]
        if labels_validation[labels_validation.keys()[x]]!=
            validation[x]:
            temp=[labels_validation[labels_validation.keys
                ()[x]],validation[x]]
            mistakeList.append(temp)
    print mistakeList
    Overview = Counter(tuple(e) for e in mistakeList)
    print Overview
    for x in range(0, len(Overview.keys())):
        print 'True:\t', positionDict[Overview.keys()[x
            ][0]], '\t', 'Guessed:\t', positionDict[Overview
            .keys()[x][1]], '\t', 'Count:\t', Overview[
            Overview.keys()[x]]

    return clfKD


#Start kommunikasjon med arduino
def arduino_start(COM, BAUD):
    print 'arduino_start'
    arduino=serial.Serial(COM,BAUD)
```

```
    sleep(5)

    return arduino

#Les av verdi(er) fra arduino
def arduino_read(arduino):
    print 'arduino_read'
    #arduino write: for    starte opp arduino, som ellers
       er i en loop "while python not sending a command, do
        nothing..."
    arduino.write('1')
    sleep(0.97)

    #venter til arduino har sendt et signal  f  r  den leser
       ...
    while True:
        if arduino.inWaiting() > 0:
            data = arduino.readline()
            data = data.strip('\r\n')
            data = data.split(',')
            #data = [map(int, data) for x in data]
            #for i in range(data):
            #    data[i] = int(data)
            #break
            #ha passende antall sensorer, her 14
               trykksensorer og to capacitive sensorer (P1
               = pressure 1)
            datapoint = pd.Series(data, index =['P1', 'P2',
                'P3', 'P4', 'P5', 'P6', 'P7', 'P8', 'P9', '
               P10','P11','P12','P13','P14'] )
            #print datapoint
            return datapoint

def Prediction(datapoint, classifier):
    guess = classifier.predict(datapoint)[0]
    #print guess
    return guess
def file_read(number):
        dir = 'C:/Users/helge/Google Drive/Master/
           Experiment (1)/SensorData/mega/Rawdata/Mega_312_
           '
        megafil = open(dir + number + '.txt', 'r')
        data = megafil.readline()
        data = data.strip('\r\n')
        data = data.split(';')
```

```
        #tid = data.pop
            (1,2,3,4,5,6,7,8,9,10,11,12,13,14,15)
        data.pop(16)
        data.pop(15)
        data.pop(0)
        print data
        datapoint = pd.Series(data, index =['P1', 'P2', 'P3
            ', 'P4', 'P5', 'P6', 'P7', 'P8', 'P9', 'P10','
            P11','P12','P13','P14'] )
        return datapoint

#Hva forskjellige labels skal v re (de du bruker n r du
    lager dataset)
positionDict = {1: 'Helt tilbake med hodet', 2: 'Helt
    tilbake uten hodet', 3: 'fremoverlent ikke kors', 4:'
    slouching med hodet',5:'slouching uten hodet',6:'
    h yrelent',7:'venstrelent',}


#Her kj rer koden, med funksjonene som er definert over...
     du kan evt kj re de fra et eget script, og kalle p
    machineLearning."funksjonnavn"...

#Velg hvilket dataset du skal bruke til training
dataset = 'positions.csv'
features_train, features_validation, labels_train,
    labels_validation = preProcessing(dataset)
classifier = model_train(features_train,
    features_validation, labels_train, labels_validation)

#Sett inn COMPORT og baudrate (du finner det i arduinoIDE
    n r du velger port)
#arduino = arduino_start('COM4',115200)

#directory = 'C:/Users/helge/Google Drive/Master/ML/logdata
    /'
#filename = directory + raw_input('Define log data filename
    : ') + '.csv'
#start_time = time()
#number = raw_input('candidate number')
#dir = 'C:/Users/helge/Google Drive/Master/Experiment (1)/
    SensorData/mega/Rawdata/Mega_312_'
#megafile = open(dir + number + '.txt', 'r')
#For  n  prediction
#datapoint = arduino_read(arduino)
```

```
#prediction= Prediction(datapoint,classifier)
#print "Predicted position: ",positionDict[prediction]
countfile = 0
#for predicitons fram til du drar ut USB:

for i in range(6,40):
        directory = 'C:/Users/helge/Google Drive/Master/ML/
            logdata/'
        filename = directory + 'nypositions' + str(i) + '.
            csv'
        dir = 'C:/Users/helge/Google Drive/Master/
            Experiment (1)/SensorData/mega/Rawdata/Mega_312_
            '
        megafile = open(dir + str(i) + '.txt', 'r')
        count = 1
        countfile += 1
        #datapoint = file_read(number)
        for line in megafile:
                print countfile
                print count
                count+=1
                data = line
                if len(line) < 22:
                        prediction = 0
                else:
                        data = data.strip('\r\n')
                        data = data.split(';')
                        #tid = data.pop
                            (1,2,3,4,5,6,7,8,9,10,11,12,13,14,15)

