

MASTER THESIS  
ARTIFICIAL INTELLIGENCE  
v1.0



RADBOUD UNIVERSITY NIJMEGEN

---

# Computerized quantification of facial weakness in facioscapulohumeral muscular dystrophy

---

Sil van de Leemput (s4085469)  
Department of Artificial Intelligence. Radboud University Nijmegen  
silvandeleemput@gmail.com

**Supervisors:**

Prof. dr. B.G.M. van Engelen  
Department of Neurology. Radboud University Medical Center  
baziel.vanengelen@radboudumc.nl

K. Mul, MSc  
Department of Neurology. Radboud University Medical Center  
karlien.mul@radboudumc.nl

dr. F. Grootjen  
Department of Artificial Intelligence. Radboud University Nijmegen  
f.grootjen@donders.ru.nl

**External examiner:**

dr. L.G. Vuurpijl  
Department of Artificial Intelligence. Radboud University Nijmegen  
l.vuurpijl@ai.ru.nl

July 30, 2015

## Abstract

Facioscapulohumeral muscular dystrophy (FSHD) is a rare hereditary and progressive muscular disease. One of the first and most characteristic symptoms of FSHD is asymmetrical weakness of the facial muscles. This weakness varies from minimal asymmetry to a complete lack of facial expression. Due to this weakness, patients are limited in the use of their facial muscles and are thus less able to express themselves in a social context, which can hinder social communication. However, at this moment studies on (the progression of) facial weakness and the consequences on communication are lacking and there is no validated outcome measure for facial weakness. Facial weakness is difficult to objectify and even more difficult to follow up over time. To facilitate future research, a standardized quantitative outcome measure for facial weakness in FSHD is required.

Within this project a grading system for objectively measuring facial weakness and a diagnosis system for predicting FSHD from facial weakness were developed. A novel dataset was created consisting of facial video recordings of FSHD patients and healthy controls while performing various tasks. Video frames at rest and maximal expression were identified and manually labeled with 68 facial landmarks. Experts on facial weakness graded the video recordings of the participants on degree of facial weakness and assessed if FSHD was present. After extracting various types of facial features which were reported to quantify facial weakness, several machine learning systems were trained and evaluated using a newly developed system evaluation pipeline. Subsequently, the best systems were compared with human experts on agreement.

The results show that the developed facial weakness grading systems perform in high agreement with the established ground truth, but that the agreement among human experts should be improved. Furthermore, the developed systems predicting whether a participant has FSHD perform above expert-level. It was found that combining multiple feature types gave the best results and that combining 2D and 3D features yielded better results than only 2D or only 3D features. It was also found that subtraction features were the most unreliable, although this is thought to be related to insufficient head stabilization. Furthermore, the system evaluation pipeline provides a useful framework to further investigate the contribution of features to grade facial weakness and diagnose FSHD.

The work in this thesis shows that it is possible to create an objective facial weakness grading system for FSHD patients with comparable performance to the current golden standard. Many research projects on the effect of various treatments on facial weakness within FSHD could benefit from an objective measure for reporting facial weakness. However, the current work should be improved in many ways before it can serve as such a measure, for example the number of participants should be increased, and a method for automatic landmark localization should be incorporated. The presented work provides a promising starting point, which could eventually lead to the development of a computerized standard for grading facial weakness within FSHD patients.



# Acknowledgments

I would like to thank my supervisors Karlien Mul, Franc Grootjen and Basiel van Engelen for their enthusiasm, encouragement and invaluable feedback. I am also grateful to the department for Artificial Intelligence at the Radboud University Nijmegen for supplying the computer equipment and the Kinect 2 sensor for recording the participants. I am thankful to Louis Vuurpijl for reviewing this thesis and for his support and cooperation with the recording equipment. I also thank Canon for providing the PowerShot cameras for this research. I am grateful to the people at the research technical support group at the faculty of social sciences at the Radboud University Nijmegen for their feedback and help on the statistical tests within this thesis. Furthermore, I would like to thank Mark Blokpoel for his feedback on the illumination for the recording setup. Finally, I would like to express my gratitude towards prof. George Padberg, dr. Carien Beurskens and Simone Knuijt for providing the ground truth by evaluating the movies acquired within this project.

# Contents

<b>Contents</b>	<b>2</b>
<b>Abbreviations</b>	<b>5</b>
<b>1 Introduction</b>	<b>6</b>
1.1 Measuring facial weakness . . . . .	7
1.2 Traditional approaches . . . . .	9
1.3 Computer-based approaches . . . . .	12
1.3.1 Facial landmark systems . . . . .	12
1.3.2 Subtraction methods . . . . .	15
1.4 Other approaches . . . . .	17
1.5 Overview of facial grading systems . . . . .	17
1.6 Project aim and motivation . . . . .	18
1.7 Research goals . . . . .	19
1.8 Within this thesis . . . . .	20
<b>2 Methods</b>	<b>21</b>
2.1 Data acquisition . . . . .	21
2.1.1 Participant demography . . . . .	21
2.1.2 Data and task description . . . . .	21
2.1.3 Acquisition setup . . . . .	29
2.1.4 Acquisition procedure . . . . .	31
2.1.5 Data processing . . . . .	32
2.2 Landmark detection pilot . . . . .	33
2.3 Manual labeling . . . . .	34
2.3.1 Labeling procedure . . . . .	35
2.4 Image processing . . . . .	36
2.4.1 Kinect 2 depth information . . . . .	36
2.4.2 Image stabilization . . . . .	39
2.5 Feature extraction . . . . .	41
2.5.1 Euclidean distances . . . . .	41
2.5.2 Triangle areas . . . . .	41
2.5.3 Motion . . . . .	42
2.5.4 Sjögreen features . . . . .	44
2.5.5 Other features . . . . .	45
2.6 Ground truth . . . . .	46
2.7 Combining features . . . . .	47
2.7.1 Feature labeling . . . . .	48
2.8 System evaluation pipeline . . . . .	49
2.8.1 Pre-filter . . . . .	50
2.8.2 Job specific filter . . . . .	51
2.8.3 Feature selection . . . . .	52
2.8.4 Model selection . . . . .	53

2.8.5	Model evaluation . . . . .	54
2.8.6	Cross validation . . . . .	54
2.9	Analyses . . . . .	55
2.9.1	Identifying best model and feature combination . . . . .	55
2.9.2	Comparison system and experts on FSHD diagnosis . . . . .	58
2.9.3	Comparison system and experts on facial weakness grading . . . . .	58
<b>3</b>	<b>Results</b>	<b>59</b>
3.1	FSHD grading system evaluation results . . . . .	59
3.1.1	Selected features . . . . .	61
3.2	Comparison expert FSHD predictions . . . . .	64
3.3	Correlation between FSHD and facial weakness . . . . .	65
3.4	Facial weakness grading system evaluation results . . . . .	65
3.4.1	Selected features . . . . .	69
3.5	Comparison expert facial weakness scores . . . . .	72
<b>4</b>	<b>Discussion</b>	<b>74</b>
4.1	Expert FSHD predictions . . . . .	74
4.2	Weakness grading . . . . .	75
4.3	Evaluation of features . . . . .	76
4.3.1	FSHD features . . . . .	76
4.3.2	Facial weakness features . . . . .	77
4.3.3	Subtraction and motion features . . . . .	78
4.4	Methodological limitations . . . . .	78
4.4.1	Landmark and timeframe annotation . . . . .	78
4.4.2	Data acquisition . . . . .	79
4.4.3	Analyses . . . . .	82
4.5	Future directions . . . . .	82
<b>5</b>	<b>Conclusions</b>	<b>83</b>
	<b>References</b>	<b>84</b>
	<b>Appendix A Consent form</b>	<b>89</b>
	<b>Appendix B Expert score sheets</b>	<b>91</b>
	<b>Appendix C Analyses listings</b>	<b>95</b>
C.1	FSHD system evaluation . . . . .	95
C.2	FSHD features . . . . .	97
C.3	Weakness grading system evaluation . . . . .	99
C.4	Best weakness grading systems . . . . .	105
C.5	Weakness grading expert reliability . . . . .	107
C.6	Weakness grading features . . . . .	112
	<b>Appendix D Software tools</b>	<b>116</b>
D.1	Data collection . . . . .	116
D.1.1	SCHEDULER TOOL . . . . .	116
D.1.2	EXPERIMENTER TOOL . . . . .	117
D.2	Data management . . . . .	119
D.2.1	PARTICIPANT INDEXER . . . . .	121
D.2.2	DATA TAGGER . . . . .	121
D.2.3	DATA TOOL . . . . .	122
D.2.4	CUE TOOL . . . . .	123
D.2.5	Kinect 2 tools . . . . .	124
D.3	Annotation tools . . . . .	126

D.3.1	VIDEO POSITION MARKER . . . . .	126
D.3.2	DRMF fitter . . . . .	127
D.3.3	FACE MARKER . . . . .	128
D.4	Feature extraction tools . . . . .	132
D.4.1	FEATURE EXTRACTOR . . . . .	133

# Abbreviations

Abbreviation	Abbreviated
FSHD	Facioscapulohumeral muscular dystrophy
MSRA	Maximal Static Response Array
AFA	Automated Face Analyses
FACE	Facial Analysis Computerized Evaluation
OSCAR	Objective Scaling of Facial Nerve Function Based on Area Analysis
LBP	Local Binary Patterns
FACS	Facial Action Coding System
AUs	Muscle Action Units
FGS	Facial Grading System
SVM	Support Vector Machine
SVC	Support Vector Machine Classifier
L1-SVC	Support Vector Machine Classifier with L1 regularization
RLR	Randomized Logistic Regression
ENet	Elastic Net Regression
RF	Random Forest
RFR	Random Forest Regression
RL	Randomized Lasso
LR	Logistic Regression
kNN	k-Nearest Neighbors
kNNR	k-Nearest Neighbors Regression

# Chapter 1

## Introduction

Facioscapulohumeral dystrophy (FSHD, also known as the disease of Landouzy Dejerine) is an incurable hereditary slow-progressing muscular disease. With a prevalence of 12/100.00 it is one of the most common muscular diseases in adults [1]. The disease is primarily characterized by a gradual and often asymmetrical weakness of the muscles in the face, shoulders and upper arms, but other muscles are involved as well. In early stages of FSHD the muscles of the shoulder girdle (see Figure 1.1) and the face are affected, subsequently the upper arms and in later stages the muscles near the abdomen, the legs and the trunk of the body get involved. The degree of muscle involvement is known to be highly variable between patients, ranging from isolated facial weakness to severe generalized weakness, with approximately 20% of the patients eventually requiring a wheelchair. Even between and within affected families, clinical severity and age of onset may vary widely [2]. Moreover, there is a high percentage (20 – 30%) of asymptomatic gene carriers [3]. The progression of the disease also varies between cases. For a large number of affected persons the symptoms do not appear. The age of discovering the first symptoms range from early infancy to late adulthood. Usually the first symptoms are discovered during adolescence [3]. Respiratory muscles and the myocardium are typically spared. Life expectancy is generally not reduced [2].

FSHD is autosomal dominantly inherited, which implies that if one parent has FSHD then each child has a fifty percent chance of inheriting the disease. However in 10 – 33% of all cases the disease occurs due to spontaneous mutation. FSHD has two forms: FSHD type 1 and 2, which have different genetic loci. FSHD type 1 is the most frequent, which is present in 95% of all FSHD cases and is caused by an anomaly in the DNA. The anomaly is the deletion of a repeating piece of DNA at the end of chromosome 4, which is located at 4q35 in the D4Z4 DNA region [4]. It was discovered in 2010 that this missing piece of DNA causes the production of a protein that causes the weakening of muscles [5].

There are various tests for diagnosing FSHD, although the most common are physical examination and DNA testing. The most conclusive test is an examination of the DNA for FSHD type 1, which covers most cases. If the test turns out negative, but there are strong FSHD related symptoms then a test for FSHD type 2 can be conducted. However DNA testing might take some time, since the DNA needs to be extracted and analyzed with specialized equipment. Hence, the first diagnostic test is usually a physical examination, where a physician looks for a characteristic pattern of muscle weakness and asks for known cases of FSHD within the family. When FSHD is present in the family and the pattern of muscle weakness is clearly present, DNA testing is sometimes considered unnecessary.

FSHD brings great complications and uncertainty for affected people. Due to the gradually increasing muscle weakness, a lot of physical tasks become more difficult to perform, which



**Figure 1.1:** Shoulders of two FSHD patients while stretching their arms.

often cause muscle pain and fatigue. For example, people with FSHD might no longer produce all facial expressions, have difficulty lifting their arms, getting up when on their back, fall more and experience difficulties when walking. Gradually simple tasks become harder to perform and require help with daily routines. Also, the weakening of the lower body eventually causes 20% of all FSHD patients to require a wheelchair [3].