                        data.pop(16)
                        data.pop(15)
                        data.pop(0)
                        #print data
                        datapoint = pd.Series(data, index
                            =['P1', 'P2', 'P3', 'P4', 'P5',
                            'P6', 'P7', 'P8', 'P9', 'P10','
                            P11','P12','P13','P14'] )
                        prediction = Prediction(datapoint,
                            classifier)
                #print "Predicted position: ", positionDict
                    [prediction]
                with open(filename, 'a') as file:
                        file.write(str(prediction) + '\n')
```

Learning Data Set Positions

DATAPOINT	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	Label
1	886	846	6	1002	998	917	1008	1008	1004	475	182	368	559	623	1
2	886	847	6	1003	998	918	1008	1008	1005	490	36	347	565	630	1
3	886	847	6	1003	997	919	1008	1009	1005	522	36	406	564	637	1
4	886	848	6	1003	998	921	1008	1008	1005	542	36	420	567	640	1
5	887	848	6	1003	998	923	1008	1009	1005	572	36	426	529	615	1
6	887	848	6	1004	998	927	1009	1009	1005	598	35	369	474	585	1
7	887	849	5	1003	998	929	1009	1009	1005	630	32	336	510	591	1
8	887	849	6	1004	999	933	1009	1010	1006	673	27	401	551	558	1
9	887	850	6	1004	999	934	1009	1010	1006	684	23	353	494	391	1
10	887	850	6	1004	999	935	1009	1010	1006	686	20	281	518	374	1
11	887	851	5	1004	999	935	1009	1010	1006	712	21	340	523	400	1
12	888	851	5	1004	999	934	1009	1010	1006	692	27	282	543	541	1
13	888	852	6	1004	999	932	1009	1010	1006	619	33	421	528	620	1
14	888	853	6	1004	999	930	1009	1010	1006	596	33	517	547	623	1
15	888	854	6	1004	999	921	1009	1010	1006	627	33	592	562	599	1
16	889	854	6	1004	999	922	1009	1010	1006	633	35	605	564	604	1
17	889	854	6	1004	999	922	1009	1010	1006	645	35	605	558	632	1
18	889	855	7	1004	999	923	1009	1010	1007	646	35	596	568	634	1
19	889	855	6	1004	999	922	1009	1010	1006	647	36	570	577	638	1
20	889	855	6	1004	1000	924	1009	1010	1006	649	36	521	566	642	1
21	889	855	6	1004	1000	923	1009	1010	1006	658	37	590	541	615	1
22	889	856	6	1004	1000	923	1009	1010	1006	657	37	581	563	641	1
23	889	856	6	1004	1000	923	1009	1010	1007	658	37	592	566	644	1
24	889	856	6	1004	1000	923	1009	1010	1007	658	37	597	573	649	1
25	889	857	6	1004	1000	925	1009	1011	1007	680	36	647	570	633	1
26	889	857	6	1004	1000	926	1009	1010	1007	688	36	640	535	607	1
27	889	858	7	1004	1000	926	1009	1011	1007	689	36	622	562	618	1
28	889	858	7	1004	1000	927	1010	1011	1007	685	36	632	569	639	1
29	889	858	6	1004	1000	928	1009	1011	1007	685	36	635	573	642	1
30	889	858	6	1004	1000	928	1009	1011	1007	684	37	633	574	652	1
31	890	858	6	1004	1000	928	1010	1011	1007	688	37	630	569	641	1
32	890	858	6	1004	1000	927	1010	1011	1007	687	37	631	538	622	1
33	890	859	6	1004	1000	928	1009	1011	1007	690	37	630	564	648	1
34	890	859	7	1004	1000	928	1010	1011	1007	690	37	631	568	658	1
35	890	859	7	1004	1000	929	1010	1011	1007	688	37	624	570	661	1
36	890	859	6	1004	1001	929	1010	1011	1007	691	39	622	563	657	1
37	890	859	6	1004	1001	929	1009	1011	1007	692	73	623	544	634	1
38	890	860	7	1004	1001	929	1010	1011	1007	698	39	644	571	652	1
39	890	860	7	1004	1000	928	1010	1011	1007	693	39	721	572	664	1
40	890	860	7	1004	1000	929	1010	1011	1007	686	37	771	574	668	1
41	890	860	7	1004	1000	928	1010	1011	1007	642	256	600	575	696	1
42	887	860	6	1003	1001	922	1010	1011	1007	647	498	522	560	680	1
43	889	861	6	1003	1001	923	1010	1011	1007	646	530	466	547	671	1
44	890	861	5	1003	1001	925	1010	1011	1007	640	591	208	561	683	1
45	890	861	6	1004	1001	927	1010	1011	1007	598	702	426	566	684	1
46	890	861	6	1003	1001	929	1010	1011	1007	671	632	531	580	682	1
47	890	861	6	1004	1001	930	1010	1011	1007	665	686	501	568	695	1
48	890	861	6	1004	1001	930	1010	1011	1007	666	708	479	567	686	1
49	890	861	6	1004	1001	931	1010	1011	1007	661	700	511	565	678	1
50	890	862	6	1004	1001	931	1009	1011	1007	656	698	517	549	672	1
51	889	863	5	999	999	910	1009	1010	1005	482	902	126	626	656	2
52	890	865	5	1000	1000	913	1009	1010	1005	331	907	123	616	724	2
53	890	865	5	1000	1001	919	1009	1011	1006	434	893	126	644	715	2
54	890	866	5	1001	1001	925	1009	1011	1006	512	835	123	614	716	2
55	891	866	5	1001	1001	926	1009	1011	1006	513	868	124	601	708	2
56	891	866	5	1001	1001	927	1009	1011	1007	516	875	123	603	699	2
57	891	866	5	1001	1001	928	1010	1011	1007	516	883	125	603	703	2
58	891	866	5	1001	1001	929	1010	1011	1007	515	882	126	631	709	2
59	891	866	5	1001	1001	930	1010	1011	1007	510	884	128	652	710	2
60	891	867	5	1001	1001	930	1010	1011	1007	516	886	126	636	711	2
61	891	867	5	1001	1001	930	1010	1011	1007	517	886	128	614	709	2
62	891	867	5	1002	1001	930	1010	1011	1007	511	889	126	643	710	2
63	892	867	5	1002	1002	930	1010	1011	1007	517	897	125	636	718	2
64	892	867	5	1002	1001	930	1010	1012	1007	490	896	128	652	724	2
65	892	867	5	1002	1001	930	1010	1012	1007	471	898	130	668	719	2
66	892	868	5	1002	1002	931	1010	1012	1007	481	898	130	639	709	2
67	892	868	5	1002	1002	931	1010	1012	1007	483	908	129	639	704	2
68	892	868	5	1002	1002	932	1010	1011	1007	487	906	133	653	714	2
69	892	868	5	1002	1002	932	1010	1011	1007	482	905	132	673	719	2
70	892	868	5	1002	1002	932	1010	1012	1007	483	905	134	676	722	2
71	892	868	5	1002	1002	933	1010	1012	1007	489	906	134	641	713	2
72	892	868	5	1002	1002	933	1010	1012	1007	489	906	136	635	708	2
73	892	868	5	1003	1002	934	1010	1012	1007	496	911	134	651	710	2
74	892	868	5	1003	1002	934	1010	1012	1007	493	910	134	664	716	2
75	892	868	5	1002	1002	934	1010	1011	1007	488	909	135	676	718	2
76	892	868	5	1003	1002	933	1010	1011	1007	273	911	133	611	703	2
77	893	868	5	1003	1002	934	1010	1012	1007	466	911	132	616	697	2
78	892	869	5	1003	1002	934	1010	1011	1007	484	913	133	642	708	2
79	893	869	5	1003	1002	935	1010	1012	1007	466	914	137	654	716	2
80	892	868	5	1003	1002	935	1010	1012	1007	471	904	138	660	701	2
81	892	869	5	1003	1002	936	1010	1011	1007	472	903	137	664	709	2
82	893	869	5	1003	1002	936	1010	1011	1007	414	919	135	610	703	2
83	893	869	5	1003	1002	936	1011	1012	1008	467	921	132	611	681	2

84	893	869	5	1003	1002	936	1011	1012	1008	476	923	131	620	693	2
85	893	869	5	1003	1002	937	1010	1012	1007	480	926	130	615	704	2
86	893	869	5	1003	1002	937	1010	1012	1007	480	923	130	621	712	2
87	890	872	5	1007	1006	988	1013	1014	1007	146	5	126	5	5	3
88	890	871	5	1007	1007	988	1013	1014	1007	240	5	129	5	5	3
89	890	871	5	1008	1007	988	1014	1014	1007	324	5	129	5	5	3
90	891	871	5	1008	1007	987	1014	1014	1008	349	5	130	5	5	3
91	891	871	5	1008	1006	987	1014	1014	1008	343	5	132	5	5	3
92	891	872	5	1008	1006	988	1014	1014	1008	356	5	132	5	5	3
93	891	872	5	1008	1007	988	1014	1014	1008	360	4	133	5	5	3
94	891	872	5	1008	1007	989	1014	1014	1008	373	5	134	5	5	3
95	891	872	5	1008	1007	989	1014	1014	1009	386	5	134	5	5	3
96	892	872	5	1008	1007	989	1014	1014	1009	395	5	135	5	5	3
97	892	871	5	1009	1007	989	1014	1014	1009	400	5	136	5	5	3
98	892	872	5	1008	1007	990	1014	1014	1009	375	5	137	5	5	3
99	892	872	5	1009	1007	990	1014	1014	1009	372	5	138	5	4	3
100	892	873	5	1008	1007	990	1014	1014	1009	372	5	139	5	5	3
101	892	873	5	1008	1007	989	1014	1014	1009	423	5	138	5	5	3
102	893	874	5	1009	1007	990	1014	1015	1009	422	5	138	5	5	3
103	893	875	5	1008	1007	990	1014	1014	1009	420	5	138	5	5	3
104	894	875	5	1008	1007	990	1014	1014	1009	425	5	139	5	5	3
105	894	876	5	1008	1008	991	1014	1015	1009	379	5	138	5	5	3
106	894	876	5	1008	1008	991	1014	1015	1009	356	5	137	5	5	3
107	894	876	5	1008	1008	991	1014	1014	1009	361	5	137	5	5	3
108	894	873	5	1009	1008	991	1014	1015	1010	365	5	137	5	5	3
109	895	874	5	1009	1008	991	1014	1015	1010	376	5	138	5	5	3
110	895	875	5	1009	1008	991	1014	1015	1009	349	5	137	5	5	3
111	895	875	5	1009	1008	992	1014	1015	1009	348	5	137	5	5	3
112	895	875	5	1008	1008	991	1014	1015	1010	357	5	137	5	5	3
113	895	875	5	1009	1008	991	1014	1014	1010	363	5	138	5	5	3
114	895	875	5	1009	1008	991	1014	1015	1010	351	5	137	5	5	3
115	895	876	5	1009	1008	991	1014	1015	1010	340	5	137	5	5	3
116	895	876	5	1009	1008	991	1014	1015	1010	334	5	138	5	5	3
117	895	876	5	1008	1008	991	1014	1014	1010	353	5	139	5	5	3
118	895	877	5	1009	1008	992	1014	1015	1010	343	5	139	5	4	3
119	895	877	5	1009	1008	992	1015	1015	1010	342	5	139	5	5	3
120	895	877	5	1009	1008	992	1014	1015	1010	340	5	139	5	5	3
121	895	877	5	1009	1008	992	1014	1015	1010	337	5	139	5	5	3
122	896	877	5	1009	1008	992	1014	1015	1010	342	5	139	5	5	3
123	895	877	5	1009	1008	992	1014	1015	1009	343	5	139	5	4	3
124	895	877	5	1009	1008	992	1014	1015	1010	342	5	139	5	5	3
125	895	877	5	1009	1008	992	1014	1014	1010	355	5	139	5	5	3
126	878	868	6	994	999	1012	1006	1007	427	6	976	421	924	837	4
127	878	868	6	994	1000	1013	1006	1008	431	6	974	430	924	840	4
128	878	869	6	994	1000	1013	1007	1008	432	7	974	522	921	836	4
129	879	869	6	995	1000	1013	1007	1008	437	6	975	524	921	835	4
130	879	869	6	995	1001	1012	1007	1008	440	6	976	528	923	841	4
131	879	870	6	995	1000	1013	1007	1008	441	6	975	549	925	844	4
132	879	870	6	996	1001	1013	1007	1008	445	6	978	494	922	841	4
133	879	870	6	996	1001	1013	1007	1008	446	6	979	525	914	822	4
134	880	871	6	996	1001	1013	1008	1009	450	7	980	533	920	835	4
135	880	871	6	996	1001	1013	1008	1008	453	7	980	552	923	844	4
136	880	871	6	996	1001	1013	1008	1009	455	7	980	555	926	847	4
137	880	871	6	996	1002	1013	1008	1009	457	6	980	548	926	849	4
138	880	871	6	997	1001	1013	1008	1009	457	7	980	554	921	842	4
139	881	872	6	997	1002	1013	1008	1009	460	7	978	577	917	830	4
140	881	872	7	997	1002	1013	1008	1009	465	7	980	588	920	834	4