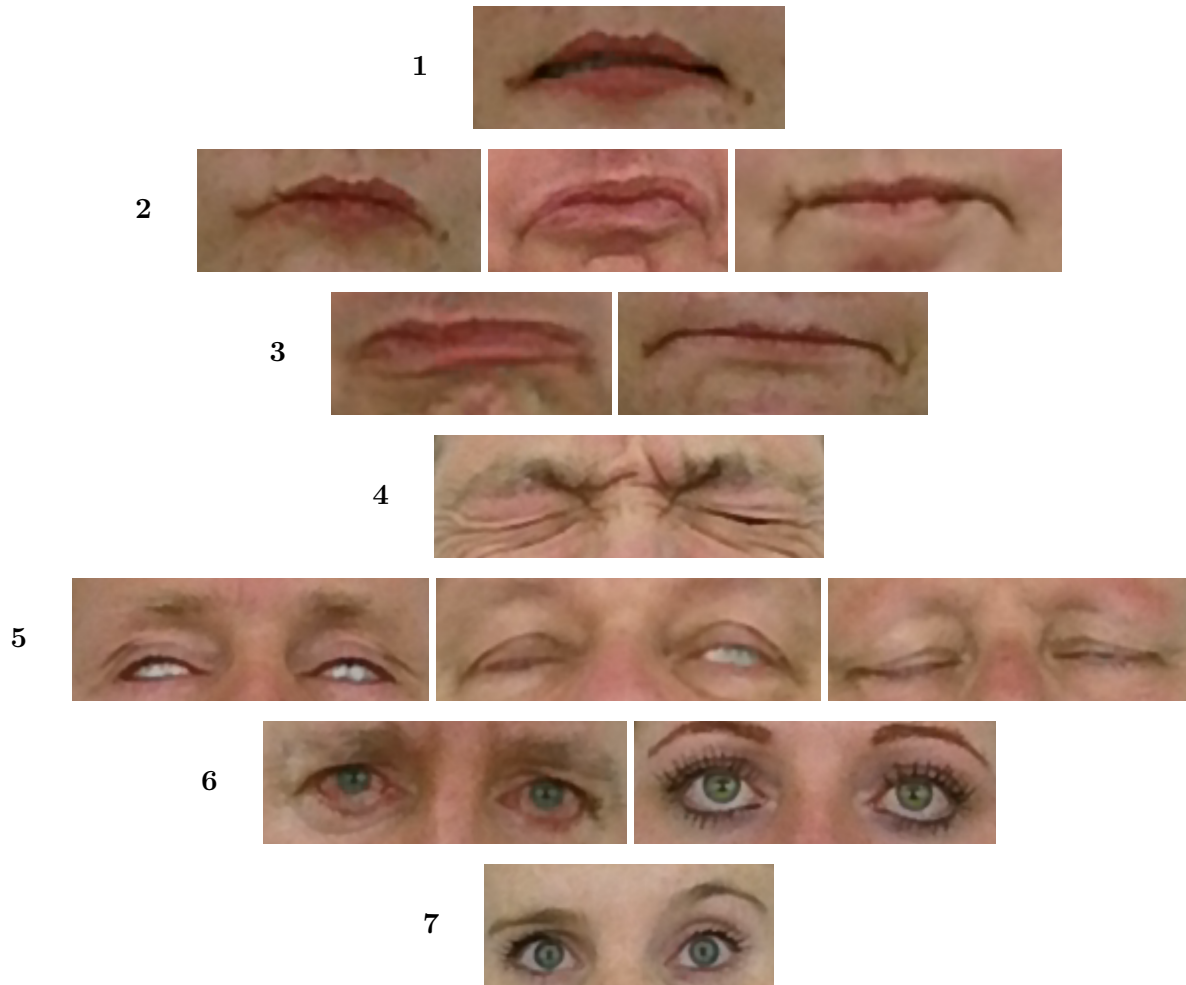
Besides the physical limitations that come with the disease there are also a lot of social implications for people with FSHD: they can lose self esteem, they are forced to quit their jobs, they are faced with a frustration of not being able to express themselves anymore and they are increasingly dependent on others.

One of the greatest difficulties due to the waste of facial muscles with FSHD, is the gradual diminishing of facial expression, which hinders communication and social interaction and changes the way they are perceived by others. A lot of FSHD patients, for example, can no longer raise the corners of the mouth, which is required to produce a smile. Hence they can be perceived as arrogant or sad.

Currently there is no cure for FSHD, but there are several ways to make living with the disease and its consequences more manageable. There are so called symptomatic treatments, which target the symptoms rather than the disease itself. For example a recent study has shown that a treatment involving aerobic exercises combined with cognitive therapy can help to reduce fatigue symptoms within FSHD patients [6]. Furthermore, people with FSHD are advised to seek out council with rehabilitation specialists and paramedics who can provide advice and guidance.

## 1.1 Measuring facial weakness

One important part of diagnosing FSHD through physical examination is the assessment of facial weakness. For FSHD patients it was found that the circular muscles around the mouth (orbicularis oris) and the muscles controlling the raising of the mouth corners (zygomaticus) are the most commonly affected muscles in early stages of the disease. This results in an inability to raise the corners of the mouth and thus patients were reported to produce a smile moving in a horizontal direction, which looks more like grinning and thus affects emotional expression (see Figure 1.2 (1)). When the orbicularis oris is very weak, then tasks like pursing of the lips, whistling and puffing of the cheeks become impossible (see Figure 1.2 (2-3)). Also, the circular muscles around the eyes (orbicularis oculi) are affected. In the beginning of the



**Figure 1.2:** (1) Patient showing her teeth. (2) Patients pursing their lips. (3) Patients puffing their cheeks. (4) Patients closing the eyes forcefully. (5) Patients closing the eyes gently. (6) Patients showing rim of the sclera below the pupils in rest. (7) Patient attempting to raise both eyebrows.

muscle weakening the eyelashes cannot be completely hidden when forcefully closing the eyes (Figure 1.2 (4)) and when orbicularis oculi muscle becomes weaker a small rim of the sclera becomes visible when closing the eyes (Figure 1.2 (5)). Additionally, when the orbicularis oculi is weak, below the eye a small rim of the sclera may become visible at rest (Figure 1.2 (6)). FSHD is usually associated with asymmetric involvement of the facial muscles (e.g. Figure 1.2 (7)). In later stages of the disease when a lot of muscles are involved, unwrinkled foreheads and expressionless faces are reported. Due to the high variability between involved muscles, for 16.8 – 18.7% of FSHD patients there are no facial muscles involved, but this group usually has muscle involvement in other muscles of the body like the shoulders and the legs [3].

The most traditional and straightforward method for describing facial weakness for a trained physician, is to ask a person to perform certain tasks involving the facial muscles like closing the eyes firmly, raising the eyebrows, smiling and pursing of the lips. Then the physician observes the face during a task and documents the facial function. The system for scoring and documenting facial weakness is important, since it allows the clinician to determine the severity of disability, to communicate the information with colleagues and to evaluate response on therapy. This is especially important for scientific research, where the effect of novel medicine



and treatment should be reported in a reliable and valid way.

Grading facial function is challenging, since the anatomy and physiology of the face are complex. Also giving a gross judgment based on observed loss of facial function is inherently subjective in nature, which has been reported to result in great inter- and intra-observer evaluation errors [7]. The use of objective measurements may decrease inter-observer variability. However, muscular weakness in different regions of the face does not always express similarly. For example, muscular weakness around the eyes and mouth are detected more easily than weakness in the forehead. Furthermore, if a weakness is only visible when the face is moving, it might be harder to find than one that also involves the face at rest. Finally, patient factors, like sex, skin, wrinkling, facial asymmetry and facial contour have an effect on measurements as well. All these factors make it hard to give a uniform assessment of facial weakness.

Ideally a method for measuring facial weakness will be created that is: reliable, sensitive, valid, relevant, simple and inexpensive. Especially the requirement of simplicity has been identified as important [8], since for clinical adoption it should be usable by clinicians without a lot of technical knowledge and training. However, creating a system that satisfies all requirements can be considered a hard task, since usually some trade-offs are made between sensitivity and simplicity. Considering the recent developments in DNA research [5] a rise in experimental therapies and treatments for improving the quality of life for FSHD patients is to be expected. Since these methods need to be scientifically evaluated on their effect on facial weakness, novel facial weakness measures are needed. Additionally, the effect of treatment can only be evaluated once the natural course within untreated patients is charted. Therefore, a grading system for facial weakness is also of importance to natural course studies. Hence, the need for an objective measuring method for facial weakness for FSHD is clear, which is further emphasized by the lack of current quantitative methods.

Testing the suitability and reliability of a grading system generally requires comparison of the new system with other, preferably established, grading systems. Experts will be asked to grade videos of patients with varying facial weaknesses performing several tasks and to score their facial function using some grading scale. Consecutively the inter- and intra-expert gradings can be compared on agreement. Some scales are also evaluated for repeatability measuring the so called test-retest reliability. Here the same measurement is performed on the same individual, under the same condition, but over a short period of time. If a difference in outcome is obtained, it causes test-retest variability. An outcome can be considered repeatable when this variation is smaller than a predetermined acceptance threshold. Test-retest variability can be caused by both intra-observer variability and intra-individual variability.

M.J. Brenner and J.G. Neely [9] report on various approaches for grading facial nerve function and distinguish between “traditional approaches” and “computer-based approaches”. Traditional approaches refer to methods that do not require specialized computer equipment. Many of those approaches rely on the use of subjective assessments by the observer. Computer-based approaches use computer equipment to objectively measure and quantify digital data, which involve collecting facial features from digital data using various techniques.

## 1.2 Traditional approaches

The traditional approaches from this section all originate from literature on grading facial weakness of patients with facial paralysis. Facial paralysis can result from a number of causes, e.g. head trauma, stroke, tumors or infection of the facial nerve, and can cause inability to express emotion, drooling and facial pain. The most common form of facial paralysis is Bell’s palsy. Bell’s palsy patients have usually one-sided facial paralysis due to the dysfunction of

a cranial nerve, which impairs motor control. For most patients with Bell's palsy complete recovery occurs in about 6 months. Since facial paralysis is a different disease from FSHD, it renders all described grading systems invalidated for measuring facial weakness within FSHD. However, since there is no existing work on grading systems for FSHD, the existing grading systems were used to draw inspiration from.

The traditional approaches for grading facial function go back several years and have greatly developed over the years. The first systematic methods were developed after 1970. Before that time were mostly descriptive reports [10] instead. In 1983 J. House presented a conceptual framework for classifying such scales as gross, regional or specific [11]. Gross scales are used for evaluating the overall severity of loss of motor function of the face and are primarily categorical and thus specify a qualitative assessment. Regional scales give independent scores to different areas of facial function and weights can be assigned to the scores to indicate greater or lesser importance to facial areas, like eye closure or mouth movement. Lastly specific scales require an observer to answer yes or no to questions related to specific areas of the face.

One well established system for the subjective assessment of facial nerve damage is the House-Brackmann grading system [12]. It was initially developed in 1983 as a gross scale after J. House reviewed eight different grading scales, and was later modified by Brackmann and Barrs [13], which finally resulted in 1985 in the House-Brackmann grading system [12]. The grading scale was originally designed for predicting recovery in Bell's palsy patients. The system is shown in Table 1.1 and is a gross scale with six categories ranging from I (normal) to VI (no movement) for grading overall facial ability. It has been adopted as the North American standard since 1985 due to its simplicity, and is used world wide in practice to grade patients with facial paralysis.

Although the House-Brackmann scale has been useful to standardize reporting of facial weakness for facial paralysis, it has its drawbacks. First of all it is unable to distinguish between subtle changes in facial function, because of the wide variety in facial movements that are all encompassed by only 6 categories. Secondly, there might be differences between two facial regions regarding facial movement, which might require different scores [14]. In a situation like that, the system only provides a single overall grade. Lastly, the scale has been reported to have significant limitations with respect to inter-observer agreement [9].

To deal with the issues from the House-Brackmann scale, so called linear systems have been proposed. The Burres-Fisch system [15] was designed to minimize observer bias and subjectivity. It is among the first objective systems that defined 15 reference points on the face (6 landmarks for each side of the face and 3 landmarks at the mid-line of the face) for which the distances should be calculated. The measurements were performed using hand held calipers and electromyographic surface electrodes. The outcome produced an objective linear continuous graded scale for facial weakness. The Burres-Fisch system did not become widely used because it was not possible to make simultaneous recordings of different facial regions and the measure procedure and involved calculations were very time-consuming and cumbersome (approximately 20 minutes [16]). Later, the Nottingham system [16] was proposed and was similar in nature to the Burres-Fisch system, but used only 8 landmarks and used more simple calculations, which allowed for a more rapid assessment (within 3 minutes) and it correlated better with the House-Brackmann grading system. However, there are limitations, including the inability to assess bilateral facial weakness. Another drawback of linear systems is that they do not necessarily correlate directly to other facial grading systems and are reported to measure different information. Also area movement is not considered. This poses an additional problem, since facial area movement near a linear measurement might move in the opposite direction to the linear measurement [17].

Grade	Description	Characteristics
I	Normal	Normal facial function in all areas
II	Slight Dysfunction	Gross: slight weakness noticeable on close inspection; may have very slight synkinesis At rest: normal symmetry and tone Motion: forehead - moderate to good function; eye - complete closure with minimum effort; mouth - slight asymmetry
III	Moderate dysfunction	Gross: obvious but not disfiguring difference between two sides; noticeable but not severe synkinesis, contracture, and/or hemi-facial spasm. At rest: normal symmetry and tone Motion: forehead - slight to moderate movement; eye - complete closure with effort; mouth - slightly weak with maximum effort.
IV	Moderate severe dysfunction	Gross: obvious weakness and/or disfiguring asymmetry At rest: normal symmetry and tone Motion: forehead - none; eye - incomplete closure; mouth - asymmetric with maximum effort.
V	Severe dysfunction	Gross: only barely perceptible motion At rest: asymmetry Motion: forehead - none; eye - incomplete closure; mouth - slight movement
VI	Total paralysis	No movement

**Table 1.1:** House-Brackmann Scale. Table was taken from the original House-Brackmann paper [12]

Yet others proposed improvement over the House-Brackmann grading system subjective scales, like the Sunnybrook scale by Ross et al. [17] and the House-Brackmann grading system 2.0 [18]. The Sunnybrook scale provided weighted subjective region scales for the calculation of single gross score and was reported by the authors to be able to distinguish between certain levels of facial function that the House-Brackmann scale could not. The House-Brackmann grading system 2.0 [18] has been proposed as an improvement over the original system, which has sharpened some of the categorical descriptions, which was reported to have improved intra- and inter-observer agreement. However, both approaches remain subjective scales.

The main advantage of traditional grading systems, is that they are easy to use by clinicians and they can present a simple numerical description of facial function. The downsides of using such systems are that they are inherently subjective with much inter- and intra-observer variation, and are not always informative for the localizing facial weaknesses [7]. To reduce subjectivity, objective or linear methods using distances and hand held measuring instrumentation were developed, but such methods are usually very time-consuming, can be difficult to consequently reproduce and are not always informative for different facial regions. The major issues with the traditional approaches are the subjectivity of grading and the bias introduced by human observers.