141	881	872	7	997	1002	1013	1008	1009	465	6	977	602	920	845	4
142	881	872	6	997	1002	1013	1008	1010	466	7	976	605	921	848	4
143	881	871	7	997	1002	1013	1008	1009	468	7	977	619	923	847	4
144	881	872	7	998	1002	1013	1008	1009	467	7	977	599	928	843	4
145	881	872	6	997	1002	1013	1008	1009	467	7	978	620	926	845	4
146	881	872	6	998	1002	1013	1008	1010	468	7	978	617	930	847	4
147	881	872	7	998	1003	1013	1008	1010	466	6	978	631	931	850	4
148	881	872	7	998	1002	1013	1009	1010	464	7	976	633	926	848	4
149	881	872	6	998	1002	1013	1009	1010	463	7	976	632	919	845	4
150	882	873	6	998	1003	1013	1008	1010	465	7	977	631	924	849	4
151	882	873	6	998	1003	1013	1009	1010	466	7	978	559	926	853	4
152	882	873	7	998	1003	1013	1009	1010	469	7	977	641	923	848	4
153	882	873	6	998	1003	1013	1009	1010	467	7	978	636	914	837	4
154	882	873	7	998	1003	1013	1009	1010	470	7	980	633	921	841	4
155	882	873	7	999	1002	1013	1009	1010	473	7	980	652	924	848	4
156	882	873	7	998	1002	1013	1009	1010	473	7	980	659	927	850	4
157	882	873	7	998	1003	1013	1009	1010	474	7	979	659	923	849	4
158	880	871	5	999	1002	1013	1009	1010	486	5	994	115	944	872	5
159	880	871	5	999	1002	1013	1009	1011	487	5	995	113	946	874	5
160	880	871	5	1000	1002	1013	1009	1011	487	5	995	112	947	875	5
161	880	871	5	1000	1003	1013	1009	1010	488	5	995	112	946	876	5
162	881	871	5	1000	1002	1013	1009	1011	488	5	995	114	945	875	5
163	881	871	5	1000	1003	1013	1009	1010	489	5	996	115	941	871	5
164	882	872	5	1000	1003	1013	1009	1010	488	5	996	119	944	872	5
165	893	879	5	989	974	1012	1006	1005	472	5	1001	118	947	877	5
166	905	892	5	908	946	1012	945	969	373	5	1003	123	959	902	5
167	878	883	5	987	984	1013	995	997	370	5	999	121	958	891	5

168	868	847	5	991	989	1014	998	1000	337	5	998	125	958	858	5
169	855	837	5	992	991	1013	1000	1001	338	5	997	125	960	865	5
170	853	838	5	993	992	1013	1001	1002	364	5	997	127	962	870	5
171	856	842	5	994	993	1014	1002	1003	360	5	996	125	962	878	5
172	857	845	5	994	994	1014	1002	1004	366	5	997	123	960	878	5
173	859	848	5	995	994	1013	1003	1004	371	5	997	123	955	875	5
174	860	849	5	995	995	1014	1003	1004	374	5	998	127	958	878	5
175	861	850	5	995	995	1014	1003	1005	367	5	998	127	959	881	5
176	861	851	5	996	996	1014	1003	1005	375	5	998	128	960	883	5
177	862	851	5	996	996	1014	1003	1005	377	5	998	128	960	884	5
178	862	852	5	996	996	1014	1004	1005	385	5	999	130	956	878	5
179	863	853	5	996	997	1014	1004	1006	390	5	999	128	955	871	5
180	863	853	5	997	997	1014	1004	1006	391	5	999	131	958	878	5
181	864	854	5	997	997	1014	1004	1006	390	5	999	131	959	881	5
182	864	854	5	997	997	1014	1004	1006	392	5	998	131	960	884	5
183	864	854	5	997	997	1013	1005	1006	394	5	999	135	957	884	5
184	864	854	5	997	998	1014	1004	1006	400	5	999	137	949	880	5
185	865	855	5	997	998	1014	1005	1006	397	5	997	135	950	881	5
186	865	855	5	997	998	1014	1005	1006	397	5	997	137	956	885	5
187	865	856	5	998	998	1014	1005	1006	399	5	998	135	957	888	5
188	865	856	5	998	998	1014	1005	1007	401	5	997	133	955	886	5
189	866	857	5	998	998	1014	1005	1007	403	5	998	135	951	879	5
190	875	881	5	955	1002	1002	1013	1008	855	5	542	116	1005	5	6
191	873	883	5	905	1003	1001	1013	1008	825	5	542	112	1001	5	6
192	862	884	5	975	1003	1005	1013	1009	906	5	5	107	1005	5	6
193	867	882	5	983	1002	1003	1013	1008	808	5	798	111	1003	5	6
194	869	881	5	984	1002	1004	1013	1008	783	5	852	115	1003	5	6
195	869	881	5	985	1002	1004	1013	1008	717	5	874	114	1004	5	6
196	870	882	5	985	1002	1004	1013	1008	720	5	876	113	1001	5	6
197	871	882	5	985	1003	1005	1013	1008	648	5	772	117	1000	5	6
198	871	884	5	974	1004	1004	1013	1009	662	5	713	133	998	5	6
199	872	885	5	982	1004	1005	1013	1009	764	5	766	135	996	5	6
200	873	885	5	983	1004	1006	1013	1009	784	5	804	136	998	5	6
201	878	884	5	972	1004	1003	1013	1009	754	5	904	135	999	5	6
202	878	886	5	951	1005	1004	1013	1009	736	5	845	133	992	5	6
203	879	887	5	942	1005	1003	1013	1009	773	5	286	107	1003	5	6
204	883	888	5	952	1005	1003	1014	1010	769	5	203	109	1005	5	6
205	882	890	5	890	1005	1004	1013	1010	815	5	41	114	1004	5	6
206	877	891	5	901	1006	1004	1013	1010	852	5	301	115	996	5	6
207	847	892	5	969	1007	1007	1012	1010	906	5	449	121	978	5	6
208	871	892	5	968	1007	1007	1013	1011	876	5	221	125	992	5	6
209	879	892	5	968	1007	1006	1013	1010	814	5	379	125	1000	5	6
210	882	892	5	963	1007	1006	1013	1011	741	5	425	127	1001	5	6
211	883	893	5	963	1007	1006	1013	1010	725	5	473	128	999	5	6
212	884	893	5	962	1007	1007	1014	1011	724	5	224	129	997	5	6
213	884	893	5	964	1007	1007	1013	1010	744	5	43	135	999	5	6
214	884	894	5	961	1008	1007	1014	1011	747	5	17	132	1001	5	6
215	882	894	5	890	1008	1007	1014	1011	832	5	97	128	994	5	6
216	882	894	5	914	1008	1007	1014	1011	851	5	273	132	992	5	6
217	875	895	5	942	1008	1007	1013	1011	864	5	302	131	994	5	6
218	874	895	5	946	1008	1007	1013	1011	867	5	201	133	996	5	6
219	874	896	5	952	1008	1008	1014	1011	874	5	105	137	993	5	6
220	874	896	5	954	1008	1008	1014	1011	876	5	31	138	993	5	6
221	871	896	5	958	1008	1008	1014	1011	877	5	113	139	994	5	6
222	870	880	5	1004	907	987	1008	1010	921	5	5	125	5	1009	7
223	874	879	5	1005	939	988	1008	1011	923	5	11	126	5	1009	7
224	876	877	5	1005	941	987	1009	1011	926	5	7	126	5	1009	7
225	878	876	5	1005	945	985	1009	1011	920	5	5	126	5	1009	7
226	880	874	5	1005	948	987	1009	1011	927	5	11	126	5	1010	7
227	881	872	5	1006	949	989	1010	1012	926	5	73	125	5	1009	7
228	881	872	5	1006	952	992	1010	1012	921	5	201	126	5	1008	7
229	880	873	5	1006	956	994	1010	1012	922	5	170	123	5	1008	7
230	882	865	5	1006	960	996	1010	1011	916	5	27	119	5	1008	7
231	882	868	5	1006	960	993	1010	1012	922	5	21	123	5	1008	7
232	883	867	5	1006	960	998	1010	1012	915	5	18	121	5	1008	7
233	883	867	5	1006	960	998	1010	1012	918	5	6	124	5	1008	7
234	884	866	5	1007	963	999	1011	1012	925	5	5	126	5	1007	7
235	885	867	5	1007	968	999	1010	1012	923	5	108	130	5	1005	7
236	882	865	5	1007	959	1000	1010	1012	923	5	206	126	5	1006	7
237	887	834	5	1007	968	1000	1011	1012	909	5	357	125	5	1003	7
238	890	826	5	1008	971	1001	1011	1012	915	5	386	124	5	1005	7
239	888	795	5	1008	986	1003	1011	1009	937	5	250	123	5	999	7
240	888	815	5	1008	981	1003	1011	1010	939	5	207	128	5	1002	7
241	883	824	5	1008	978	1002	1011	1011	932	5	173	129	5	1007	7
242	883	829	5	1008	968	1003	1011	1011	913	5	22	132	5	1007	7
243	886	831	5	1008	969	1003	1011	1012	925	5	27	133	5	1006	7
244	887	832	5	1008	971	1003	1011	1012	929	5	207	134	5	1005	7
245	887	834	5	1009	968	1003	1011	1012	931	5	237	136	5	1006	7
246	887	835	5	1009	968	1003	1011	1012	935	5	268	136	5	1007	7
247	888	835	5	1009	969	1003	1011	1012	937	5	261	137	5	1007	7
248	869	777	5	949	966	579	1010	1011	980	5	257	149	286	4	1
249	828	783	5	960	974	586	1010	1012	983	126	469	211	389	5	1
250	845	789	5	963	975	586	1010	1012	985	129	492	248	415	5	1
251	850	792	5	964	975	591	1010	1012	986	133	540	250	428	5	1

252	853	794	5	964	975	592	1011	1012	987	136	571	248	443	5	1
253	855	795	5	965	975	592	1011	1012	987	136	590	251	451	5	1
254	857	797	5	965	976	597	1011	1012	988	140	592	266	453	5	1
255	859	798	5	966	976	592	1011	1012	988	142	611	270	461	5	1
256	859	799	5	966	976	599	1011	1012	988	142	616	278	466	5	1
257	860	800	5	966	976	599	1011	1013	989	144	641	275	474	5	1
258	861	800	5	967	976	597	1011	1013	988	188	653	293	478	5	1
259	862	801	5	967	976	601	1011	1012	989	227	721	297	477	5	1
260	862	801	5	967	976	600	1011	1013	988	263	716	304	472	5	1
261	863	802	5	967	977	598	1011	1013	989	270	667	307	477	5	1
262	864	803	5	967	976	612	1011	1013	989	132	633	297	444	5	1
263	864	803	5	968	976	607	1011	1013	989	148	644	285	461	5	1
264	864	803	5	968	977	604	1011	1013	989	148	650	297	461	5	1
265	865	804	5	968	977	610	1011	1013	989	163	666	301	464	5	1
266	865	804	5	968	977	612	1011	1013	989	283	772	287	480	5	1
267	865	804	5	968	977	613	1011	1013	989	325	798	289	504	5	1
268	866	805	5	968	977	611	1011	1013	989	322	791	292	507	5	1
269	866	805	5	968	977	615	1012	1013	989	329	794	286	500	5	1
270	867	805	5	969	977	615	1011	1013	990	315	796	292	497	5	1
271	867	805	5	969	977	617	1011	1013	990	309	796	296	494	5	1
272	867	805	5	969	977	620	1011	1013	989	312	782	285	497	5	1
273	868	806	5	969	977	623	1012	1013	990	309	776	299	496	5	1
274	868	806	5	969	977	622	1012	1013	989	284	785	308	489	5	1
275	868	806	5	969	977	624	1011	1013	989	291	780	308	493	5	1
276	868	806	5	970	977	624	1012	1013	989	299	788	310	489	5	1
277	869	806	5	969	977	626	1012	1013	990	302	796	304	490	5	1
278	869	807	5	970	978	625	1011	1013	990	279	796	307	493	5	1
279	869	807	5	970	978	627	1011	1013	990	273	773	303	479	5	1
280	869	807	5	970	978	627	1012	1013	990	284	781	309	490	5	1
281	871	809	4	972	980	638	1012	1013	990	297	543	121	469	4	2
282	871	809	5	973	980	646	1012	1013	990	306	575	127	466	5	2
283	871	809	5	973	980	647	1012	1013	991	323	612	124	480	5	2
284	871	809	5	973	980	651	1012	1013	991	330	626	121	482	5	2
285	872	809	5	973	980	649	1012	1013	991	328	640	122	483	5	2
286	872	809	5	973	980	643	1012	1014	991	329	638	122	487	5	2
287	872	809	5	973	980	646	1012	1013	992	328	654	126	487	10	2
288	872	809	5	973	980	649	1012	1013	992	329	660	127	490	5	2
289	872	810	5	973	981	649	1012	1013	992	335	693	127	490	5	2
290	872	809	5	973	980	644	1012	1013	992	345	709	125	490	5	2
291	872	810	4	974	981	646	1012	1013	992	307	629	125	492	4	2
292	873	810	5	974	981	645	1012	1013	992	297	590	127	491	5	2
293	873	810	5	974	981	644	1012	1013	992	294	589	128	493	5	2
294	873	810	5	974	981	648	1012	1013	992	301	612	128	495	5	2
295	873	810	5	974	981	647	1012	1013	993	312	606	125	473	88	2
296	873	810	5	974	981	650	1012	1013	992	309	599	128	474	96	2
297	873	811	5	974	981	649	1012	1013	993	323	625	130	477	104	2
298	873	811	4	974	981	648	1012	1013	993	330	652	131	482	115	2
299	874	811	5	975	981	650	1012	1014	993	333	649	132	484	116	2
300	874	811	5	975	981	647	1012	1013	992	338	651	133	484	119	2
301	874	811	5	975	981	650	1012	1013	993	341	655	129	486	124	2
302	874	811	5	975	981	648	1012	1013	992	342	677	129	488	113	2
303	874	812	5	975	981	650	1012	1014	993	345	697	130	490	23	2
304	874	811	5	975	981	648	1012	1013	992	345	688	126	494	117	2
305	874	812	5	975	981	648	1012	1013	993	343	684	127	491	135	2
306	874	812	5	975	981	649	1012	1013	993	343	688	126	490	137	2
307	874	812	5	975	981	648	1012	1014	993	349	693	127	493	138	2
308	874	812	5	975	981	647	1012	1013	993	350	693	127	493	147	2
309	874	812	5	975	981	650	1012	1013	993	351	723	126	493	127	2
310	874	812	5	975	981	651	1012	1014	993	356	753	124	496	102	2
311	874	812	5	975	981	649	1012	1013	993	351	723	125	494	144	2
312	874	812	5	975	981	647	1012	1013	994	348	717	128	488	158	2
313	874	812	5	975	982	655	1012	1013	994	357	751	129	493	5	2
314	874	812	5	975	982	654	1012	1013	994	360	683	130	501	141	2
315	875	812	5	976	982	652	1012	1013	994	356	676	131	493	142	2
316	875	812	5	976	982	652	1012	1014	994	344	692	132	492	161	2
317	875	812	5	976	982	653	1012	1014	994	350	674	132	486	158	2
318	875	812	5	976	982	654	1012	1014	994	351	669	133	487	160	2
319	875	812	5	976	982	651	1012	1014	994	354	689	132	490	165	2
320	875	812	5	976	982	651	1012	1013	994	352	688	132	493	153	2
321	875	812	5	976	982	652	1012	1014	994	356	711	131	489	108	2
322	875	812	5	976	982	653	1012	1014	994	362	691	131	498	136	2
323	875	812	5	976	982	654	1013	1013	994	359	666	132	494	150	2
324	879	833	5	988	989	686	1012	1011	1007	5	4	97	4	4	3
325	880	835	4	989	989	693	1012	1012	1007	4	4	99	4	4	3
326	881	837	4	989	989	691	1012	1012	1007	5	4	99	4	4	3
327	881	839	5	990	990	693	1012	1012	1007	5	4	98	4	4	3
328	882	841	4	990	990	696	1012	1012	1008	5	4	98	4	4	3
329	882	837	4	990	989	696	1012	1012	1008	4	5	97	4	4	3
330	882	840	4	991	990	701	1012	1012	1008	5	4	97	4	4	3
331	883	844	5	991	990	699	1012	1012	1007	5	4	100	4	4	3
332	883	844	5	991	990	703	1012	1012	1008	4	4	101	4	4	3
333	884	846	5	991	990	705	1012	1012	1008	5	4	101	5	4	3
334	886	843	4	991	991	704	1012	1012	1008	5	5	103	4	4	3
335	885	841	5	991	991	703	1012	1012	1008	5	4	104	5	4	3

336	884	839	4	991	990	705	1012	1012	1008	5	4	103	5	5	3
337	885	843	5	992	991	703	1012	1012	1008	5	5	103	5	4	3
338	886	844	4	991	991	705	1012	1012	1008	5	4	104	4	4	3
339	886	842	5	992	990	705	1012	1012	1008	5	4	104	5	5	3
340	886	843	5	992	990	707	1013	1012	1008	5	4	104	4	5	3
341	886	844	5	992	990	704	1012	1012	1008	5	4	104	5	5	3
342	885	842	5	992	990	708	1012	1012	1008	5	4	104	5	4	3
343	885	844	4	992	990	710	1013	1012	1008	5	4	105	4	4	3
344	886	841	4	992	990	714	1013	1012	1008	5	5	106	4	4	3
345	886	818	4	992	990	708	1013	1012	1008	5	4	105	4	4	3
346	883	801	4	992	990	709	1013	1012	1008	5	4	106	4	4	3
347	882	821	4	992	989	704	1013	1012	1009	5	4	106	4	4	3
348	882	822	4	992	989	700	1013	1013	1008	5	5	106	4	4	3
349	882	820	4	992	989	703	1013	1012	1008	4	4	106	5	4	3
350	883	820	4	991	989	700	1013	1012	1008	5	4	106	4	4	3
351	883	816	4	992	988	703	1013	1013	1008	5	4	106	4	4	3
352	884	819	4	992	988	698	1013	1012	1008	5	4	106	5	4	3
353	885	824	5	993	987	698	1013	1012	1009	5	4	107	4	4	3
354	886	828	4	992	988	702	1013	1012	1009	5	4	107	4	5	3
355	886	828	5	993	988	701	1013	1013	1009	7	4	106	4	4	3
356	859	810	5	732	972	1011	979	1002	352	5	974	206	736	852	4
357	860	811	5	732	973	1011	978	1003	353	5	975	213	739	853	4
358	861	812	5	735	974	1011	979	1003	364	5	974	211	736	853	4
359	861	813	5	725	974	1011	979	1003	362	5	975	218	720	847	4
360	862	814	5	737	975	1011	979	1002	360	5	976	212	733	851	4
361	862	814	5	735	975	1011	980	1003	365	5	977	210	735	852	4
362	863	814	5	744	975	1011	980	1003	352	5	978	215	736	853	4
363	863	816	5	742	976	1011	980	1002	357	5	979	213	738	853	4
364	863	816	5	744	976	1011	981	1003	356	5	980	212	739	852	4
365	864	817	5	744	976	1011	981	1002	351	5	981	207	738	852	4
366	864	817	5	735	976	1011	982	1003	344	5	982	206	737	851	4
367	865	818	5	722	976	1011	981	1003	357	5	984	203	730	850	4
368	865	818	5	737	976	1011	982	1003	353	5	983	200	738	852	4
369	865	818	5	733	977	1011	982	1003	360	5	982	197	739	852	4
370	866	819	5	732	977	1011	982	1003	355	5	982	204	738	852	4
371	866	819	5	734	977	1011	982	1003	354	5	983	209	741	852	4
372	866	820	5	735	977	1011	981	1003	350	5	982	215	739	852	4
373	866	820	5	734	977	1011	982	1003	364	5	983	219	736	851	4
374	866	821	5	723	978	1011	982	1003	359	5	985	221	711	846	4
375	867	821	5	720	978	1011	982	1003	361	5	986	221	698	841	4
376	867	821	5	729	978	1011	982	1003	353	5	986	221	733	846	4
377	867	822	5	737	978	1011	982	1003	355	5	986	209	736	848	4
378	867	821	5	740	978	1011	982	1003	355	5	987	210	738	849	4
379	867	822	5	742	978	1011	983	1003	350	5	987	213	739	849	4
380	867	822	5	744	978	1011	983	1003	350	5	987	208	740	849	4
381	868	822	5	746	978	1011	983	1003	346	5	987	217	739	849	4
382	868	823	5	747	978	1011	983	1003	349	5	987	214	740	849	4
383	868	823	5	741	978	1011	983	1003	336	5	987	216	739	849	4
384	868	823	5	740	979	1011	983	1003	345	5	988	216	732	848	4
385	869	824	5	735	979	1011	984	1003	346	5	989	216	716	846	4
386	869	824	5	742	978	1011	984	1003	348	5	988	217	733	848	4
387	869	824	5	748	979	1012	984	1003	357	5	988	214	741	848	4
388	869	824	5	750	979	1011	984	1004	355	5	987	217	742	849	4
389	870	826	5	734	979	1010	984	1003	346	5	995	136	732	846	5
390	870	826	5	742	979	1011	984	1003	354	5	994	134	742	851	5
391	870	826	5	745	979	1011	984	1003	355	5	995	135	745	854	5
392	870	826	5	746	979	1010	984	1003	351	5	994	137	746	855	5
393	871	827	5	744	979	1011	984	1003	348	5	994	139	748	855	5
394	870	827	5	746	979	1011	985	1003	352	5	995	138	746	854	5
395	870	827	5	740	979	1010	985	1003	351	5	995	138	743	850	5
396	871	827	5	742	979	1011	985	1004	337	5	996	138	732	850	5
397	871	828	5	752	979	1011	985	1004	342	5	995	140	746	856	5
398	871	828	5	756	979	1011	985	1003	345	5	995	140	748	856	5
399	871	828	5	753	979	1011	985	1003	345	5	995	141	751	856	5
400	871	828	5	753	979	1011	984	1004	350	5	996	141	750	857	5
401	871	828	5	752	979	1011	985	1004	347	5	996	141	748	854	5
402	871	828	5	744	980	1010	985	1004	353	5	996	141	740	850	5
403	871	828	5	746	980	1011	985	1004	352	5	996	140	742	853	5
404	872	828	5	751	979	1011	985	1004	349	5	996	140	750	857	5
405	872	829	5	751	980	1011	985	1004	349	5	996	142	751	857	5
406	872	829	5	749	980	1011	985	1004	344	5	996	139	751	857	5
407	872	829	5	750	980	1011	985	1004	357	5	996	138	751	857	5
408	872	829	5	748	980	1011	985	1004	353	5	996	139	746	854	5
409	872	829	5	737	981	1011	985	1004	353	5	997	139	722	849	5
410	872	830	5	752	980	1011	985	1004	359	5	996	139	747	856	5
411	872	829	5	754	980	1011	985	1003	355	5	996	139	751	857	5
412	873	830	5	756	980	1011	985	1004	361	5	996	139	751	858	5
413	873	830	5	756	980	1011	986	1004	357	5	996	139	751	858	5
414	873	830	5	754	980	1011	986	1003	357	5	996	139	751	858	5
415	873	830	5	746	981	1011	985	1004	358	5	997	139	743	853	5
416	873	830	5	739	981	1010	986	1004	359	5	997	140	708	849	5
417	873	830	5	741	981	1011	986	1003	357	5	997	140	741	854	5
418	873	830	5	750	980	1011	985	1004	353	5	997	140	749	858	5
419	873	831	5	753	980	1011	986	1004	355	5	997	140	751	858	5

420	873	831	5	757	981	1011	986	1004	353	5	997	140	751	858	5
421	873	831	5	752	980	1010	986	1004	350	5	997	140	752	858	5
422	873	831	5	751	981	1011	986	1004	352	5	997	140	751	857	5
423	874	831	5	741	981	1011	986	1004	357	5	998	140	727	850	5
424	874	831	5	742	981	1011	986	1004	352	5	998	141	733	851	5
425	873	831	5	749	981	1011	986	1004	354	5	997	141	748	858	5
426	839	802	4	632	989	852	1007	1011	302	5	538	131	777	5	6
427	844	806	4	617	991	852	1007	1011	310	4	118	131	818	4	6
428	848	805	4	643	992	851	1007	1011	302	4	276	133	828	4	6
429	850	808	5	643	992	850	1007	1011	305	5	354	134	826	4	6
430	851	808	4	655	992	848	1008	1011	305	5	398	134	822	4	6
431	851	807	4	655	993	845	1007	1011	297	4	407	135	827	5	6
432	852	808	5	648	993	846	1008	1011	292	4	368	135	833	5	6
433	853	808	5	654	993	844	1008	1011	307	4	389	135	833	4	6
434	853	808	4	653	993	841	1008	1011	309	4	431	137	832	4	6
435	854	805	4	670	993	836	1008	1011	316	4	456	137	835	4	6
436	854	805	4	666	993	822	1008	1012	323	4	456	137	835	4	6
437	854	804	5	664	993	863	1008	1011	328	4	470	137	833	5	6
438	854	805	4	667	993	863	1008	1011	334	4	483	137	834	4	6
439	855	803	5	670	994	863	1008	1012	334	5	482	138	837	4	6
440	855	803	5	688	994	862	1008	1011	327	4	494	137	838	5	6
441	855	804	5	691	994	860	1008	1012	327	4	506	138	839	5	6
442	856	805	4	696	994	860	1008	1011	321	4	464	139	841	4	6
443	856	807	5	687	994	855	1009	1012	323	4	493	139	841	5	6
444	856	806	5	691	995	856	1008	1012	337	4	504	139	835	5	6
445	856	807	5	689	995	858	1008	1012	345	5	524	139	831	4	6
446	856	807	4	681	995	860	1009	1012	338	4	548	139	831	4	6
447	856	807	4	687	995	858	1009	1012	343	4	535	139	838	5	6
448	857	797	5	674	994	850	1008	1012	335	4	661	140	823	5	6
449	857	802	5	681	994	853	1008	1012	334	4	591	140	833	4	6
450	857	803	4	684	994	853	1009	1012	324	4	585	140	834	5	6
451	857	804	5	689	995	854	1008	1012	319	4	563	140	836	5	6
452	857	804	5	690	995	855	1009	1012	317	5	552	140	836	5	6
453	857	805	4	692	995	855	1009	1012	311	4	565	141	833	4	6
454	858	805	5	692	996	857	1009	1012	309	4	577	141	827	4	6
455	857	806	5	694	996	858	1009	1012	307	4	617	141	815	4	6
456	858	806	5	690	996	856	1008	1012	309	4	604	141	827	5	6