## 1.3 Computer-based approaches

Since traditional methods did not produce sufficiently objective, meaningful, usable and reproducible measuring systems, researchers kept looking for alternatives. Diverse computer and digital processing techniques have been proposed in order to produce accurate and reproducible systems. The technology involved ranged from simple photograph analysis to advanced 3D video scanning equipment. Computer-based methods can remove the burden of doing calculations by hand and might provide more accurate and reproducible results. The computer-based approaches can be roughly subdivided into facial landmark approaches and subtraction methods. Facial landmark approaches define a set of points on the face. The position of the landmarks subsequently compared for the face at rest and for the face at maximal expression. Subtraction methods try to measure movement for a facial area using simple computer vision techniques.

In some of the more modern computer-based approaches machine learning techniques are used [19, 20]. Machine learning deals with algorithms that allow computers to learn and infer from data. Machine learning algorithms are used in many applications and domains. Supervised machine learning algorithms deal with problems for which labeled data is available, i.e. they involve a set of examples or features  $x_1 \in X$  which have a corresponding ground truth  $\{y_i\} \in Y$  and a learning algorithm  $F : X \rightarrow Y$ , which learns a connection between  $X$  and  $Y$ . If  $Y$  is discrete and has a finite number of classes, it is called a classification problem. Otherwise if  $Y$  is continuous it is called a regression problem. In practice such machine learning algorithms are first trained giving it a sufficiently large feature-ground truth pairs  $(x_i, y_i)$  after which it assigns a novel input feature vector without a ground truth an inferred value.

In the case for grading facial weakness, one application is to learn a computer to assign grades from a human observer. A human expert observer can provide a ground truth by labeling a set of videos of faces performing a certain task like smiling or lip pucker on a certain grading scale (e.g. House-Brackmann grading scale [12]). Subsequently, features relevant to assessing facial weakness are extracted from the same videos, like the distance between various landmarks or the number of pixels moved around the mouth. Next, the extracted features and the ground truth are passed to the algorithm, which learns the connections from the features to the provided ground truth. On novel instances of faces, features can be extracted again for which the trained algorithm can estimate the outcome based on the underlying data.

### 1.3.1 Facial landmark systems

Facial landmark systems rely on calculating movement for a predefined set of facial landmarks. The most simple systems only compare movement between the face at rest and the face at maximal expression, while more complex systems create landmark movement profiles over time. Landmarks can be labeled by hand on video or photographs, they can be marked on the face using physical markers and subsequently be tracked from video or they can be obtained from video using face recognition and landmark extraction techniques.

#### Landmark systems with physical markers

A lot of systems use the placement of a set of physical markers on the face [21, 22, 7, 23]. Markers can vary in shape (e.g. adhesive dots, stickers) and size (2 – 6 mm), but usually have a clear contrasting color to the skin. The distinct color allows a computer to easily track the markers from video using a color threshold. Additionally, the distinction between active and passive marker based systems can be made. Passive markers are only reflective materials that

should contrast with the skin. Active marker systems on the other hand use markers which emit a signal on their own, which could be used to identify a marker.

Johnson et al. [21] developed a method called the Maximal Static Response Array (MSRA) using facial markers (dots) that were placed on key positions of the face and also a ruler to determine the pixel to metrics. Subsequently photographs of the face in rest position and of the face at maximal task expression were taken. A grid was then placed over the images and was used to quantify the displacement of the dots between the two positions. A ruler was positioned in the photographs, to enable recalculating the number of pixels to metrics. However, this was reported to be prone to error due to head tilt for some of the subjects. The process was also time-consuming and introduces manual measuring and marking errors. Additionally, the method does not incorporate displacements over time.

Isano et al. [22] used 24 different markers that were placed on relevant positions on the face and were recorded using a video camera. Ten frames were analyzed per movement from rest to maximal movement. They used 44 healthy subjects and 11 subjects with varying degrees of facial paralysis and was reported to be time consuming yet effective in measuring facial displacement.

Linstrom et al. [7] used the Peak Motus Motion Measurement System, which is a system used in other motion studies like physical therapy, orthopedics and sports, recording black and white video. Their method uses 24 reflective markers placed on the face at anatomic reference points and was recorded on video. They recorded 34 normal subjects and 26 subjects with abnormal facial function with different etiological backgrounds. The ratios between the left and right parts of the face were compared and plotted over time, showing the amplitude and velocity of facial motions. The results showed to be effective for measuring facial movements, but it was very time consuming to perform and subjects reported placement of the markers to be unpleasant.

Frey et al. [24] were one of the first to introduce and emphasize the use of 3D measurements. They emphasize the potential of 3D measurements for improvement upon 2D methods regarding accuracy and robustness. They developed the VICON Motion System (VICON Motion Systems, Lake Forest, California), which is a complex system of mirrors, calibration grid, digital video camera and specialized software, that can track the 3D trajectories for up to 18 facial markers. Two of the facial markers in the mid-line are static and serve as registration points whereas the other 16 landmarks are dynamic.

Recently Azoulay et al. [19] created a mobile application that uses machine learning techniques to assign a House-Brackmann facial grading score to a face video from a person with 13 reflective stickers placed at reference points on the face. From the landmarks they calculated features that would capture asymmetry. For each of 9 predefined tasks 11 features were extracted including a combination of triangle areas and linear distances. For each measurement the temporal value was determined during a task, which were normalized by its initial value at rest. Then the values for the right and left sides of the face were subtracted. Finally, the features over time were set at the value with the maximal movement for each task, resulting in 99 features. Two machine learning models were trained on the features: a Support Vector Machine [25] for classifying whether a person was sick or not and a Ridge Regression model [26] with a third degree polynomial to determine the severity of facial asymmetry using the House-Brackmann facial grading score. High correlations with the evaluated grading scales were reported. There are several interesting aspects to this approach: it uses a mobile application and the calculations are fast. The possibility of gathering a lot of user data becomes also apparent. A drawback of the method is that the placement of the physical markers is required by the patients, which might compromise the accuracy and reliability of the method.

Facial landmark systems employing physical markers are reported to accurately capture facial movements and make landmark tracking through image processing simpler in both 2D and 3D. However, the marker placement is considered to be obstructive [7] and was found to be difficult to accurately replicate by clinicians or technical personal [27]. Additionally, the whole grading process can be quite time consuming, which hinders standardization.

### Landmark systems without physical markers

Because of the advances in computer vision and facial landmark recognition and tracking and the availability of commercial facial tracking software, e.g. FaceReader [28], 3DMDface<sup>TM</sup>, DI3D<sup>TM</sup> and FaceShift<sup>TM</sup> there is an increasing number of facial analysis grading systems that employ tracking and detecting landmarks for evaluating motion without the use of physical markers. These methods are interesting in the sense that they do not require the placement of markers on subjects and could potentially be fully automated. However, tracking of landmarks without physical markers is a challenging technical task, which might introduce additional bias and some more technical expertise from clinicians.

Wood et al. [29] introduced a method called video microscaling in which videos were superimposed with a computer generated scale to measure distances between facial landmarks digitally. 11 subjects were video recorded while raising the eyebrows and smiling 5 times on 2 different days using this method. The authors found low average variability. However, the process was inconvenient, since the method could only track a single vector movement and not multiple markers movements at once and no information about velocity or acceleration of movement was given.

Wachtmann et al. [30] compared a 2D automated facial feature tracking method called automated face analyses (AFA) developed for detecting and extracting emotions and paralinguistic expressions [31] with a manual marking method called the Maximal Static Response Array (MSRA) [21]. For MSRA the maximum effort frame and the repose image frames are determined for a task from videos. Then both frames were manually labeled. For AFA the first frame of the video at rest was labeled by adding virtual markers using a computer and subsequently track the markers for the other frames using the Lucas Kanade [32] algorithm. Their method shows promising results for automated marker tracking, but 2D methods were reported to underestimate 3D displacement by 43% [33]. Because of this finding, the authors finally suggest that moving towards 3D tracking might be more promising.

Mishima et al. [34] produced a very complex system involving multiple depth and color cameras and a special helmet in order to track 3D motion of the lip in real time. Although the system is claimed to reliably track the movements and various landmarks of the lips, the required equipment makes it difficult to adopt and extend the system beyond the domain of lip tracking.

Popat et al. [35] reviewed a number of three-dimensional imaging techniques which could be used for the analysis of facial movement. They also reported on two commercially available systems: 3DMDface<sup>TM</sup> dynamic system and the 4D capture system DI3D<sup>TM</sup>. 3DMDface uses active stereophotogrammetry and projected unstructured infra-red light to capture 3D data. The setup uses six 1.3 mega pixel cameras (four gray scale and two color cameras) that record at 60 frames per second. Additionally, the systems support audio and time recording. The DI3D systems use passive stereophotogrammetry and come in varying sizes and formats. A basic DI3D capture system uses a pair of low noise 4 mega pixel monochrome cameras that capture 3D data up to 60 frames per second. Both systems produce output files which describe 3D geometry. However, the systems can be quite expensive and there are not many data available that compare the systems.

The advent of marker-less systems able to track the face in 2D and 3D, open numerous of possibilities for clinical facial grading systems. The systems automatically position the landmarks on the face, which removes the need for accurate physical marker placement and could possibly achieve higher spatial resolution. However, marker-less systems often introduce expensive and elaborate equipment and the challenging task of landmark localization. Furthermore, depending on the acquisition procedure, subjects have to keep their heads quite steady during task performance for reliable landmark detection and tracking.

### 1.3.2 Subtraction methods

In contrast to using landmarks, image subtraction techniques estimate the facial movement from areas of the face. These techniques are all digital and can involve images or video recordings. A computer is used to calculate the difference between a frame at rest and subsequent motion frames to determine a measure of facial movement over time. Generally the difference between two images can be calculated by first converting the images to gray-scale and by subsequently subtracting their pixel intensities from one another. When subtracted pixels intensities are near 0 (black), there is approximately no movement and when the pixel intensities are near 255 (white) there is much movement for that pixel. Finally, to obtain a measure for movement within a face area, the subtracted pixels within a face area can be summed.

One of the earlier subtraction methods was the Facial Analysis Computerized Evaluation (FACE) method developed by Neely and Cheung in 1988 [36, 37, 38, 39, 40]. Within this work they video recorded subjects performing voluntary facial expressions under highly controlled conditions, i.e. the illumination was kept constant, a fixed camera distance was used and the head of the subject was stabilized using a head mount. The videos were digitized and processed by subtracting the rest frame at the start of the movie  $I_0$  from each other image  $I_i$  (with  $i > 0$ ) using an interval of approximately 100 milliseconds. The degree of facial surface deformation was measured for the forehead, eye and mouth by summing the gray scale values of all the pixels per region per subtraction pairs. This boils down to the following formula, which calculates the absolute difference  $F_i$  for each pixel value for image  $i$  out of the sequence for a set region  $R$  of pixels:

$$F_i = \sum_{(x,y) \in R} |I_i[x, y] - I_0[x, y]|$$

By calculating this quantity for all images, the curve amplitude is obtained, i.e. the summed gray values over time for each area for each facial expression. A maximal curve amplitude for the abnormal side of the face was calculated by subtracting the gray scale values at the end rest conditions from the mean maximum amplitude value on the abnormal side. They also defined a composite index for computing an overall score based on curve type and maximal curve amplitudes. Thus, they provided a measure for facial regions, and for the overall facial function.

Neely et al. later evaluated their approach by recording 27 patients with varying diseases affecting the face and compared their system with House-Brackmann grades [12] given by human observers, which were highly correlated.

Meier-Gallati and Scriba et al.[41, 42] independently developed the Objective Scaling of Facial Nerve Function Based on Area Analysis (OSCAR) method that is very similar to the FACE method. The measuring procedure takes 10 minutes and also involves a head mount to ensure the head remains in the same position for the rest and at maximum expression. The

disadvantages for both the FACE and OSCAR systems are the need for rigor fixation of the face, the required time and the inability to perform linear vector analysis between landmarks.

The work of Shu He et al. [27] introduced a method for automatic localization of facial regions, image stabilization, subtraction of key movements and estimation of motion magnitude by image subtraction and optical flow from biomedical videos of the face. Additionally, several machine learning classifiers were trained (Support Vector Machines, k-Nearest Neighbors and Radial Neural Networks) to give a qualitative House Brackmann grade estimate. Their method was tested on gray videos for various subjects that were asked to perform five facial movements: raising eyebrows, closing the eyes gently, closing the eyes firmly, screwing up the nose and smiling.

Shu He et al. use an horizontal/vertical projection to search for the facial regions within the video signal. Horizontal and vertical projection work well for face localization in uncluttered background. The method encompasses summing the horizontal and vertical intensities:

$$HI(x) = \sum_{y=1}^n I(x, y) \quad VI(y) = \sum_{x=1}^m I(x, y)$$

Subsequently, a Sobel filter was applied to extract the facial contours like the nose, mouth and eyebrows and face outline. Then, by calculating the intensity histograms, the left and right  $x_{left}, x_{right}$  regions of the face were determined together with the top  $y_{top}$  of the head. These positions are used to estimate the center of the head to subsequently apply a thresholded Gaussian-weighted image for removing the hair and shoulders within the image. Then using vertical and horizontal projection once more the regions of interest can be set to further refine the pupils and mouth corner positions.

Since subtraction methods can only assess the magnitude of the movement and not the direction of movement, an optical flow method was also used which can track the direction of movement. This information can in some cases be very relevant, e.g. when a person with paralysis to one side of the face smiles, the normal functioning side of the face might pull the paralyzed side. In such a case a subtraction method will detect movement on the paralyzed side, but an optical flow method will detect that the movement is not towards the expected side. The optical flow is calculated by putting a grid of equally distributed pixels on top of the image at rest and subsequently tracking the changes for the image at maximal motion using the Lucas-Kanade algorithm [32].