457	858	806	4	698	996	857	1009	1012	307	5	618	141	828	4	6
458	858	807	5	698	996	856	1009	1012	316	5	594	141	832	5	6
459	870	812	4	832	826	906	1006	1003	281	4	5	136	4	1006	7
460	875	781	4	762	902	902	1006	1006	326	4	4	138	5	1006	7
461	880	739	4	806	928	913	1008	1005	386	4	4	141	4	995	7
462	882	746	4	811	941	911	1008	1006	371	5	4	141	4	999	7
463	883	751	4	806	949	915	1008	1006	371	5	4	142	4	999	7
464	884	753	4	805	953	918	1008	1006	371	4	4	142	4	999	7
465	886	756	4	809	956	919	1008	1007	385	5	4	142	4	999	7
466	886	757	4	800	958	921	1008	1007	383	4	4	142	4	999	7
467	886	757	4	783	961	922	1008	1007	383	4	4	142	4	998	7
468	887	760	4	785	962	921	1008	1008	385	4	5	142	4	997	7
469	888	761	4	786	964	918	1008	1007	383	4	4	142	4	998	7
470	888	764	4	791	964	919	1008	1008	389	4	4	142	4	998	7
471	889	764	4	796	965	919	1008	1008	386	4	4	144	4	998	7
472	889	765	4	796	966	920	1008	1008	389	5	4	144	4	998	7
473	889	765	4	795	966	920	1008	1008	396	4	4	144	4	999	7
474	890	766	4	796	967	920	1009	1008	393	4	4	144	4	999	7
475	890	768	4	796	968	922	1008	1008	388	5	4	144	4	999	7
476	890	768	4	798	968	919	1009	1008	394	4	4	144	5	999	7
477	891	768	4	797	969	920	1008	1008	389	5	4	144	4	999	7
478	891	768	4	800	969	921	1009	1008	386	5	5	145	4	999	7
479	891	769	4	798	970	924	1009	1008	386	4	4	145	4	1000	7
480	891	770	4	797	970	924	1009	1008	385	5	4	145	4	1000	7
481	891	770	4	796	970	925	1009	1008	386	5	4	145	4	999	7
482	892	771	4	794	971	925	1009	1008	385	4	4	145	5	999	7
483	892	770	4	796	971	925	1009	1009	388	5	4	146	5	999	7
484	892	771	4	797	971	924	1009	1008	389	5	5	146	4	1000	7
485	892	771	4	799	972	925	1009	1009	385	5	4	146	5	999	7
486	892	771	4	798	972	925	1009	1009	384	4	4	146	5	999	7
487	892	772	4	797	972	925	1009	1009	384	5	5	146	4	999	7
488	893	772	4	797	973	926	1009	1009	383	4	4	146	5	1000	7
489	893	773	4	794	973	926	1009	1009	382	4	4	146	4	999	7
490	893	774	4	794	973	926	1009	1009	380	4	5	146	4	999	7
491	893	773	4	789	974	926	1008	1009	380	4	4	145	5	1000	7
492	893	774	4	786	974	925	1009	1009	378	5	4	145	4	1000	7

Learning Data Set Leaning

DATAPOINT	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	Label
1	853	847	4	950	979	454	1010	1012	988	5	311	72	270	4	1
2	854	851	4	949	980	448	1010	1012	989	5	547	83	263	4	1
3	856	854	5	948	980	451	1010	1012	990	5	572	86	268	4	1
4	857	857	5	947	981	458	1010	1012	991	5	589	91	252	5	1
5	859	859	4	946	981	459	1010	1012	991	4	593	96	252	4	1
6	860	860	5	947	981	468	1010	1012	992	5	594	98	244	5	1
7	861	861	5	946	981	471	1011	1012	992	5	605	102	244	4	1
8	862	862	4	946	982	470	1011	1012	993	5	608	100	240	4	1
9	862	863	5	946	982	479	1011	1013	993	5	624	105	244	4	1
10	863	864	5	947	982	490	1011	1012	993	5	641	107	265	4	1
11	863	865	5	946	982	494	1011	1013	993	5	661	109	259	4	1
12	864	866	5	946	983	500	1011	1013	994	5	659	106	242	5	1
13	864	866	5	947	983	494	1011	1013	994	5	638	106	246	5	1
14	865	867	5	948	983	497	1011	1013	994	5	642	108	245	4	1
15	865	867	5	949	982	496	1011	1013	994	5	645	111	242	4	1
16	865	868	5	948	983	499	1011	1013	994	4	645	113	242	4	1
17	866	868	5	949	983	502	1011	1013	994	5	645	115	244	4	1
18	866	869	5	949	983	505	1011	1013	994	5	646	113	243	5	1
19	866	869	5	949	983	503	1011	1013	995	5	648	114	246	5	1
20	866	869	5	949	983	505	1011	1013	995	5	647	115	247	5	1
21	867	869	5	950	983	508	1011	1013	995	5	648	117	248	9	1
22	867	870	5	949	983	507	1011	1013	995	5	647	120	244	9	1
23	867	870	5	949	983	506	1011	1013	995	5	647	120	246	8	1
24	867	871	5	948	983	502	1011	1013	995	5	620	115	247	5	1
25	868	871	5	949	983	507	1011	1013	995	5	635	120	242	5	1
26	868	871	5	950	984	515	1011	1013	995	5	644	121	247	9	1
27	868	871	5	949	984	518	1011	1013	995	5	633	121	240	16	1
28	868	871	5	950	984	519	1011	1013	995	5	638	123	256	20	1
29	869	872	5	949	984	519	1011	1013	995	5	630	121	267	5	1
30	869	872	4	949	984	520	1011	1013	995	5	593	123	257	5	1
31	869	872	5	948	985	521	1011	1013	995	5	632	119	249	5	1
32	870	874	5	952	985	536	1011	1013	996	5	557	121	334	178	2
33	871	874	5	952	985	539	1012	1013	996	5	619	127	340	174	2
34	871	875	5	953	985	541	1011	1013	996	5	636	133	342	180	2
35	871	875	5	953	985	546	1011	1013	996	5	636	134	338	177	2
36	871	875	5	953	985	548	1012	1013	996	5	636	134	345	183	2
37	871	875	5	953	985	551	1012	1013	996	5	643	134	352	187	2
38	872	875	5	953	985	554	1011	1013	996	5	645	135	355	186	2
39	872	875	5	953	986	552	1011	1013	996	5	571	133	339	186	2
40	872	875	5	954	985	557	1012	1013	996	5	632	133	356	181	2
41	872	876	5	954	986	561	1012	1013	997	5	657	136	363	183	2
42	872	876	5	953	986	563	1011	1014	997	5	671	135	366	176	2
43	872	876	5	954	986	562	1012	1014	997	5	655	135	356	188	2
44	872	876	5	954	986	563	1012	1013	997	5	626	134	369	187	2
45	872	876	5	954	986	561	1011	1013	997	5	636	136	371	185	2
46	872	877	5	954	986	561	1011	1013	997	5	650	136	368	187	2
47	873	877	5	954	986	562	1011	1014	997	5	655	135	370	187	2
48	873	877	5	954	986	562	1012	1014	997	5	658	136	366	188	2
49	873	877	5	953	986	565	1012	1013	997	5	675	135	371	178	2
50	873	877	5	954	986	563	1012	1013	997	5	690	136	375	184	2
51	873	877	5	954	986	565	1012	1013	997	5	655	137	363	188	2
52	873	877	5	954	986	562	1012	1013	997	5	591	140	367	187	2
53	873	877	5	954	986	568	1012	1013	997	5	590	141	342	181	2
54	873	878	5	954	987	567	1012	1014	997	5	603	140	345	186	2
55	874	878	5	954	986	570	1012	1013	997	5	603	141	335	185	2
56	873	878	5	954	986	568	1012	1013	998	5	628	142	343	180	2
57	873	878	5	955	987	571	1012	1013	998	5	631	142	348	187	2
58	874	878	5	955	987	572	1012	1014	997	5	652	141	376	185	2
59	874	878	5	954	987	578	1012	1013	997	5	662	142	380	181	2
60	874	878	5	955	987	578	1012	1014	998	5	639	143	375	191	2
61	874	878	5	955	987	579	1012	1014	998	5	649	142	364	192	2
62	874	879	5	955	987	578	1011	1014	998	5	649	141	367	189	2
63	874	879	5	956	987	577	1011	1014	998	5	652	141	369	190	2
64	874	879	5	956	987	576	1012	1013	998	5	652	142	376	184	2
65	874	879	5	956	987	577	1012	1013	998	5	654	143	369	180	2
66	874	879	5	956	987	574	1012	1014	998	5	665	142	361	184	2
67	874	879	5	956	987	575	1012	1013	998	5	652	141	378	191	2
68	889	889	5	943	976	509	1012	1014	1000	285	642	107	427	179	3
69	889	890	5	944	976	510	1012	1014	1000	278	634	106	433	197	3
70	889	890	5	944	976	506	1012	1014	1000	271	704	107	437	229	3
71	889	890	5	944	976	505	1012	1013	1000	268	717	108	439	233	3
72	889	890	5	944	976	510	1012	1014	1000	275	715	108	439	164	3
73	889	890	5	944	976	511	1012	1014	1000	282	713	111	423	158	3
74	889	890	5	945	976	512	1012	1014	1000	282	701	111	434	221	3
75	889	890	5	945	976	514	1012	1013	1000	272	710	112	440	244	3
76	889	890	5	945	976	512	1012	1014	1001	263	718	113	443	244	3
77	890	890	5	945	976	509	1012	1014	1001	263	714	113	448	247	3
78	890	890	5	945	976	508	1012	1014	1001	271	713	113	450	255	3
79	890	890	5	945	976	509	1012	1014	1001	277	716	113	452	230	3
80	890	890	5	946	976	511	1012	1014	1001	273	708	114	445	224	3
81	890	890	5	946	977	507	1012	1014	1001	272	710	115	454	242	3
82	890	890	5	946	976	503	1012	1014	1001	277	718	114	453	249	3
83	890	890	5	946	976	503	1012	1014	1001	278	721	114	457	257	3

84	890	890	5	946	976	501	1012	1014	1001	244	734	116	472	253	3
85	890	890	5	946	977	508	1012	1014	1001	254	733	116	463	259	3
86	890	890	5	946	977	511	1012	1014	1001	274	737	116	456	237	3
87	890	890	5	946	977	515	1012	1014	1001	287	739	115	438	194	3
88	891	890	5	946	977	518	1012	1014	1001	289	723	115	443	250	3
89	890	890	5	946	977	515	1012	1014	1001	303	726	116	445	281	3
90	891	890	5	946	977	512	1012	1014	1001	293	728	116	442	281	3
91	891	890	5	947	977	509	1012	1014	1001	297	723	116	450	281	3
92	891	890	5	947	977	507	1012	1014	1001	290	740	117	449	289	3
93	891	890	5	946	977	506	1012	1014	1001	293	758	116	444	279	3
94	890	890	5	947	977	504	1012	1014	1001	297	721	116	429	256	3
95	891	891	5	947	977	507	1012	1014	1001	297	734	117	434	284	3
96	891	890	5	947	977	507	1012	1014	1001	293	728	117	439	291	3
97	891	891	5	947	977	512	1012	1014	1001	289	741	117	441	300	3
98	891	891	5	947	977	516	1012	1014	1001	292	742	117	445	302	3
99	891	890	5	947	977	512	1012	1014	1001	292	733	117	442	306	3
100	877	869	5	976	979	645	1011	1010	1005	5	4	107	4	4	4
101	875	856	4	976	973	598	1012	1011	1007	5	4	111	4	5	4
102	879	856	5	976	974	590	1012	1011	1008	5	4	111	4	4	4
103	881	854	4	976	974	583	1012	1011	1008	5	4	112	4	5	4
104	882	853	4	977	974	583	1012	1011	1008	5	4	113	5	5	4
105	878	845	5	976	973	584	1012	1011	1009	5	5	113	4	4	4
106	875	853	5	977	972	586	1012	1011	1009	5	4	114	5	4	4
107	874	859	4	978	973	591	1013	1011	1009	5	4	114	4	5	4
108	874	859	4	978	972	596	1012	1011	1009	5	5	115	4	4	4
109	873	859	4	979	972	606	1012	1012	1009	15	4	115	4	4	4
110	877	859	5	979	973	612	1013	1012	1009	17	4	116	5	4	4
111	876	852	4	979	973	615	1013	1012	1009	15	4	116	5	5	4
112	880	851	4	979	973	615	1013	1012	1010	13	5	117	4	4	4
113	867	849	4	979	974	619	1013	1012	1010	72	4	117	5	4	4
114	869	826	4	979	972	625	1013	1012	1010	119	4	117	4	4	4
115	873	830	5	980	973	626	1013	1011	1010	168	4	117	4	4	4
116	889	837	4	980	974	630	1013	1012	1010	68	4	118	4	5	4
117	888	842	4	981	974	635	1013	1012	1010	16	4	118	4	5	4
118	833	847	4	980	974	638	1013	1012	1010	36	4	118	4	4	4
119	849	849	4	980	974	637	1013	1012	1010	11	4	119	4	4	4
120	853	850	4	979	974	626	1012	1012	1010	16	4	119	4	4	4
121	859	851	4	980	975	631	1012	1012	1010	21	4	119	5	4	4
122	860	852	5	980	975	635	1012	1012	1010	17	4	120	5	4	4
123	861	853	4	980	973	639	1013	1012	1010	30	4	120	5	5	4
124	862	854	4	980	974	641	1012	1012	1010	21	4	120	4	4	4
125	863	854	5	981	974	638	1013	1012	1010	61	4	120	5	4	4
126	863	855	4	981	974	643	1013	1012	1010	79	4	121	5	5	4
127	863	855	5	981	975	644	1013	1012	1010	115	4	121	5	4	4
128	862	853	5	981	975	642	1013	1012	1010	218	5	121	5	5	4
129	863	854	4	981	975	645	1013	1012	1010	140	4	121	5	4	4
130	866	855	4	981	975	645	1013	1012	1010	130	4	121	4	5	4
131	867	855	4	981	975	645	1013	1012	1010	239	4	121	4	4	4
132	871	856	5	981	975	645	1013	1012	1010	251	5	121	4	4	4
133	888	857	5	982	975	651	1013	1012	1010	38	4	121	4	4	4
134	904	857	4	983	975	656	1013	1012	1011	31	4	121	5	5	4
135	864	869	5	1002	999	833	1011	1010	1009	781	6	127	398	151	1
136	867	871	5	1003	999	831	1011	1011	1009	779	5	231	403	131	1
137	868	872	5	1003	1000	830	1011	1011	1009	804	5	229	425	113	1
138	868	873	5	1003	1000	836	1011	1011	1010	808	5	285	419	162	1
139	870	874	6	1003	1000	831	1011	1011	1010	810	6	472	437	148	1
140	871	875	6	1003	1000	834	1011	1011	1010	795	6	472	431	127	1
141	872	876	6	1003	1000	830	1011	1011	1010	834	6	560	400	6	1
142	873	876	6	1004	1000	833	1011	1011	1010	835	6	533	428	135	1
143	874	877	6	1003	1000	843	1011	1011	1010	836	6	577	430	25	1
144	874	877	6	1004	1000	837	1011	1012	1010	843	7	593	439	134	1
145	874	877	6	1003	1000	829	1012	1012	1010	876	6	602	436	109	1
146	875	878	6	1003	1000	835	1011	1011	1010	887	7	625	382	6	1
147	876	878	7	1003	1000	837	1011	1012	1010	890	7	636	396	6	1
148	876	879	6	1003	1001	838	1012	1012	1010	890	7	649	416	7	1
149	876	879	7	1004	1001	841	1012	1012	1010	888	7	658	424	6	1
150	876	879	6	1003	1001	840	1012	1012	1010	888	8	649	401	76	1
151	877	879	6	1004	1001	841	1012	1012	1010	891	7	658	421	47	1
152	877	879	7	1004	1001	843	1012	1012	1010	898	7	674	425	7	1
153	877	880	7	1003	1001	845	1012	1011	1010	902	7	676	390	7	1
154	877	880	6	1004	1001	844	1012	1012	1011	904	8	669	411	6	1
155	878	880	6	1003	1001	845	1012	1012	1011	903	7	686	420	7	1
156	878	880	6	1004	1001	847	1012	1012	1011	899	8	642	436	6	1
157	878	880	7	1004	1001	849	1012	1011	1010	903	8	693	450	7	1
158	878	879	7	1004	1001	848	1011	1012	1010	908	8	667	453	67	1
159	878	880	7	1004	1001	846	1012	1012	1011	913	8	679	454	16	1
160	879	880	7	1004	1001	836	1011	1012	1010	920	8	681	395	7	1
161	879	881	6	1004	1001	830	1012	1012	1011	923	8	682	421	7	1
162	879	881	7	1004	1001	836	1012	1012	1011	923	8	658	419	7	1
163	879	881	6	1004	1001	842	1012	1012	1011	925	8	669	431	68	1
164	879	881	7	1003	1002	845	1011	1012	1011	925	8	673	457	101	1
165	879	881	7	1004	1002	844	1011	1012	1011	927	8	662	460	103	1
166	879	881	6	1003	1001	844	1012	1012	1011	929	8	617	464	62	1
167	880	881	7	1003	1002	841	1011	1012	1011	931	8	620	451	6	1

168	880	882	5	1004	1002	836	1012	1012	1011	940	20	150	534	237	2
169	880	882	5	1004	1002	835	1012	1012	1011	940	23	148	590	295	2
170	880	882	5	1004	1002	832	1012	1012	1011	938	23	146	606	312	2
171	880	882	5	1003	1002	830	1011	1012	1011	938	23	148	603	300	2
172	880	882	5	1004	1002	829	1012	1012	1011	939	25	147	583	286	2
173	881	882	5	1004	1002	830	1012	1012	1011	941	24	160	609	305	2
174	881	882	5	1004	1002	832	1012	1012	1011	941	24	158	608	315	2
175	881	882	5	1004	1002	834	1012	1012	1011	940	25	159	623	340	2
176	881	883	5	1004	1002	835	1012	1012	1011	938	25	160	621	332	2
177	881	883	5	1004	1002	832	1012	1012	1011	938	25	164	608	320	2
178	881	883	5	1004	1002	835	1012	1012	1011	940	24	168	599	319	2
179	881	883	5	1004	1002	834	1012	1012	1011	940	25	167	611	329	2
180	881	883	5	1004	1002	836	1012	1012	1011	939	25	170	622	325	2
181	881	883	5	1004	1002	838	1012	1012	1012	937	25	163	624	354	2
182	881	883	5	1004	1002	839	1012	1013	1011	939	25	166	625	345	2
183	881	883	5	1004	1002	839	1012	1012	1012	941	25	166	616	322	2
184	882	883	5	1003	1002	838	1012	1012	1011	944	25	168	576	308	2
185	882	884	5	1004	1002	835	1012	1012	1012	945	25	167	615	325	2
186	882	883	5	1004	1002	832	1012	1012	1011	946	24	169	621	307	2
187	882	883	5	1004	1002	834	1012	1012	1011	946	24	171	629	326	2
188	882	883	5	1004	1002	841	1012	1012	1012	943	25	171	629	354	2
189	882	883	5	1004	1002	842	1012	1012	1012	944	25	170	633	350	2
190	882	883	5	1004	1002	843	1012	1012	1012	945	25	169	636	349	2
191	882	883	5	1004	1002	847	1012	1012	1011	946	25	168	630	311	2
192	883	883	5	1004	1002	851	1012	1012	1012	946	25	176	584	275	2
193	883	883	6	1004	1002	850	1012	1013	1012	948	25	176	611	289	2
194	883	884	5	1004	1002	850	1012	1012	1011	948	25	176	619	300	2
195	883	884	5	1004	1003	850	1012	1012	1012	948	25	176	623	302	2
196	883	884	5	1004	1002	850	1012	1012	1011	948	26	175	634	327	2
197	883	884	5	1004	1002	850	1012	1012	1011	946	26	175	641	335	2
198	883	884	5	1004	1002	850	1012	1012	1011	946	26	175	640	328	2
199	883	884	5	1004	1002	844	1012	1012	1012	945	26	175	632	314	2
200	883	884	5	1004	1002	840	1012	1012	1012	949	25	174	624	329	2
201	883	884	5	1004	1002	844	1012	1012	1011	950	25	174	633	315	2
202	883	884	5	1004	1003	844	1012	1013	1012	948	25	178	635	348	2
203	883	884	5	1004	1003	845	1012	1012	1011	950	23	179	633	333	2
204	883	884	5	1004	1003	845	1012	1012	1012	949	24	180	598	289	2
205	883	885	5	1004	1003	844	1012	1012	1012	950	24	179	603	278	2
206	883	884	5	1004	1002	846	1012	1013	1012	951	24	179	625	320	2
207	883	884	5	1004	1003	846	1012	1012	1012	951	24	180	633	333	2
208	883	884	5	1001	1000	833	1012	1012	1012	993	15	115	429	163	3
209	883	884	5	1001	1000	836	1012	1012	1012	994	12	115	385	160	3
210	883	884	5	1001	1000	840	1012	1012	1012	994	13	121	401	186	3
211	883	883	5	1001	1000	839	1012	1012	1012	995	12	119	427	197	3
212	883	883	5	1002	1000	833	1012	1012	1012	995	13	119	403	190	3
213	883	883	5	1002	1001	829	1012	1012	1012	996	14	121	440	182	3
214	883	883	5	1001	1000	836	1012	1012	1012	995	8	125	416	137	3
215	883	883	5	1002	1000	849	1012	1012	1013	996	7	125	355	78	3
216	883	884	5	1001	1000	851	1012	1012	1013	996	8	129	423	161	3
217	883	884	5	1002	1001	851	1012	1012	1012	996	11	129	434	176	3
218	883	884	5	1002	1001	851	1012	1012	1012	996	11	129	444	182	3
219	883	884	5	1002	1000	851	1012	1012	1012	996	11	130	445	179	3
220	883	884	5	1002	1000	851	1012	1012	1012	996	13	132	446	178	3
221	883	884	5	1002	1000	850	1012	1012	1012	996	8	135	437	144	3
222	883	884	5	1002	1001	844	1012	1012	1012	996	8	136	446	157	3
223	883	884	5	1002	1000	844	1012	1013	1012	996	8	136	449	181	3
224	884	884	5	1002	1001	840	1012	1012	1013	997	11	138	453	184	3
225	883	884	5	1002	1000	844	1012	1012	1012	997	13	136	449	196	3
226	883	884	5	1002	1001	847	1012	1012	1012	997	8	136	437	205	3
227	883	884	5	1002	1001	848	1012	1012	1013	997	8	132	395	139	3
228	884	884	5	1002	1001	845	1012	1013	1013	997	7	133	399	88	3
229	884	884	5	1003	1001	838	1012	1013	1012	997	7	135	394	95	3
230	884	884	5	1002	1001	839	1012	1013	1013	998	7	137	400	100	3
231	884	885	5	1002	1001	832	1012	1012	1013	998	8	138	427	127	3
232	884	884	5	1002	1001	828	1012	1012	1013	998	8	139	422	129	3
233	884	884	5	1003	1001	848	1012	1013	1013	998	8	137	391	129	3
234	883	884	5	1002	1001	849	1012	1012	1013	998	7	136	401	87	3
235	884	884	5	1003	1001	849	1012	1013	1013	999	5	136	347	7	3
236	884	884	5	1003	1001	848	1012	1013	1013	999	5	137	362	7	3
237	884	884	5	1002	1001	850	1013	1013	1013	999	5	137	362	6	3
238	884	884	5	1003	1001	851	1012	1013	1013	999	5	137	350	6	3
239	884	884	5	1003	1001	855	1012	1013	1013	999	5	137	357	6	3
240	884	884	5	1003	1001	855	1012	1013	1013	999	5	136	357	5	3
241	884	884	5	1003	1001	857	1013	1013	1013	999	5	136	287	5	3
242	884	884	5	1003	1001	858	1012	1012	1013	999	5	138	304	5	3
243	884	884	5	1003	1001	857	1012	1013	1013	999	5	138	374	5	3
244	884	884	5	1003	1001	857	1012	1013	1013	999	7	138	441	75	3
245	884	884	5	1003	1001	857	1013	1013	1013	999	5	141	420	78	3
246	884	884	5	1003	1001	858	1013	1013	1013	999	5	141	404	70	3
247	883	884	5	1003	1001	859	1012	1013	1013	1000	5	138	336	5	3
248	884	884	5	1003	1001	858	1012	1013	1013	1000	5	137	334	5	3
249	883	884	5	1003	1001	859	1013	1013	1013	1000	5	138	357	5	3
250	876	876	5	1006	1004	889	1013	1014	1012	939	5	115	5	4	4
251	879	877	5	1006	1005	895	1014	1014	1011	884	5	117	5	5	4

252	879	878	5	1007	1005	904	1014	1014	1012	912	5	122	5	5	4
253	879	878	5	1007	1005	903	1014	1014	1012	921	5	125	5	5	4
254	880	879	5	1007	1005	905	1014	1013	1012	929	5	126	5	5	4
255	880	879	5	1007	1005	906	1014	1014	1012	928	5	127	5	5	4
256	880	880	5	1007	1005	905	1014	1014	1012	932	5	127	5	5	4
257	881	880	5	1008	1005	906	1014	1014	1012	933	5	127	5	5	4
258	881	880	5	1008	1006	906	1014	1014	1012	936	5	128	5	5	4
259	881	881	5	1008	1006	907	1014	1014	1012	936	5	129	5	5	4
260	882	881	5	1008	1005	908	1014	1014	1012	932	5	128	5	5	4
261	882	881	5	1008	1006	908	1014	1014	1012	931	5	129	5	5	4
262	882	881	5	1008	1006	908	1014	1014	1012	935	5	129	5	5	4
263	881	882	5	1008	1006	909	1014	1014	1012	940	5	129	5	5	4
264	882	882	5	1008	1006	908	1014	1014	1012	941	5	130	5	5	4
265	882	882	5	1008	1006	908	1014	1014	1012	942	5	130	5	5	4
266	882	882	5	1008	1006	909	1014	1014	1012	942	5	129	5	5	4
267	883	882	5	1008	1005	909	1014	1014	1012	944	5	129	4	5	4
268	882	882	5	1008	1006	909	1014	1014	1012	943	5	129	5	5	4
269	883	882	5	1008	1006	910	1014	1014	1012	946	5	129	5	5	4
270	883	882	5	1008	1006	909	1014	1014	1012	948	5	131	5	5	4
271	883	882	5	1008	1006	909	1014	1014	1013	949	5	131	5	5	4
272	883	883	5	1008	1006	910	1014	1014	1013	947	5	132	5	5	4
273	883	882	5	1008	1006	910	1014	1014	1012	948	5	132	5	5	4
274	884	883	5	1008	1006	910	1014	1014	1013	948	5	134	5	5	4
275	883	883	5	1008	1006	910	1014	1014	1013	949	5	135	5	5	4
276	884	883	5	1008	1006	910	1014	1014	1013	950	5	135	5	5	4
277	884	883	5	1008	1006	910	1014	1014	1013	953	5	136	5	5	4
278	884	883	5	1008	1006	910	1014	1014	1013	953	5	136	5	5	4