Later work of Shu He et al. [20] replaced their optical flow method with Local Binary Patterns (LBP). A Local Binary Pattern is a type of feature that is found to be powerful for texture classification and was originally introduced as the Texture Spectrum model by Wang and He [43], but was later modified by Ojala et al. [44, 45] to a true binary version. Computing the LBP for a pixel is best explained by using a  $3 \times 3$  image with differing pixel (e.g. gray-scale) intensities. Computation involves taking the pixel intensity of the center pixel and to compare it with the other 8 pixels. If a neighboring pixel has an intensity value lower than that of the center pixel, that pixel is set to 0, otherwise that pixel is set to 1. By now concatenating the 8 neighboring pixels in a predefined order the LBP has been obtained, which can be turned into an integer for convenience. The main advantage of LBP is that it is an intensity invariant feature, i.e. it is not affected by global illumination variations. LBP are however sensitive to rotation, since the order of the neighbors determines the encoding. Their LBP approach performed better than their earlier optical flow approach and was reported to be computationally faster.

Subtraction techniques show much promise for being reliable measures for facial motion. Especially their ability to address motion within facial areas makes their applications interesting compared to landmark methods, that mainly measure linear vector movements. The approach



seems also interesting for the detection of key movement (rest state and most active state during tasks) within videos. Additionally, optical flow estimation techniques and Local Binary Patterns can determine directional information.

## 1.4 Other approaches

Besides landmark and subtraction methods, there have been other proposed methods within the literature that do not fall into those categories, but are nonetheless interesting for developing facial grading systems. Within this section some of this work is described, including moiré topography [46] and the Facial Action Coding System (FACS) [47].

Moiré topography [48] has been applied for objectively assessing facial function. The method involves the use of optical stripes that are projected by light through a grid upon the face of a subject to create a facial contour map. The deformations of the optical stripes can be measured and analyzed to quantify small differences in facial contour. The drawback of this approach are the need for specialized equipment, its time-consuming nature and the knowledge needed to perform the technique. Because of these limitations the method has not been used much.

Some work from the early 1970s by Paul Ekman et al. [47] might have some potential use for computerized grading of facial weakness. Their research tried to characterize relationships between facial expression and emotions. They developed an anatomically based system to describe muscle action units, termed the Facial Action Coding System (FACS). The system provides a way to identify the elementary muscle action units (AUs) that constitute facial movement.

Some research teams have been looking to automatically detect AUs from videos and worked on the reliable recognition of muscle action units. The system by Cohn et al. [49] is such a system called the Automated Facial Analysis (AFA). Their system provides a way for automatic recognition and coding of AUs and aimed to remove the need for manual coding in the long run. Their method uses image stabilization techniques and feature point tracking using the Lucas-Kanade algorithm [32].

Tracking action units with the Automated Facial Analysis (AFA) program and Moiré topography can be applied to assess facial weakness. However, the most promising systems are likely to involve human observer scales and landmark-based systems [14, 15, 16].

## 1.5 Overview of facial grading systems

The reviewed literature covers a lot of different approaches and different grading systems. It should be emphasized that most of the systems were designed for detecting facial weakness for people with facial paralysis rather than people with FSHD. This is because the former is far more common than the latter and therefore there is almost no established literature on facial weakness grading systems for FSHD. Facial paralysis often causes more severe facial weakness than FSHD and hence the reviewed methods might not automatically be sufficient for FSHD. However, the systems can be used as inspiration for designing facial weakness grading systems for FSHD.

An overview of the reviewed literature on facial grading systems (FGS) can be found in Table 1.3. Each system is categorized by year, method and estimated/reported duration for calculating the grades. For the methods within the table, abbreviations were used that can be found in Table 1.2.

abbreviation	meaning
T	traditional approach
C	computer-based approach
O	other approach
M-n	uses n landmarks
F	uses physical face markers
V-mode	uses videos, where mode can be color (rgb) or black and white (gray)
P	uses photographs
ML-methods	uses machine learning algorithms, methods can be artificial neural networks (ANN), ridge regression (RR) and support vector machines (SVM)

**Table 1.2:** Abbreviations used to categorize the facial grading systems, with their explanation.

system	year	method	remarks	duration
May FGS [10]	1970	T	subjective, gross scale	< 1 min
House Brackmann FGS [12]	1985	T	subjective, gross scale	< 1 min
House Brackmann FGS 2.0 [18]	2009	T	subjective, gross scale	< 1 min
Sunnybrook FGS [17]	1996	T	subjective, region scale	< 1 min
Burres-Fisch [15]	1986	T, M-15	linear hand measurements	20 mins
The Nottingham system [16]	1994	T, M-8	linear hand measurements	3 mins
Johnson et al. [21]	1994	C, M-9, F, P	MSRA, linear, grid	> 20 mins <sup>?</sup>
Isano et al. [22]	1996	C, M-24, F, V-rgb	distance over time	> 20 mins <sup>?</sup>
Linstrom et al. [7]	2002	C, M-26, F, V-gray	Peak Motus Motion Measurement System	> 20 mins <sup>?</sup>
Frey et al. [24]	1999	C, M-18, F, V-rgb	VICON 3D measurement through mirrors	> 20 mins <sup>?</sup>
Azoulay et al. [19]	2014	C, M-13, F, V-rgb, ML-RR&SVM	mobile app	5-10 min
Wood et al. [29]	1994	C, M-18, V-rgb	video microscaling	> 20 mins <sup>?</sup>
Wachtmann et al. [30]	2001	C, M-18, V-rgb	2d tracking, no markers	> 20 mins <sup>?</sup>
FACE [37, 36, 38, 40, 39]	1988-1996	C, S, V-grey	subtraction	10 mins <sup>?</sup>
OSCAR [41, 42]	1998	C, S, V-grey	subtraction	10 mins
Shu He et al. [27, 20]	2007-2008	C, S, V-grey, ML-ANN	tracking, subtraction, Local Binary Patterns	5 mins
Yuen et al. [48]	1997	O, P	Moiré topography	> 20 mins <sup>?</sup>

**Table 1.3:** Overview of the reviewed facial grading systems (FGS) describing year, method, average method duration and additional characteristic remarks. Explanation of the method abbreviations can be found in Table 1.2. A <sup>?</sup> indicates a time estimation by the author, i.e. no reported times were found in the literature.

## 1.6 Project aim and motivation

This project has been initiated by the department of Neurology at the Radboud University Medical Center Nijmegen with a focus on developing an objective measure for quantifying FSHD. The availability to the large FSHD patient base from the outpatient clinic at the department of

Neurology at the Radboud University Medical Center Nijmegen provides a valuable opportunity to recruit a substantial number of patients to collect data from. Currently there is no known study on the development for a computerized facial grading system for objectively measuring facial weakness for FSHD patients within the literature.

However, the need for such a grading system is urgent, since recent advances in FSHD DNA research will likely steer future research towards the development of various medicine and treatment studies for FSHD. Such scientific studies will require an objective quantifiable measure of facial weakness for FSHD in order to evaluate and compare the effectiveness of medicine and treatment. Furthermore, when such a grading system can be used for monitoring the progression of facial weakness and is able to detect minor facial weakness within FSHD patients, it might improve the diagnostic process. Additionally, such systems might find their use within other medical domains and eventually may find a standardized place within clinical practice.

With the appearance of modern and affordable depth sensors like the Microsoft Kinect sensor [50] and the availability of numerous facial landmark tracking software [28] and powerful machine learning techniques, there are numerous possibilities towards a computerized objective facial weakness measuring method from video data, without physical markers for FSHD.

## 1.7 Research goals

The aim of the project is to develop a modern computerized marker-less facial weakness grading system that gives an objective measure for facial weakness within FSHD patients, and finally to validate and compare the outcomes of such a system with judgments of human experts, which serves as the current golden standard for measuring facial weakness. Thus, the main research question becomes:

*“Is it possible to develop a computerized marker-less facial weakness grading system that gives an objective measure for facial weakness for FSHD patients which has comparable performance to the current golden standard?”*

Regarding this research question there are several sub-questions that need answering. There are multiple descriptions of systems that measure facial weakness, but none so far have tried to relate facial weakness and calculate an objective outcome for FSHD. To do that, the relation between facial weakness and FSHD should be charted.

Furthermore, automated marker-less systems usually require a form of landmark tracking or region localization. It is desirable to find out if current available automated facial marking tools are sufficiently accurate for this task.

It is helpful to know if a computer model can diagnose FSHD based on facial characteristics and how that compares to human expert diagnosis.

The next question that needs answering is: what does a facial grading system encompass and what machine learning models and facial features work best for this task?

Additionally, we want to know if 3D depth information acquired from a Microsoft Kinect 2 sensor can help to improve the outcome classification of facial weakness with regard to solely 2D information.

## 1.8 Within this thesis

This thesis is organized as follows: Chapter 2 describes the methods used for collecting the data, extracting features from the data and for training and describes various approaches used for obtaining and evaluating grading systems. Chapter 3 lists all results. Within Chapter 4 the results are discussed, giving an answer to the research question and deriving directions for future work. Finally Chapter 5 summarizes and concludes this thesis.

## Chapter 2

# Methods

### 2.1 Data acquisition

A novel dataset was created containing video recordings of the front and the sides of FSHD patients from the patient-base of the Radboud University Medical Center Nijmegen and healthy controls over seven predefined tasks. A physical video recording installation with four cameras was created for recording the participants and an accompanying data acquisition procedure was developed that describes the steps taken during a recording session. Within this section, first the demography of the participants is described in section 2.1.1, consecutively the acquisition procedure and the physical recording setup are described in detail in sections 2.1.4 and 2.1.3. The major administrative aspects of the data are described in section 2.1.2. Finally, section 2.1.5 lists the common data processing steps which were applied on the raw recorded data.

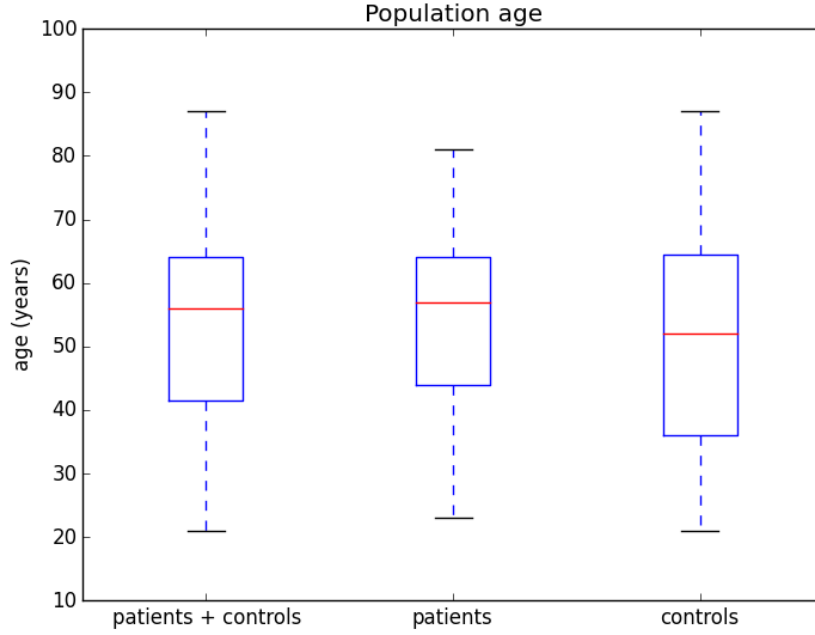
#### 2.1.1 Participant demography

For this study 91 participants have been recruited. The participants consist of a FSHD patient group of 56 FSHD patients (24 females) and a control group of 35 healthy controls (19 females). The FSHD patients who participated in the study were all patients visiting the outpatient clinic at the department of Neurology at the Radboud University Medical Center Nijmegen. In case a patient brought another person with them, he or she would be invited to also take part in the study within the healthy control group. If a person accepted to participate in the study, he or she would sign a consent form in which the person agreed to be filmed for the purpose of scientific research. The consent form can be found in Appendix A. Four participants within the FSHD patient group have been dropped from the study, because their DNA test results were not decisive about the presence of FSHD.

The participants within the patient group range between the ages of 23 and 81 years and for the control group they range between the ages of 21 and 87 years. The age distributions have been summarized in Figure 2.1.

#### 2.1.2 Data and task description

Both patients and healthy controls were video recorded while performing seven different facial tasks in randomized order: closing the eyes gently, closing the eyes firmly, raising the eyebrows, frowning, pursing the lips, showing the teeth and puffing of the cheeks. The tasks are commonly used for measuring facial weakness and have an emphasis on the affected facial muscles within FSHD (see section 1.1 for details). All tasks were characterized by a participant moving from a “rest condition” in which all facial muscles were relaxed to an “active condition” where the

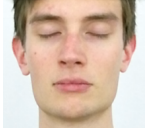



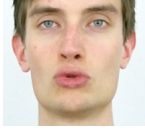




**Figure 2.1:** Shows population age distributions for the patient and control groups and combined groups.

participant maximally exerted the facial muscles in order to perform the task optimally. Each of the tasks was assigned a specific identification number and a specific tone. The tones were assigned for synchronization of the videos and are identified by their frequency, which could range from 140 Hertz to 440 Hertz. Table 2.1 shows the tasks, with their assigned identification numbers and tone frequencies. During the session, each task was repeated three times in succession, resulting in  $7 \times 3 = 21$  tasks per participant to account for intra person variability per task.