279	884	883	5	1008	1006	910	1014	1014	1013	954	5	137	5	5	4
280	884	883	5	1008	1006	910	1014	1014	1013	954	5	137	5	5	4
281	884	883	5	1008	1006	911	1014	1014	1013	953	5	137	5	5	4
282	884	883	5	1008	1006	911	1014	1014	1013	955	5	137	5	5	4
283	884	884	5	1008	1006	911	1014	1014	1013	955	5	136	5	5	4
284	884	884	5	1008	1006	911	1014	1014	1013	956	5	136	5	5	4
285	885	884	5	1008	1006	911	1014	1014	1013	948	5	136	5	5	4
286	885	884	5	1008	1006	912	1014	1014	1013	952	5	137	5	5	4
287	885	884	5	1008	1006	912	1014	1014	1013	951	5	137	5	5	4

Appendix J - Participant Consent Form

NTNU
Fakultet for ingeniørvitenskap og teknologi
Institutt for Maskinteknikk og Produksjon

Vår dato	Vår referanse
11.05.17	MS
Deres dato	Deres referanse

Request for Participation in Research Project

Warning
If you are epileptic, please make us aware of this.

Background and Purpose
The purpose of this project is to study the change of physiological reactions due to changes in affect. This experiment is part of a MSc at MTP, Norwegian University of Science and Technology.

What does participation in the project imply?
The participant will be asked to conduct a select number of rounds in a computer game, and data from these will be stored. After being introduced to the task, the participant will be guided through the experiment. The experiment is comprised of three parts; Part I, Part II and Part III. The participant will be asked to fill out several surveys as part of the experiment. A video recording of the participant will be made. The participant will be presented with true and erroneous information.

What will happen to the information about you?
All personal data will be treated anonymously. No name is connected to the gathered data. The only persons having access to the data are the two master students and their supervisor. In case of a publication, participants will therefore not be recognizable by name. The project is scheduled for completion by 01.07.2017. After this date the personal data will be stored encrypted.

Voluntary participation
The participation of this experiment is voluntary, and you can at any time choose to stop and withdraw from the experiment. If you would like to participate or if you have any questions concerning the project, please contact Erik Borge (+47 95222034) or Helge Garsmark (+47 94898375).

Consent for participation in the study
I have received information about the project and am willing to participate. I agree that data is collected, analyzed and published anonymously. I further agree to be confidential about the experiment to provide non-biased conditions for every participant.

Name of the participant (*Please use capital letters*) e-mail *(for award purposes)*

Signature (*Please include Place & Date*)

Postadresse
7491 Trondheim

Org.nr. 974 767 880
E-post:
martin.steinert@ntnu.no
http://www.ivt.ntnu.no/ipm/

Besøksadresse
Richard Birkelandsvei 2b
Lab M66
Gløshaugen

Telefon
+ 47 91 89 78 30
Telefaks
+ 47 73 59 41 29

Appendix K - Background Information Questionnaire

NTNU
Fakultet for ingeniørvitenskap og teknologi
Institutt for produktutvikling og materialer

Vår dato	Vår referanse
11.05.17	HSG
Deres dato	Deres referanse

Background Information Questionnaire

This questionnaire is designed to collect additional background information about you.

Part A. Education

The questions in this section are designed to collect information on your education.

A1. What is your current level of achieved education?

- ☐ High School
- ☐ Bachelor's Degree
- ☐ Master's Degree
- ☐ Ph.D.

A2. When did you graduate?

Month ☐☐ Year ☐☐☐☐

A3. Please record your <u>primary</u> area of specialization.

Primary Area of Specialization: ________________________________

A4. Are you currently studying for a degree? If no, skip to part B. If yes, please specify:

- ☐ High School
- ☐ Bachelor's Degree
- ☐ Master's Degree
- ☐ Ph.D.

A5. When do you plan to graduate?

Month ☐☐ Year ☐☐☐☐

A6. Please record your <u>primary</u> area of specialization.

Primary Area of Specialization: ________________________________

Part B: Demographic Information

Postadresse
7491 Trondheim

Org.nr. 974 767 880
E-post:
helgesg@stud.ntnu.no
http://www.ivt.ntnu.no/ipm/

Besøksadresse
Richard Birkelandsvei 2b
Gløshaugen

Telefon
+ 47 41 64 68 04
Telefaks
+ 47 73 59 41 29

The questions in this section are designed to collect some of your demographic information.

B1. Are you:

- ☐ Male
- ☐ Female

B2. In what year were you born?

Year of Birth: ☐☐☐☐

B3. What is your nationality (i.e. citizenship)?
Please specify if you have multiple citizenships.

Answer: ____

Part B: Specifics

The questions in this section are designed to collect some of your demographic information.

C1. How many cups of coffee did you drink today?

- ☐ None
- ☐ 1
- ☐ 2 or 3
- ☐ 4 or more

C2. Are you feeling sick today?

- ☐ Yes
- ☐ No

C3a. Do you have any chronic illness? If no skip question C3b.

- ☐ Yes
- ☐ No

C3b. Is the illness affecting your emotion?

- ☐ Yes
- ☐ No

Helge Soltvedt Garsmark, Master student
Department of Mechanical and Industrial Engineering(MTP), Norwegian University of Science and Technology (NTNU)
Richard Birkelandsvei 2B, NO-7491 Trondheim, Norway.
Phone: +47 94 89 83 75, helgesg@stud.ntnu.no

Part D: Further Participation

D1. Are you willing to receive follow-up questions or surveys of this study via e-mail in the future? If yes, please write your e-mail address below.

E-mail address: ____

Part E: General Information

You have just participated in an experiment on emotional state evaluation, containing elements of stimuli and physiological measurements.

As stimuli, you were presented with three tasks with different difficulty, in addition you were stimulated with different visuals and audio during the three tasks. The audio stimuli was both performance based and in general connected to the different tasks. You provided us with three different types of emotional feedback during this experiment. Also physiological data was collected from you through EMG, ECG, a distance sensor and the chair. The goal of this experiment is to provide qualitative data on the correlation between emotional state and physiological measurements, and further how a possible correlation can be used in affective research.

We wish to remind you to be confidential about the content of this experiment to provide non-biased conditions for every participant, as stated in the consent form. We hope you enjoyed participating, and thank you kindly for your commitment of time to this experiment!

Thank you for your time and participation!

Helge Soltvedt Garsmark, Master student
Department of Mechanical and Industrial Engineering(MTP), Norwegian University of Science and Technology (NTNU)
Richard Birkelandsvei 2B, NO-7491 Trondheim, Norway.
Phone: +47 94 89 83 75, helgesg@stud.ntnu.no

Appendix L - Correlation matrices

Table 7.1: Correlation Matrix, HRV, AD ACL, and Affect Grid

		Heart Rate Mean	pNN50	LF power	HF power	LF/HF	A1	A2	B1	B2	AD-ACL Total	Grid y-score
Heart Rate Mean	Correlation Coefficient	1.000	-.819	-.602	-.635	.223	.133	-.198	.149	-.197	.194	.088
	Sig. (2-tailed)	.	.000	.000	.000	.074	.292	.114	.237	.116	.121	.483
	N	65	65	65	65	65	65	65	65	65	65	65
pNN50	Correlation Coefficient	-.819	1.000	.697	.878	-.420	-.219	.239	-.214	.279	-.248	-.151
	Sig. (2-tailed)	.000	.	.000	.000	.000	.080	.055	.087	.025	.046	.230
	N	65	65	65	65	65	65	65	65	65	65	65
LF power	Correlation Coefficient	-.602	.697	1.000	.673	.108	-.152	.025	-.107	.069	-.090	-.004
	Sig. (2-tailed)	.000	.000	.	.000	.393	.227	.846	.394	.584	.478	.974
	N	65	65	65	65	65	65	65	65	65	65	65
HF power	Correlation Coefficient	-.635	.878	.673	1.000	-.601	-.224	.165	-.144	.249	-.180	-.142
	Sig. (2-tailed)	.000	.000	.000	.	.000	.072	.188	.252	.045	.152	.260
	N	65	65	65	65	65	65	65	65	65	65	65
LF/HF	Correlation Coefficient	.223	-.420	.108	-.601	1.000	.128	-.165	.061	-.195	.111	.123
	Sig. (2-tailed)	.074	.000	.393	.000	.	.310	.189	.632	.119	.378	.330
	N	65	65	65	65	65	65	65	65	65	65	65
A1	Correlation Coefficient	.133	-.219	-.152	-.224	.128	1.000	-.717	.641	-.659	.811	.697
	Sig. (2-tailed)	.292	.080	.227	.072	.310	.	.000	.000	.000	.000	.000
	N	65	65	65	65	65	67	67	67	67	67	67
A2	Correlation Coefficient	-.198	.239	.025	.165	-.165	-.717	1.000	-.659	.737	-.879	-.725
	Sig. (2-tailed)	.114	.055	.846	.188	.189	.000	.	.000	.000	.000	.000
	N	65	65	65	65	65	67	67	67	67	67	67
B1	Correlation Coefficient	.149	-.214	-.107	-.144	.061	.641	-.659	1.000	-.682	.840	.691
	Sig. (2-tailed)	.237	.087	.394	.252	.632	.000	.000	.	.000	.000	.000
	N	65	65	65	65	65	67	67	67	67	67	67
B2	Correlation Coefficient	-.197	.279	.069	.249	-.195	-.659	.737	-.682	1.000	-.873	-.794
	Sig. (2-tailed)	.116	.025	.584	.045	.119	.000	.000	.000	.	.000	.000
	N	65	65	65	65	65	67	67	67	67	67	67
AD-ACL total	Correlation Coefficient	.194	-.248	-.090	-.180	.111	.811	-.879	.840	-.873	1.000	.827
	Sig. (2-tailed)	.121	.046	.478	.152	.378	.000	.000	.000	.000	.	.000
	N	65	65	65	65	65	67	67	67	67	67	67
Grid y-score	Correlation Coefficient	.088	-.151	-.004	-.142	.123	.697	-.725	.691	-.794	.827	1.000
	Sig. (2-tailed)	.483	.230	.974	.260	.330	.000	.000	.000	.000	.000	.
	N	65	65	65	65	65	67	67	67	67	67	67

Table 7.2: Complete correlation between the posture measures and the HRV features

		Number of Position Changes	Leaning Chair	Leaning Lidar	Heart Rate Mean	pNN50	LF power	HF power	LF/HF