The videos were recorded with four cameras: a Microsoft Kinect 2 sensor and three Canon PowerShot cameras (one PowerShot HS280 camera and two PowerShot HS270 cameras). The Kinect 2 sensor and the (black) PowerShot HS280 were positioned in front of the participant, a gray PowerShot HS270 was positioned to the left of the participant and a blue PowerShot HS270 was positioned to the right of the participant. Figure 2.2 shows the cameras and their position in space. The following settings were used for the cameras:

- Raw infrared information, audio and uncompressed color streams were recorded from the Kinect 2 sensor. The Kinect settings for the streams are fixed and are by default: a resolution of  $1920 \times 1680$  pixels for the color stream and a resolution of  $512 \times 424$  pixels for the depth stream with 11 bits used for depth per pixel. Both images are sampled at 30 frames per second. The audio is recorded with four microphones that are equally distributed over the width of the device as a microphone array and are sampled at 48 kHz each.
- All PowerShot cameras were set to record video at a resolution of  $1920 \times 1680$  at 60 frames per second.

task id	task name	tone frequency (Hertz)	active condition
0	closing the eyes gently	140	
1	closing the eyes firmly	190	
2	raising the eyebrows	240	
3	frowning	290	
4	pursing the lips	340	
5	showing the teeth	390	
6	puffing of the cheeks	440	

**Table 2.1:** Identification number, name, assigned tone frequencies and exemplar image at the active condition for each of the 7 tasks.

### Video synchronization

To be able to make a computer automatically obtain the video materials corresponding to certain tasks, the videos needed to be synchronized with each other. This was accomplished by using task related auditory tones. The EXPERIMENTER TOOL would play an audio tone for half a second after the experimenter started any task. Table 2.1 shows frequencies for each of the 7 tasks, which characterize a distinguishable computer generated tone unique to each task. The tone would be audio recorded by all the 4 cameras, while simultaneously recording video. The recorded audio cues were used during data processing (see section 2.1.5) to automatically retrieve the timing of the tasks and to cut the videos accordingly. The tones were chosen for easy recognition by their frequency magnitudes using time-frequency analysis<sup>1</sup> and for being sufficiently distinguishable from human speech signals.

An external computer speaker was used for emitting the tones and was positioned on the floor

<sup>1</sup>see Cohen [51] for a review on time-frequency analysis.



**Figure 2.2:** The cameras used and their positioning. (left) The left gray HS270 Powershot camera. (center) The Kinect 2 sensor and the higher positioned black frontal HS280 Powershot camera. (right) The positioning of the cameras around a stool for participants to sit on.

under the two frontal cameras. This position was chosen to minimize the effects of differences in distance between speakers and the audio source. Hence it was positioned this way to avoid possible timing differences of the auditory and video signals for a recording device. The distances between the audio source and the recording devices were kept within the 3 meter range. Given that speed travels at approximately 340 meters per second and the highest recording speed was 60 frames per second, the limit was  $340/60 \approx 5.67$  meters per frame.

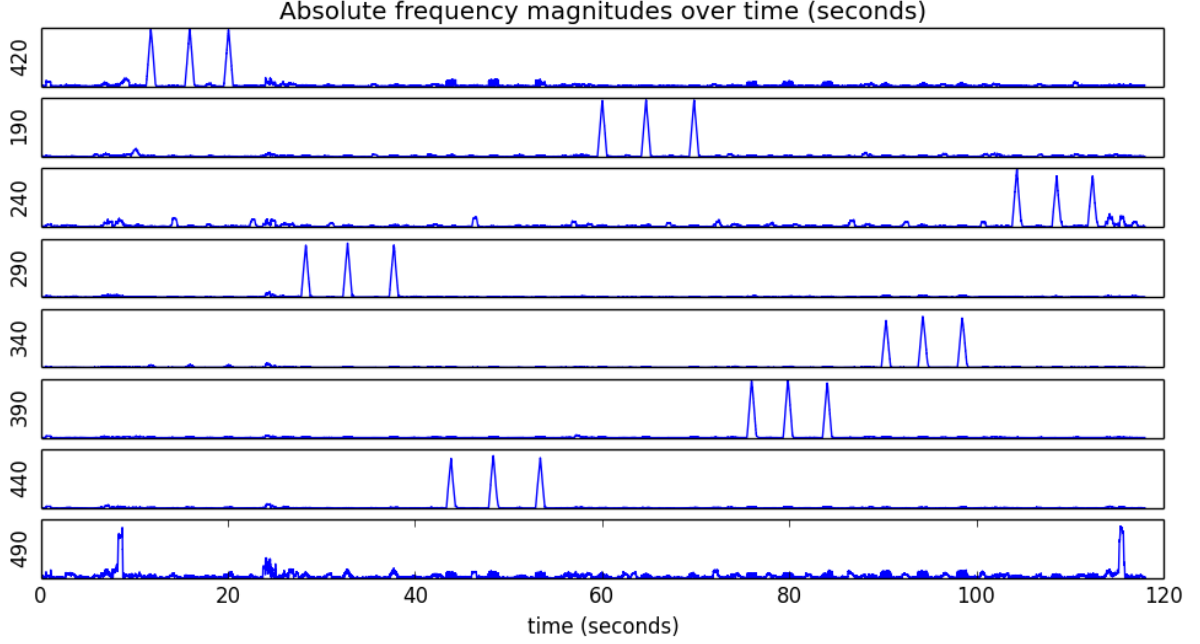
The EXPERIMENTER TOOL kept track of the time at which a task was cued by the experimenter and would store it in the participant log. The stored information for each task (and iteration) was the time difference between the beginning of the session and the cue onset of a task. This cue timing information was used for retrieving the ordering of tasks during data processing and also helped to resolve recordings where the audio signals were distorted.

**Time-frequency representation** First the auditory signals were extracted from the videos using the DATA TOOL. The raw audio was recorded at 48000 Hertz, which implies 48000 amplitude samples per second. To speed up computations, only every 40th sample was used, thus the sample rate became  $r = 48000/40 = 1200$  samples per second. Consecutively, a time-frequency representation was calculated, which is used to analyze the signal in both time and frequency domains simultaneously and describes the frequency magnitude over time. The representation was calculated for the reduced signal using a discrete Short-Time Fourier Transform (STFT) with a window size of 600 samples (corresponding to the duration of a cue) for a set of preselected frequencies  $P_{freq} = (420, 190, 240, 290, 340, 390, 440)$ . Finally, for all the calculated magnitudes the absolute value was taken to enable cue detection by taking the maximum. For this section, the absolute frequency magnitude over time for a single frequency is represented as a vector  $\mathbf{m} = (m_0, m_1, \dots, m_{n-1})$ , where  $n$  is the discrete number of samples taken which corresponds to the total number of samples taken from the audio stream of the video. Then the absolute time-frequency representation for the eight frequencies from  $P_{freq}$  can be represented as a matrix  $\mathbf{F} = (\mathbf{m}_0, \mathbf{m}_1, \dots, \mathbf{m}_7)^T$ .

The first seven preselected frequencies from  $P_{freq}$  relate directly to the 7 tasks specified in Table 2.1. However, the first frequency was picked to be 420 Hertz, which is different than the tone frequency for the first task. It was found that a frequency of 420 Hertz gave a better magnitude signal over time compared to the original tone frequency of 140 Hertz. This is probably due to acoustic resonance, since the former frequency is a multiple of the latter. Another explanation for the phenomena is that the generated tones were not perfect sinusoids, but were cubic in form. Therefore, smaller wave forms might be better suited to pick up the signal. The last element of the preselected frequencies  $P_{freq}$  of 490 Hertz was added as an additional tone for a task that involved moving the arms and shoulders. However, because the



task did not involve facial features and it made the acquisition procedure more time consuming, it was later dropped. Figure 2.3 shows the absolute magnitude for a sound frequency (Hertz) over time (seconds) for each of the preselected frequencies  $P_{freq}$ , which were obtained for the absolute time-frequency representation  $\mathbf{F}$ . The peaks in magnitude indicate the position of the task cues in time for the video and can be discriminated based on frequency.



**Figure 2.3:** The frequency magnitudes over time for multiple frequencies in a video file of approximately 2 minutes. The y-axis labels indicate the frequencies in Hertz corresponding to each tone frequency. The peaks indicate the position of the cues in time in the video.

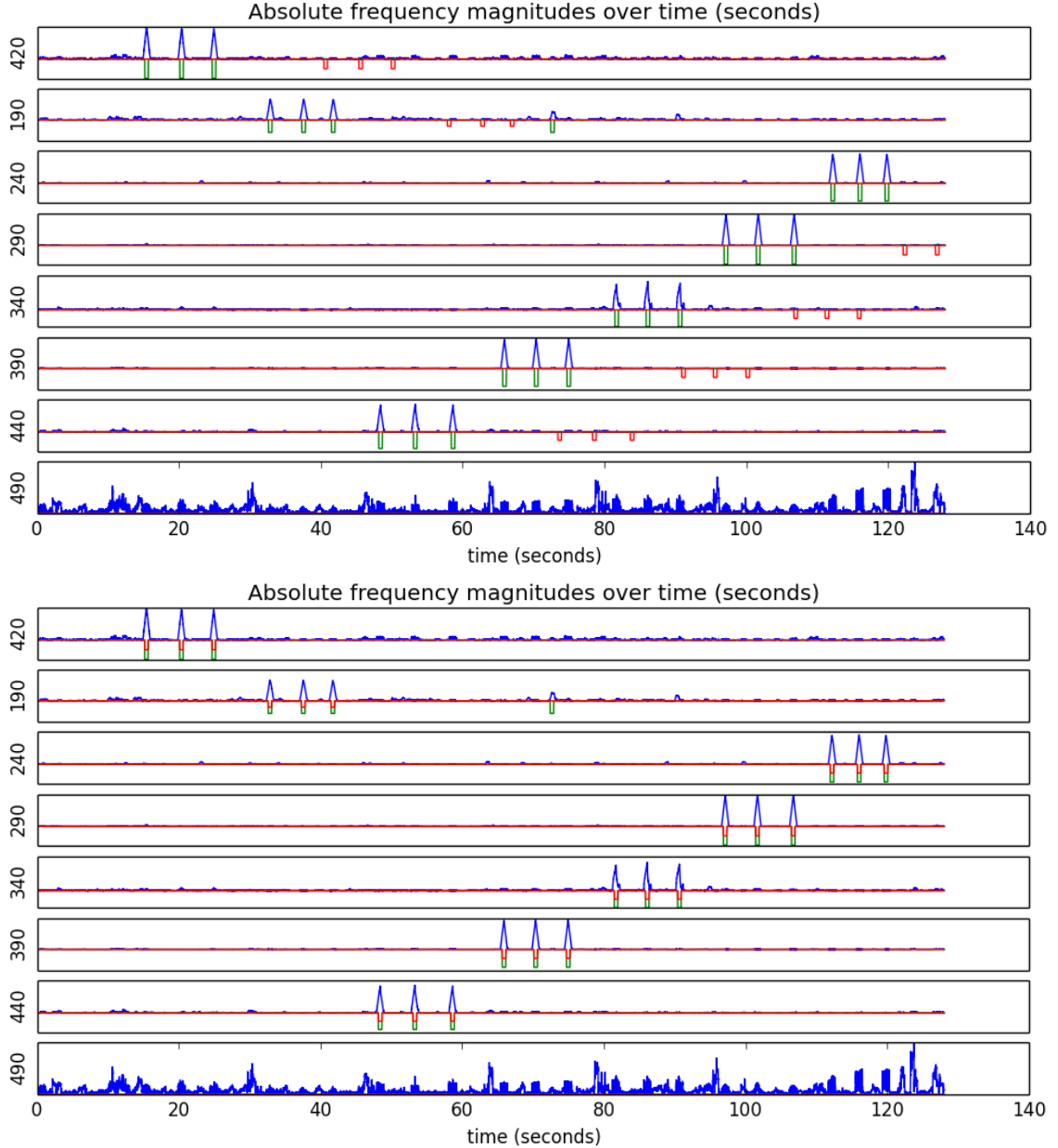
**Cue detection** Having matrix  $\mathbf{F}$  with the absolute frequency magnitudes over time makes extracting the timing  $t$  and related task identification number  $i$  for each cue  $c = (t, i)$  straight forward: first a threshold  $k$  was defined for the magnitudes for each frequency. This was set to a magnitude of  $k = 350000$  for all the PowerShot videos and to a magnitude of  $k = 40000$  for the Kinect recordings. Then the following algorithm would extract a set with all the cues  $C$  with timing information in seconds for a video:

```

Input: A matrix  $\mathbf{F}$  with the absolute magnitudes over time for 8 frequencies and a
          threshold  $k$  and a sample rate  $r$ 
Output: A set of cues  $C$ , where each cue  $c = (t, i)$  has a time offset  $t \in \mathbb{R}$  in seconds
          and a task identification number  $i \in (0, 1, \dots, 7)$ 
for  $0 \leq i < 8$  do
     $\mathbf{m} \leftarrow \mathbf{F}_i$  ;
     $\text{argmax} \leftarrow \text{index where } m_{\text{index}} \in \mathbf{m} \text{ and } \forall s \in \mathbf{m}, t \geq s$  ;
    while  $m_{\text{argmax}} > k$  do
       $C \cup (\text{argmax}/r, i)$ ;
      for  $\text{argmax} - 300 < j < \text{argmax} + 300$  do
         $m_j \leftarrow 0$  ;
return  $C$ 

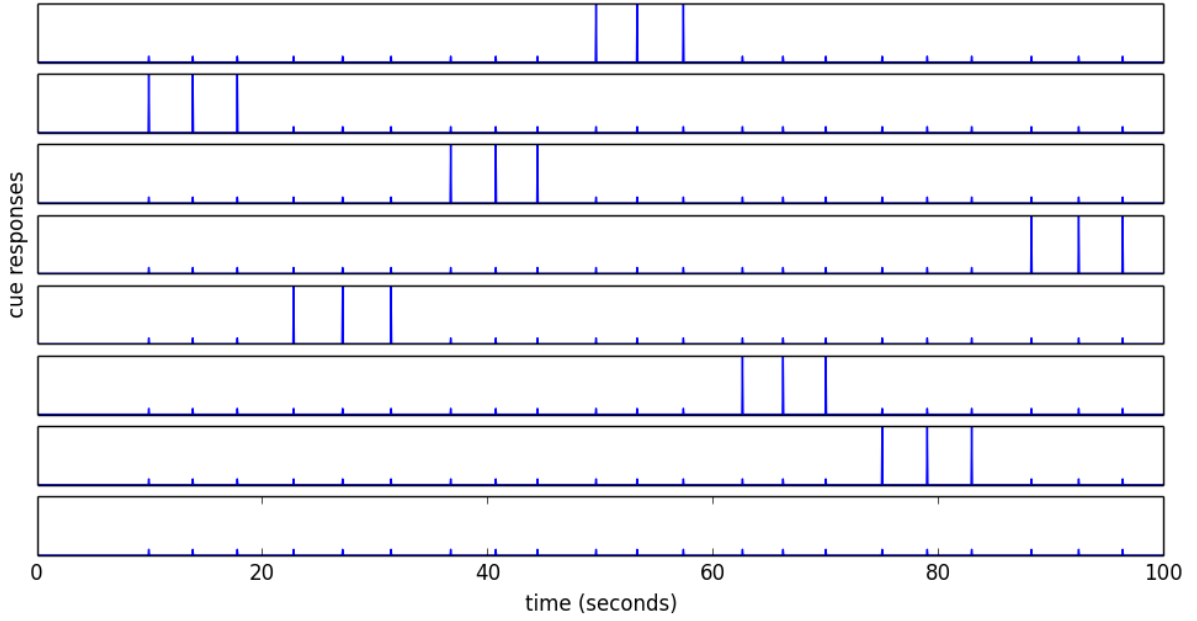
```

**Cue synchronization** Cue synchronization is the process of correctly aligning a set of cues to the right place in the audio stream of the video files, in other words finding the optimal displacement in time to fit the cues to the target audio stream as shown in Figure 2.4. Two methods were developed to accomplish this, both methods use a similar principle, but the first method aligns a set of cues to a set of cues extracted from an audio stream and the second method aligns the cues directly to the frequency magnitudes of the time-frequency representation  $\mathbf{F}$ . The first method is computationally more efficient than the second, but requires the cues to be extracted from the audio signal.