Number of Position Changes	Pearson Correlation	1	-.107	.003	.318	-.235	-.027	-.124	.150
	Sig. (2-tailed)		.387	.983	.010	.059	.829	.324	.234
	N	67	67	67	65	65	65	65	65
Leaning Chair	Pearson Correlation	-.107	1	-.407	.088	-.261	-.283	-.164	.120
	Sig. (2-tailed)	.387		.001	.488	.036	.022	.191	.341
	N	67	67	67	65	65	65	65	65
Leaning Lidar	Pearson Correlation	.003	-.407	1	-.208	.336	.257	.094	.082
	Sig. (2-tailed)	.983	.001		.096	.006	.039	.457	.516
	N	67	67	67	65	65	65	65	65
Heart Rate Mean	Pearson Correlation	.318	.088	-.208	1	-.776	-.578	-.368	.158
	Sig. (2-tailed)	.010	.488	.096		.000	.000	.003	.208
	N	65	65	65	65	65	65	65	65
pNN50	Pearson Correlation	-.235	-.261	.336	-.776	1	.628	.671	-.367
	Sig. (2-tailed)	.059	.036	.006	.000		.000	.000	.003
	N	65	65	65	65	65	65	65	65
LF power	Pearson Correlation	-.027	-.283	.257	-.578	.628	1	.503	.178
	Sig. (2-tailed)	.829	.022	.039	.000	.000		.000	.157
	N	65	65	65	65	65	65	65	65
HF power	Pearson Correlation	-.124	-.164	.094	-.368	.671	.503	1	-.426
	Sig. (2-tailed)	.324	.191	.457	.003	.000	.000		.000
	N	65	65	65	65	65	65	65	65
LF/HF	Pearson Correlation	.150	.120	.082	.158	-.367	.178	-.426	1
	Sig. (2-tailed)	.234	.341	.516	.208	.003	.157	.000	
	N	65	65	65	65	65	65	65	65

Table 7.3: Complete correlation between the posture measures and the subjective measures

		Number of Position Changes	Leaning Chair	Leaning Lidar	A1	A2	B1	B2	AD-ACL Total	Grid y-score
Number of Position Changes	Correlation Coefficient	1.000	-.112	.032	.363	-.461	.276	-.464	.465	.433
	Sig. (2-tailed)	.	.365	.795	.003	.000	.024	.000	.000	.000
	N	67	67	67	67	67	67	67	67	67
Leaning Chair	Correlation Coefficient	-.112	1.000	-.391	.040	.153	.054	.074	-.057	.005
	Sig. (2-tailed)	.365	.	.001	.745	.216	.667	.552	.644	.966
	N	67	67	67	67	67	67	67	67	67
Leaning Lidar	Correlation Coefficient	.032	-.391	1.000	.118	-.190	.043	.021	.071	-.026
	Sig. (2-tailed)	.795	.001	.	.340	.124	.728	.866	.570	.833
	N	67	67	67	67	67	67	67	67	67
A1	Correlation Coefficient	.363	.040	.118	1.000	-.717	.641	-.659	.811	.697
	Sig. (2-tailed)	.003	.745	.340	.	.000	.000	.000	.000	.000
	N	67	67	67	67	67	67	67	67	67
A2	Correlation Coefficient	-.461	.153	-.190	-.717	1.000	-.659	.737	-.879	-.725
	Sig. (2-tailed)	.000	.216	.124	.000	.	.000	.000	.000	.000
	N	67	67	67	67	67	67	67	67	67
B1	Correlation Coefficient	.276	.054	.043	.641	-.659	1.000	-.682	.840	.691
	Sig. (2-tailed)	.024	.667	.728	.000	.000	.	.000	.000	.000
	N	67	67	67	67	67	67	67	67	67
B2	Correlation Coefficient	-.464	.074	.021	-.659	.737	-.682	1.000	-.873	-.794
	Sig. (2-tailed)	.000	.552	.866	.000	.000	.000	.	.000	.000
	N	67	67	67	67	67	67	67	67	67
AD-ACL total	Correlation Coefficient	.465	-.057	.071	.811	-.879	.840	-.873	1.000	.827
	Sig. (2-tailed)	.000	.644	.570	.000	.000	.000	.000	.	.000
	N	67	67	67	67	67	67	67	67	67
Grid y-score	Correlation Coefficient	.433	.005	-.026	.697	-.725	.691	-.794	.827	1.000
	Sig. (2-tailed)	.000	.966	.833	.000	.000	.000	.000	.000	.
	N	67	67	67	67	67	67	67	67	67

Appendix M - Risk Assessment

NTNU	Kartlegging av risikofylt aktivitet	Utarbeidet av	Nummer	Dato
HMS		HMS-avd.	HMSRV2601	22.03.2011
		Godkjent av		Erstatter
		Rektor		01.12.2006

Enhet: **Institutt for Maskinteknikk og Produksjon** **Dato:** **05.05.2017**

Linjeleder: **Torgeir Welo**

Deltakere ved kartleggingen (m/ funksjon)**:** Martin Steinert, veileder/Jørgen A. B. Erichsen, Coach/Andreas Wulvik, Coach/ Helge S. Garsmark, student

Kort beskrivelse av hovedaktivitet/hovedprosess: Masteroppgave Helge S. Garsmark.
Experimentally piloting and testing wether we can use body language to measure arousal.

Er oppgaven rent teoretisk? (JA/NEI)**:** NEI

Signaturer: *Ansvarlig veileder:* *Martin Steinert* *Studenter:* *Helge Soltvedt Garsmark*

ID nr.	Aktivitet/prosess	Ansvarlig	Eksisterende dokumentasjon	Eksisterende sikringstiltak	Lov, forskrift o.l.	Kommentar
1	Bruk av Trolllabs workshop.	HG	Romkort	Romkort		
1a	Bruk av roterende maskineri	HG	Maskinens brukermanual	Ukjent	Ukjent	
1b	Bruk av laserkutter	HG	Maskinens brukermanual	Ukjent	Ukjent	
1c	Bruk av 3D printer	HG	Maskinens brukermanual	Ukjent	Ukjent	
1d	Bruk av skjæreverktøy	HG	Ukjent			

1e	Bruk av samenføynigsmidler (lim og lignende.)	HG	Produktets brukermanual og datablad	Datablad	Ukjent	
2	Tilstedeværelse ved arbeid utført av andre.	Andre	Andres HMSRV2601	Andres HMSRV2601	Prosessavhengig	
3	Eksperimentelt arbeid	HG	Risikovurdering		Prosessavhengig	

Enhet: **Dato:**

Linjeleder:

Deltakere ved kartleggingen (m/ funksjon): Martin Steinert, veileder/Jørgen A. B. Erichsen, Coach/Andreas Wulvik, Coach/ Helge S. Garsmark, student

Risikovurderingen gjelder hovedaktivitet: Masteroppgave Helge S. Garsmark. Experimentally piloting and testing wether we can use body language to measure arousal.

Signaturer: *Ansvarlig veileder:* *Student:*

ID nr	Aktivitet fra kartleggings-skjemaet	Mulig uønsket hendelse/ belastning	Vurdering av sannsyn-lighet (1-5)	Vurdering av konsekvens: Menneske (A-E)	Ytre miljø (A-E)	Øk/ materiell (A-E)	Om-dømme (A-E)	Risiko-Verdi (menn-eske)	Kommentarer/status Forslag til tiltak
1	Bruk av Trolllabs workshop.								Vær nøye med opplæring i bruk av maskineri.
1a-i	Bruk av roterende maskineri	Stor kuttskade	2	D	A	A	D	2D	Sørg for at roterende deler er tilstrekkelig sikret/dekket.
1a-ii		Liten kuttskade	3	B	A	A	A	3B	Ikke ha løse klær/tilbehør på kroppen.
1a-iii		Klemskade	2	D	A	A	C	2D	Ikke ha løse klær/tilbehør på kroppen.
1a-iv		Flygende spon/gjenstander	3	C	A	A	B	3C	Bruk øyevern og tildekk hurtig roterende deler (Fres og lignende.)
1a-v		Feil bruk - ødelagt utstyr	3	A	A	C	A	3C	Opplæring.

1b-i	Bruk av laserkutter	Klemskade	2	D	A	A	C	2D	Ikke ha løse klær/tilbehør på kroppen.
1b-ii		Brannskade	3	B	A	A	A	3B	Bruk hansker ved håndtering av varme materialer.
1b-iii		Øyeskade-laser	2	D	A	A	C	2D	Bruk øyevern! Skru av laser når maskinen ved oppsett.
1b-iv		Brann	2	B	A	D	C	2B	Ha slukkeutstur tilgjengelig.
1c-i	Bruk av 3D-printer	Brannskade	3	B	A	A	A	3B	Vær oppmerksom.
1c-ii		Innhalering av plast/ printemateriale	5	A	A	A	A	5A	Bruk åndedrettsvern/ vernHGriller.
1c-iii		Feil bruk - ødelagt maskineri	3	A	A	C	A	3A	Opplæring.
1d-i	Bruk av skjæreverktøy	Stor kuttskade	2	D	A	A	D	2D	Bruk skarpe verktøy og riktig skjæreunderlag.
1d-ii		Liten kuttskade	3	B	A	A	A	3B	Bruk skarpe verktøy og riktig skjæreunderlag.
1e-i	Bruk av samenføynigsmidler (lim og lignende.)	Eksponering på øyet	2	D	A	A	B	2D	Bruk øyevern, ha datablad tilgjengelig.

1e-ii		Eksponering hud	4	A	A	A	A	4A	Bruk hansker, ha datablad tilgjengelig.
1e-iii		Eksponering åndedrett	4	A	A	A	A	4A	Bruk åndedretsvært/ god ventilasjon. Ha datablad tilgjengelig.
1e-iv		Søl	4	A	B	A	A	4A	Ha papir/ rengjøringsmateriell tilgjengelig. Ha datablad tilgjengelig.
2	Tilstedeværelse ved arbeid utført av andre.	Se andres risikovurdering om sikkerhet betviles.	3	C	C	C	C	3C	Hold et øye med hva som foregår rundt deg.
3-i	Eksperimentelt arbeid	Skade ved fall e.l.	2	A	A	A	A	2A	Sikre eksperimentelt utstyr. Førstehjelps-kit tilgjengelig.
3-ii		Anfall grunnet mye impuls	2	B	A	A	A	2B	I forkant sikre at testsubjekter er rustet til det som skal gjøres i eksperimentet. Førstehjelps-kit tilgjengelig.
3-iii		Skade ved bruk av sensorikk i nærheten av mennesker	2	B	A	A	A	2B	Sørge for at sensorikk brukes på forsvalig vis og lese datablad på sensorene.

Sannsynlighet vurderes etter følgende kriterier:

Svært liten 1	Liten 2	Middels 3	Stor 4	Svært stor 5
1 gang pr 50 år eller sjeldnere	1 gang pr 10 år eller sjeldnere	1 gang pr år eller sjeldnere	1 gang pr måned eller sjeldnere	Skjer ukentlig

Konsekvens vurderes etter følgende kriterier:

Gradering	Menneske	Ytre miljø Vann, jord og luft	Øk/materiell	Omdømme
E Svært Alvorlig	Død	Svært langvarig og ikke reversibel skade	Drifts- eller aktivitetsstans >1 år.	Troverdighet og respekt betydelig og varig svekket
D Alvorlig	Alvorlig personskade. Mulig uførhet.	Langvarig skade. Lang restitusjonstid	Driftsstans > ½ år Aktivitetsstans i opp til 1 år	Troverdighet og respekt betydelig svekket
C Moderat	Alvorlig personskade.	Mindre skade og lang restitusjonstid	Drifts- eller aktivitetsstans < 1 mnd	Troverdighet og respekt svekket
B Liten	Skade som krever medisinsk behandling	Mindre skade og kort restitusjonstid	Drifts- eller aktivitetsstans < 1uke	Negativ påvirkning på troverdighet og respekt
A Svært liten	Skade som krever førstehjelp	Ubetydelig skade og kort restitusjonstid	Drifts- eller aktivitetsstans < 1dag	Liten påvirkning på troverdighet og respekt

Risikoverdi = Sannsynlighet x Konsekvens
Beregn risikoverdi for Menneske. Enheten vurderer selv om de i tillegg vil beregne risikoverdi for Ytre miljø, Økonomi/materiell og Omdømme. I så fall beregnes disse hver for seg.

Til kolonnen "Kommentarer/status, forslag til forHGyggende og korrigerende tiltak":
Tiltak kan påvirke både sannsynlighet og konsekvens. Prioriter tiltak som kan forhindre at hendelsen inntreffer, dvs. sannsynlighetsreduserende tiltak foran skjerpet beredskap, dvs. konsekvensreduserende tiltak.

MATRISE FOR RISIKOVURDERINGER ved NTNU

KONSEKVENS	Svært alvorlig	E1	E2	E3	E4	E5
	Alvorlig	D1	D2	D3	D4	D5
	Moderat	C1	C2	C3	C4	C5
	Liten	B1	B2	B3	B4	B5
	Svært liten	A1	A2	A3	A4	A5
		Svært liten	Liten	Middels	Stor	Svært stor
		SANNSYNLIGHET				

Prinsipp over akseptkriterium. Forklaring av fargene som er brukt i risikomatrisen.

Farge	Beskrivelse
Rød	Uakseptabel risiko. Tiltak skal gjennomføres for å redusere risikoen.
Gul	Vurderingsområde. Tiltak skal vurderes.
Grønn	Akseptabel risiko. Tiltak kan vurderes ut fra andre hensyn.