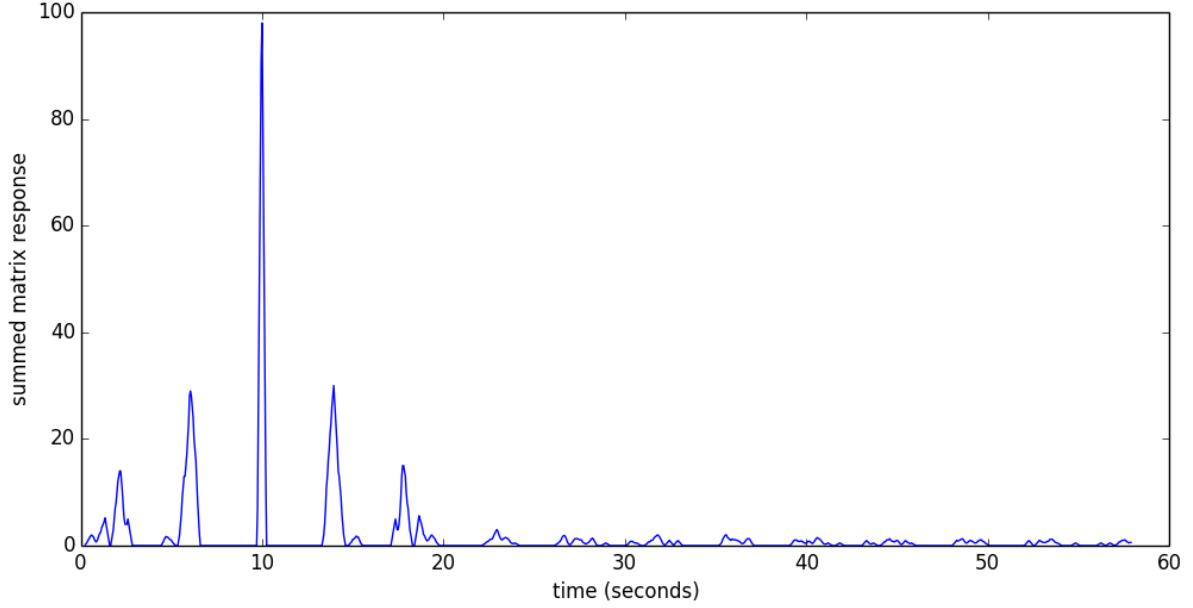
**Figure 2.4:** The frequency magnitudes over time for multiple frequencies for a video file. The y-axis labels indicate the frequencies in Hertz corresponding to each tone frequency. The red markers represent the participant log cues and the green markers represent the extracted cues from the time-frequency representation of the auditory signal. The top Figure show the positions of the cues before fitting and the Figure at the bottom shows the positions of the cues after fitting.

Both aligning procedures first convert the input cue set into a  $7 \times n$  matrix  $\mathbf{C}$  and optionally converts the cue set for the audio stream into a similar  $7 \times m$  matrix  $\mathbf{R}$ , where  $n$  and  $m$  are chosen in such a way, that the sampling rate is the same for both matrices. If the time-frequency representation is used,  $\mathbf{R}$  is set to  $\mathbf{F}$  and the number of samples  $n$  is adjusted according to the sampling rate of the time frequency representation. Figure 2.5 shows an example discretization for a set of cues. A cue signal is set to a high response value of 1.0 for each channel that corresponds to the task identification number of the cue at the exact onset time. For each high response in time, low responses with a value of 0.1 were also added for the same time on the other channels to have a higher probability of finding the right cue timing pattern in the rare case that the whole order was given using mismatching audio cues.



**Figure 2.5:** A discretization of a set of cues showing all cue responses over time for each task frequency. The high responses have a value of 1.0 and the lower values have a value of 0.1.

After discretization of the inputs, the problem comes down to maximizing the best fit between matrices  $\mathbf{R}$  and  $\mathbf{C}$ . For the following approach it is assumed that the  $m > n$ , since generally the cues to be matched are encompassed by the target audio stream. Subsequently, by taking each possible submatrix of length  $n$  from each  $\mathbf{R}$ , multiplying all those matrices with  $\mathbf{C}$  and taking their summed responses, results in a vector  $\mathbf{s}$  of  $m - n$  values over time. Taking the index at the maximum of  $\mathbf{s}$  gives the offset for the optimal alignment, which can be translated back to the offset in seconds. Figure 2.6 shows an example of the summed cue matrix responses over time. The maximum and hence the required aligning offset lies at the peak.



**Figure 2.6:** Summed cue matrix responses over time.

**Cue validation and correction** The previously described cue extraction procedure was able to detect most of the cues from the audio signals of the videos, but sometimes cues were missed or noise in the signals would generate false positives. Additionally, participants would sometimes perform a task in a wrong way, what could cause an experimenter to cue the task again or the experimenter accidentally made a mistake during a recording session.

To identify all the correct cue positions and remove noise and redundant cues, the participant logs were used. Participant log cues were derived from the cue onset timing, relevance factors and task durations from the timing and task information from the participant logs. The participant log cues were a list of cue information with various properties. Table 2.2 lists the properties for each participant log cue. The relevance factor could be set to indicate if the cue should be cut in all the recordings or not. The time offset could be set to translate a single cue relative to their onset timestamp at the time of cutting of the movies.

Property	Unit	Description
Onset timestamp	seconds	Difference between session start and cue onset
Task identification number	integer value in the interval $[0, 7]$	Task identification number
Task duration	seconds	The task duration was determined by the timing difference $dt$ (in seconds) between this cue and the next. Task duration was $duration = \min(dt, 5)$
Relevance factor	real value in the interval $[0, 1]$	1 represents relevant and 0 represents irrelevant
Time offset	seconds	Manual offset for small cut corrections. Default was 0 seconds

**Table 2.2:** Properties of a participant log cue.

The cues were validated by manually inspecting the participant log cues and their synchronization to the video files through the CUE TOOL. In general, the cues and auditory signals were already extracted from the recordings, subsequently the CUE TOOL could visualize the audio signals with the recognized cues and the participant log cues over-layed. Figure 2.4 shows two such visualizations, where at the top the participant log cues have not yet been synchronized with the cues from the audio signal.

The CUE TOOL provided methods to manually correct the participant log cues and the cue alignment. For aligning the cues the synchronization methods described in section 2.1.2 could be applied. Usually the cue to cue matching was sufficient, but in particular for the Kinect recordings, the cues could sometimes not be correctly obtained, resulting in an empty cue list. In that case the other method could be used. The participant log cues were stored in a JSON encoded list and could be edited using a text editor in order to modify any of the properties. The most frequently used operation was to change the relevance factors to include or exclude particular cues from being used to cut the task videos. For each task three cues had to be selected and in some cases there were more than three cues for a task. In that case the frontal video for the participant was watched to identify which cues to select.

### **Administrative aspects**

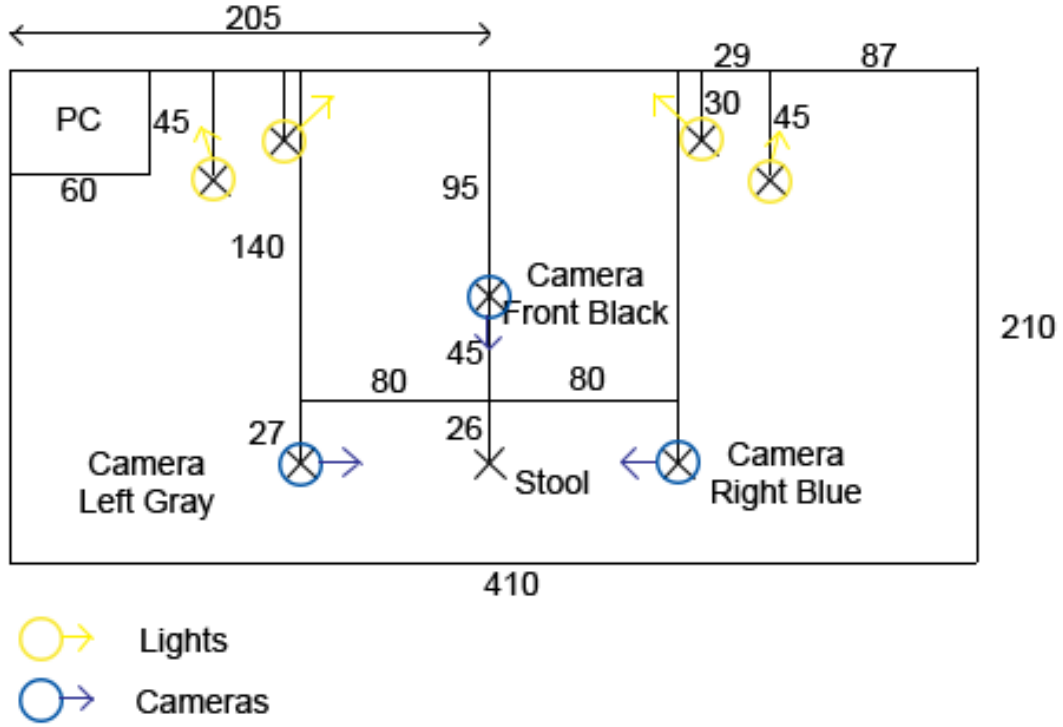
After the data were recorded, the video footage for all four cameras would be gathered on an encrypted external data disk using Samsung SecretZone disk encryption to ensure patient confidentiality by denying unauthorized access. Two additional backup disks were used to store copies of the data. For each participant the following additional information was stored in a participant log:

1. A personal identification tag for the participant
2. Whether the participant is a control or a patient (control / patient)
3. The age of the participant
4. The gender of the participant (male / female)
5. Recorded session timing information

The personal identification tag could be used by the principle investigators to track back the identities of the participants, but not by anyone else. Furthermore, the identification tag was also used for labeling the video files. Information like names and other medical details were not included in the participant-database. The recorded session timing information was used for obtaining the video fragments belonging to each task.

### **2.1.3 Acquisition setup**

A small area (2.10 by 4.00 meters) inside the Radboud University Medical Center Nijmegen was reserved to stall equipment for the duration of the study. The reserved area was partitioned from the rest of the room by hanging white curtains from the ceiling. In the center of the area a height adjustable stool was positioned where a participant could sit. Four cameras were mounted on tripods and arranged around the stool for simultaneously recording the front and sides of a single person. The lighting within the small area was accomplished using two Falcon Eyes LHK-240 Lighting sets that each contained two lamps, that were directed at the walls to create a soft diffuse indirect illumination on the subject. These lighting sets were the only



**Figure 2.7:** Overview of the acquisition area including the physical distances between the different pieces of equipment. The aiming direction for the lights and cameras are depicted with a yellow and a blue arrow respectively. The wall is the top line in the image.

enabled lights during recording. A computer was positioned in a corner of the area and was connected to the Kinect 2 sensor and to a small mono audio speaker. The computer was used by the experimenter to direct the acquisition procedure and to record the video streams from the Kinect 2 sensor.

Figure 2.7 shows a schematic overview of the acquisition setup inside the room and shows the exact positioning used for the cameras, lighting equipment, stool, audio speaker and computer. At the bottom center of the setup there was the height adjustable stool for participants to sit on, which could be adjusted in height. The Kinect 2 depth camera, a black colored PowerShot HS280 and an mono audio speaker were positioned on tripods 71 cm in front of the stool and 95 cm from the center of the wall. A gray and a blue PowerShot HS270 camera were positioned respectively left and right from the stool at an 80 cm distance and were both positioned 167 cm parallel to the wall. All PowerShot cameras were mounted on a height of 118 cm measured from the floor to the center of the lenses. The Kinect camera was positioned at 98 centimeters from the floor to the center of the device. All cameras were mounted vertically (in stead of the standard horizontal way) to be able to record as much of the body of the participant as possible. The two lighting sets were positioned out of sight of the cameras. Each set had two lamps which were positioned at 87 and 116 centimeters from left and right sides of the area and 30 and 45 centimeters from the side at the top. The computer was positioned in the left corner of the area on a  $60 \times 45 \times 61$  tray at the upper left corner.

## Computer specifications and software

The computer was used by the experiment instructors for cuing tasks in succession during a recording session by using specifically developed EXPERIMENTER TOOL, and for recording video data with the Kinect 2 sensor using Microsoft Kinect Studio. The EXPERIMENTER TOOL would trigger a uniquely identifiable auditory tone when the instructor triggered the next task of the 7 tasks, which would be emitted by the mono audio speaker. The video materials could later be automatically time-synchronized based on the recorded auditory tones and the different tasks could be automatically identified by the computer. The starting times of the tasks would also be logged by the EXPERIMENTER TOOL and would later be stored. The computer had an Intel Core i7-3770 3.40GHz CPU with 16 GB RAM, a NVIDIA NVS 315 video card and had Windows 8.1 Pro installed.

### 2.1.4 Acquisition procedure

This section describes the full video recording procedure and explains how an instructor would video record a participant and what instructions were followed. The job of the instructor was to guide the recording process and to ensure that the participant created valid data. Each recording session took approximately three minutes to complete.

For cuing the participant to perform a certain task, the EXPERIMENTER TOOL was used, which showed the instructor the randomized order of the tasks for the participant. First the instructor would tell the participant the upcoming task. Next the instructor would trigger an auditory cue (belonging to that task) through the EXPERIMENTER TOOL after which the participant would perform the task. After the participant finished performing the task, the last two steps would be repeated until at least 3 valid recordings were acquired, i.e. additional recordings were made when a participant laughed during a task or performed the wrong task.

The instructor ensured that the following preparation conditions were fulfilled before video recording a participant:

- All cameras and sensors were mounted at the correct positions
- Sufficient space was available on the camera memory cards and on the computer hard drive for recording the videos
- The Kinect 2 sensor was properly connected to the computer
- The computer was turned on and running the EXPERIMENTER TOOL and Microsoft Kinect Studio software
- All lights were off except the lighting sets
- All cameras were not yet recording

Next, the instructor would invite a participant to sit on the stool. The stool would be adjusted if the participant was particularly long or small to ensure that the face would be at the same level as the PowerShot cameras. In case the participant used a wheelchair he or she was asked to try and sit on the stool for the duration of the session, which was in general not longer than 3 minutes. If the participant did not want or was unable to do so then the participant was video recorded in the wheelchair.

The instructor would then explain the acquisition procedure and demonstrate all of the tasks to the participant. Most of the tasks were pretty straightforward and easy to explain. However, some participants did not know very well how to frown the eyebrows, in such cases

the participants were instructed to make a face as if they were angry. For the closing the eyes gently task, the participant was asked to close the eyes in a relaxed manner. For pursing the lips, the instruction was given to shape the mouth like giving a kiss. The participant was instructed to keep his or her feet on the floor and to look at the lens of the black PowerShot camera while performing the tasks. To ensure that the participant understood all instructions a trail task would be performed, i.e. to cue the participant with the auditory tones and letting the participant perform a particular task three times in succession.

Before recording, the participant was instructed to remove any glasses that he or she was wearing and to move hair that covered parts of the face (e.g. eyebrows or eyes) out of view for the duration of the session. Then all cameras were put in recording mode and the instructor would sit in front of the computer and start triggering the auditory tones for all of the  $7 \times 3$  tasks until done. After the recordings, the cameras would be stopped and the participant would be thanked for his time.

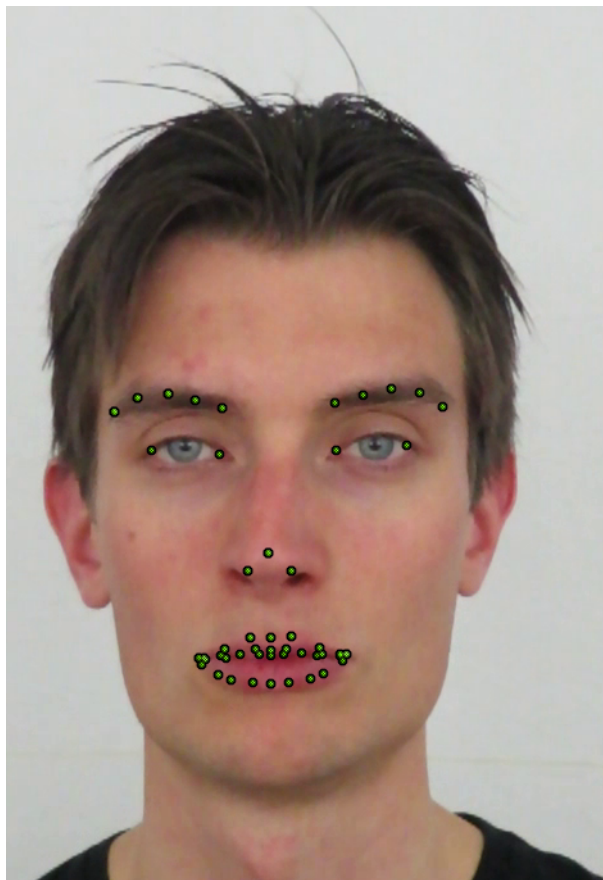
### 2.1.5 Data processing

The following set of operations was performed on the data before the data were used for the analyses. The data initially consists of text files with the participant logs (.txt), the raw video data from the Kinect 2 (.xrf), and the raw data from the three PowerShot cameras (.mp4). At the end of the processing procedure, there are 3 iterations of cut video recordings for each task per participant.

1. For the first step, necessary information was extracted from the participant logs using the PARTICIPANT INDEXER utility. First of all, a list with the personal identification tag, age, gender and the time of log creation was extracted for each participant and the combined information was stored in one file which was ordered on log creation time. This file served as an index for other utilities to look up the available participant tags and occasionally to look up other information. The recorded session timing information was not stored in the index file, but was stored for each participant in a separate file.
2. The following step was to associate the still unlabeled video files with their corresponding participant tag. This was convenient to be able to quickly associate a video file to the corresponding participant by its tag. For this purpose, the DATA TAGGER utility was developed, which aided in “tagging” all unlabeled video data. The utility could be used to first validate if the video data would be correctly tagged before actually renaming it, and was useful for identifying missing or redundant video recordings.
3. The DATA TOOL was used to automatically extract, analyze and find the tone onset locations for all audio streams within the tagged videos. Both the onset locations and the audio streams were saved. Subsequently, the CUE TOOL was used to validate the correctness of all onset locations. The CUE TOOL was developed for visualizing the audio streams with the identified onset locations overlayed. Furthermore, the tool could be used to edit the onset locations as well as the cut duration. Once a user was satisfied with the onset locations and the cut durations, they were stored as a final version.
4. Using the final onset locations and durations from the previous step, the video files were automatically cut for all seven tasks and three iterations using the DATA TOOL cut option. The duration of an action was generally no more than 5 seconds. This resulted in cut videos for each participant for each task an each iteration. The cut videos were tagged with the participant tag, task number and iteration number for easy identification.



## 2.2 Landmark detection pilot



**Figure 2.8:** The 49 FaceReader landmarks (green dots).

Using the acquired data, the initial approach was to use FaceReader 5.0 [28] (Noldus, Wageningen The Netherlands) for 2D automated landmark detection, which is commercial landmark detection and tracking software for the automated detection of facial expressions and emotions. The software uses an appearance-based approach (i.e. based on a model learned from data) and is trained on more than 10000 manually annotated images for training the software. It is claimed to generate objective observations. The software has various built-in modules for tracking emotion and landmarks over time. Figure 2.8 shows the predefined set of 49 landmarks for the brows (10), eyes (4), nose (3) and mouth (22).

The main reasons for using FaceReader were the availability of the software within the department and the reported reliable objective tracking. The cut movies for each task and iteration ( $7 \times 3 = 21$ ) for the frontal PowerShot videos for 93 subjects were entered into the software together with their sex, age and participant tag by using an automation script written in Java, and were subsequently analyzed and processed by the software as a batch process. Finally, the landmarks and other extracted data over time for the movies were written to log files for further processing outside of FaceReader. The automation script was created since manually adding the movies was very time consuming and FaceReader 5.0 has no method for programmatically adding movies for analysis.

However, despite following the instructions for proper acquisition within the manual concerning the lighting, image quality and capturing conditions, it became quickly apparent that

the software was not able to reliably track the faces. First of all, in many cases the software was periodically not able to localize the face and the landmarks and would simply yield an error and return no data for those frames in time. Secondly, even when the face was tracked, it could not accurately capture asymmetries within the face and was in some cases not able to detect opening or closing of the mouth. Thirdly, tracking of the eyes was done with only 2 markers for each eye and a discrete variable for being open or closed. However, upon closer examination for the tasks involving the eye, this state was frequently missed, even when the eyes were closed.

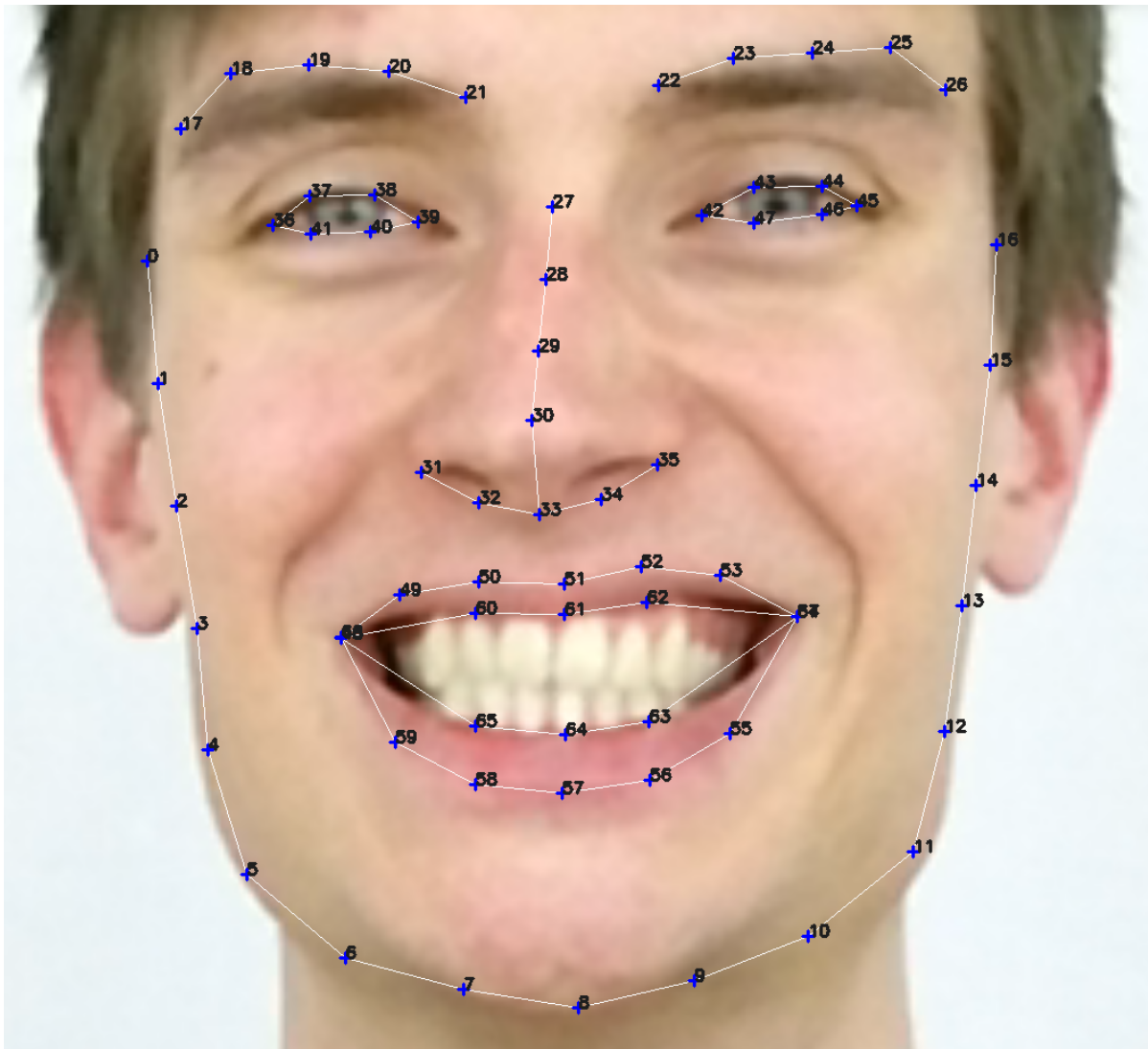
Due to these findings it is clear that FaceReader 5.0 is not suited for the objective grading for facial weakness, while still being useful for other things like facial expression analysis. Therefore, the decision was made to focus more on the extraction and validation of relevant features for training an objective grading system for facial weakness in FSHD patients and leave the automated landmark detection for future work.

## 2.3 Manual labeling

Because of the findings from section 2.2 that automated landmark tracking using FaceReader was not sufficiently reliable, the choice was made to manually label two extracted images from the videos with facial landmarks for the active and rest positions. While the choice for manual labeling does break automation and adds some subjective bias by the labeler, it was thought to help more accurately identify the facial features that are relevant to facial weakness for FSHD, independent of the performance of a marker tracking system. Once it is verified that the used landmarks and the related extracted features are truly relevant for assigning facial grades, an automated tracking system for those landmarks could be developed.

The chosen landmarks were based on the areas known to be most involved with FSHD, in particular the areas around the eyes (*orbicularis oculi*) and around the mouth (*orbicularis oris*), but also the eyebrows were marked. The nose and face contour landmarks were added to provide an additional reference frame. Figure 2.9 shows the used landmarks for manual labeling. Each visible eyebrow was given 5 landmarks, which were positioned just above the visible eyebrow hair, closely following the contour of the brows. Markers 19 and 24 were at the center of the brows, markers 17 and 25 were at the end of the brow farthest away from the nose and markers 21 and 22 were at the end of the brow closest to the nose. For each eye 6 markers were used: two for each eye corner (inner canthus and outer canthus), two for the bottom of the eyelid contour and two for the top of the eyelid contour. If the eyelashes covered the bottom eyelid contour when closing the eyes, the bottom markers were placed just below the eyelashes. For the 9 nose markers, 4 markers were placed on the nose and 5 markers were placed just below the nose. Marker 33 was placed below the columella just above the philtrum, marker 27 was placed at the nasion, marker 30 was on the supratip of the nose and markers 31 and 35 were placed at the sides of the nose. 20 markers were used for the mouth, which mainly encompass the contours of the upper and underlip. Markers 48 – 54 range from the left to the right oral commissure along the upper lip contour (vermillion border), markers 54 – 59 together with marker 48 define the lower lip contour along the vermillion border, markers 48, 60 – 62 and 54 define the under upper lip contour and markers 54, 63 – 65, 48 indicate the upper under lip contour. Marker 51 is positioned at Cupids’s bow. Markers 66 and 67 were used to indicate a trailing fold in the skin at the oral commissures if it was present. If it was not present, the markers were positioned at the same position as respectively marker 48 and 54. For Figure 2.9 the markers were not positioned on top of the other markers to indicate their existence. Finally the head contour used 17 markers ranging from 0 – 16. Here marker 8 was positioned at the center of the chin, markers 0 and 16 were positioned just above the ears and markers 3 and 13

were positioned just below the ears.



**Figure 2.9:** The 68 facial markers with their identification numbers. The marker numbers are grouped into the following facial regions: face outline 0 – 16, left eyebrow 17 – 21, right eyebrow 22 – 26, left eye 36 – 41, right eye 42 – 47, nose 27 – 35 and mouth 48 – 67.

### 2.3.1 Labeling procedure

After the processing steps from section 2.1.5, manual labeling was achieved by performing the steps listed below on the data. Two software tools were specifically written for this purpose: the VIDEO POSITION MARKER for marking the repose and active conditions for each task video and the FACE MARKER tool which was used for manually placing the landmarks on the faces. The reader is directed towards the appendix D.3 for a detailed description of the tools.

1. Two time positions within the videos for a rest condition and an active condition were manually determined for each task video for each participant using the VIDEO POSITION MARKER utility. In the rest condition the participant would relax all facial muscles and in the active condition the participant would maximally exert the muscles. Images for

both conditions were labeled for both the Kinect recordings and the frontal videos.

2. The images for the rest and active conditions were extracted for the identified time positions for both the Kinect recordings and the frontal videos for each of the seven tasks and three iterations.
3. DRMF face fitting software [52] was used to obtain a rough 66-point face landmark estimation for the extracted images.
4. For each participant a rectangle called the crop box was defined by hand, using the overview mode of the FACE MARKER tool. The crop box was used in the interface to segment the head and face from the rest of the image in order to place the markers more accurately.
5. For each participant the pre-fitted 68 facial landmarks were corrected by hand using the face mode of the FACE MARKER tool for each of the two extracted state images and for each of the seven tasks, according to the labeling approach described in 2.3. This last step has currently only been performed on the first iteration on all Kinect 2 extracted color images for 91 participants, due to its time consuming nature.

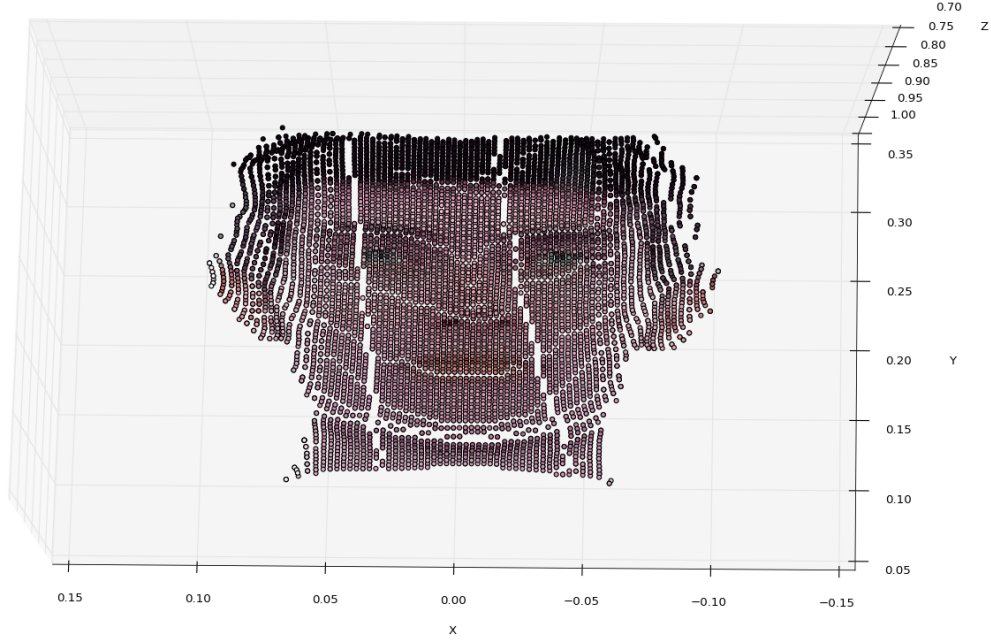
## 2.4 Image processing

### 2.4.1 Kinect 2 depth information

Each annotated landmark within the Kinect 2 color images corresponds to a specific real world 3D  $(x, y, z)$  geometric location represented in meters, which can be extracted from the Kinect 2 recordings. When using both the 3D and 2D coordinates for each of the 68 landmarks for calculating coordinate based features, it is possible to compare the two modalities and to assess the value of 3D landmark information over 2D landmark information. Most of the extracted features from section 2.5 are hence tailored to work both for 2D and 3D landmarks.

The metrics for the extracted 3D landmark coordinates relate to real world meters and can be used to yield objective metric distance information, but it should be noted that the reliability of these measures depends on the accuracy and precision of the Kinect 2 device. These have for as far as the author knows not yet been verified and compared for the Kinect 2, but for the previous version of the device there has been an actual metrological evaluation [53], which reported the device to be valid for low range mid-accuracy applications with accuracy of maximum of 1.5 cm error within the 1 meter range. It is, however, to be expected that the Kinect 2 has better accuracy and precision due to a different depth acquisition method (time-of-flight as opposed to structured light projection), which has a higher spatial resolution, i.e.  $512 \times 424$  depth pixels as compared to  $320 \times 240$  depth pixels and is less susceptible to shadows.

For extracting the depthdata, the KINECT IMAGEGRABBER tool was used, which would be given a cut Kinect 2 recording for a certain participant together with a time offset for either a face at rest or at maximal exertion. The tool would then use the CoordinateMapper from the Kinect SDK to map the 2D image coordinates to their respective 3D coordinates and extract those as a pointcloud. A pointcloud is a simple list of  $(x, y, z)$  coordinates called voxels describing 3D geometry. Because a pointcloud file can quickly become large in size, only all voxels within the cropbox of the participant were extracted, which limits the pointcloud to only the relevant voxels of the head. In addition to the extracted 3D points, the exported file also contained the cropbox dimensions, to later easily retrieve the mapping within the original



**Figure 2.10:** A pointcloud visualization of the Kinect 2 3D depth information, where each point has been mapped to its respective color from the color stream.

image. Figure 2.10 shows a 3D reconstruction from such an exported pointcloud file. Each color pixel within the selected face image cropbox is plotted against their respective 3D point. For the visualization all voxels (3D pixels) more than 1 meter away from the Kinect sensor have been removed.

The stored 3D pointcloud information can be used as a lookup table to map the manually labeled markers (2D positions) to their corresponding positions in 3D space. Figure 2.11 (left) shows the direct mapping of facial landmarks (red dots) within the entire pointcloud of the face. Note however that for this naive mapping the manually placed 2D markers do not necessarily align with the facial jaw contour in 3D, i.e. some of the markers for the chin fall in 3D on the neck (resulting in a higher  $z$  value) and some contour markers are missed (missed values are assigned the coordinate (99,99,99)), because they fall outside of the face. If these cases are not dealt with, they can cause outliers to appear when calculating features utilizing these coordinates, which is undesirable since it could potentially render 3D contour related features to become useless. To prevent such outcomes, a simple contour correction routine was applied on all 3D contour points before feature calculation.

### Contour correction

The contour correction approach moves each contour landmark  $\mathbf{p}_{x,y}$  (landmarks 0 until and including 16) from the original 2D color image in a straight line towards the nose tip  $\mathbf{n}_{x,y}$  (landmark 30), one pixel at a time per step  $i$  over distance  $D$  between both points, resulting in the new position  $\mathbf{p}_i$ :

$$D = \|\mathbf{p} - \mathbf{n}\|$$

$$\mathbf{p}_i = \left( \frac{\mathbf{n} - \mathbf{p}}{D} \right) i + \mathbf{p}$$

Where the optimal number of steps  $i^*$  is determined by minimizing the following error function  $E$ :

$$E = (\alpha i)^2 + \sum_{j=1}^3 (d_j w_j)^2$$

$$i^* = \arg \min_i E$$

where  $\alpha$  and vector  $\mathbf{w} = (w_1, w_2, w_3)$  are tuning parameters. The absolute distances for each axis vector  $\mathbf{d} = (d_1, d_2, d_3)$  are calculated using the 2D to 3D lookup using a pointcloud and the following transformation  $PC : \mathbb{N}^2 \rightarrow \mathbb{R}^3$ , 2D landmark position  $\mathbf{p}_i$  and the nose landmark position  $\mathbf{n}$ :

$$\mathbf{d} = |PC(\mathbf{p}_i) - PC(\mathbf{n})|$$

The intuition behind the contour correction is to penalize the number of steps taken  $i$  and the absolute distances of each component of  $\mathbf{d}$  (describing  $x, y$  and  $z$  absolute distances) between the nose and the evaluated landmark. Because the influence of each factor is not the same for all contour landmarks, the weighting was adjusted per landmark number  $m$  using the following manually found parameter settings:

$$\alpha = 0.0035$$

$$\mathbf{w} = (0.005, 0.75, w_z)$$

$$w_z = \text{smooth} \left( 1 - \frac{m-8}{8}, -1.8 \right)$$

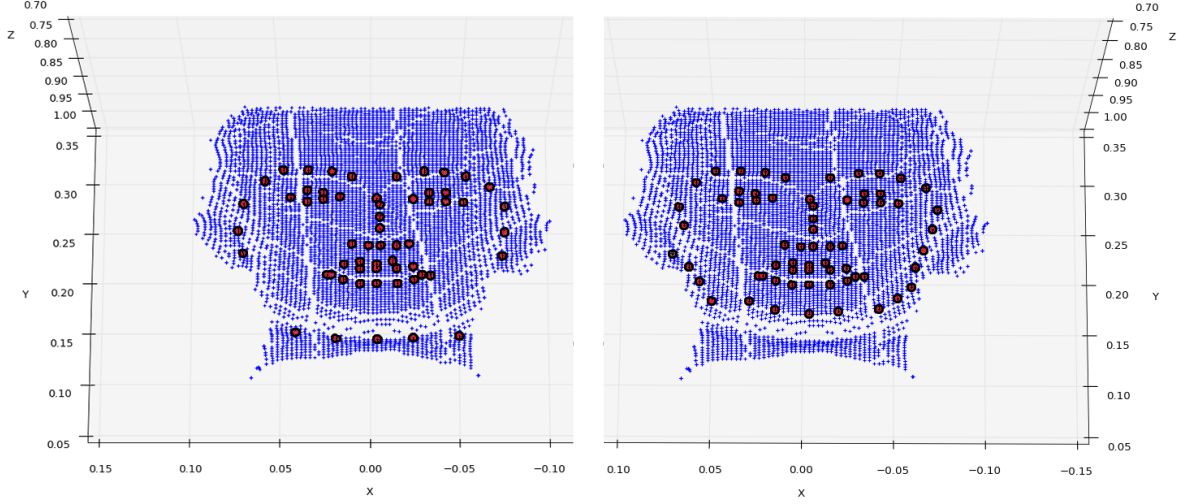
where  $m$  is the landmark number and  $\text{smooth}(x, k)$  is a modifiable normalized sigmoid-like function<sup>2</sup> which smooths the input using the following formula:

$$\text{smooth}(x, k) = \frac{1}{2} \left( \frac{k|f(x)|}{k - |f(x)| + 1} \text{sign}(f(x)) + 1 \right) \quad f(x) = 2x - 1$$

where  $\text{sign}(x) = 1$  if  $x \geq 0$  and  $\text{sign}(x) = -1$  otherwise. Essentially, the weight  $w_z$  thus depends on how close the landmark under evaluation is to the center of the chin (landmark number 8). The closer it gets to the chin center, the higher the  $w_z$  weight gets, which results in adding more importance to decreasing the z-distance between the landmark position and the nose position. Since the  $z$  distance decreases considerably when climbing the neck towards the chin, but less once over the chin rim, it intuitively finds an optimum on the chin rim for the markers close to the chin. For the markers further away from the chin, which have a lower weight to the  $z$ -distance, the face contour correction favors a higher  $z$  value. Hence, these markers move more towards the back of the head. Since the relation between marker number distance from landmark number 8 towards  $w_z$  was found to be non-linear, the  $\text{smooth}$  function was applied with a  $k = -1.8$  to better fit the weighting for the  $z$  distance penalty.

---

<sup>2</sup>Formula adopted from: <https://dinodini.wordpress.com/2010/04/05/normalized-tunable-sigmoid-functions/>



**Figure 2.11:** Pointcloud visualizations of the 3D depth information with the 68 markers (in red) fitted on the 3D face data (blue). On the left without contour correction and on the right with contour correction.

The contour correction method was validated on all the labeled cases by checking for missed values (coordinate (99,99,99)) and by checking for outliers outside of the sensible facial volume ( $z > 1$  meter,  $y < -0.5$  meter). Finally, the chin contours were validated by manually inspection of the 3D pointcloud visualizations. The method was tuned until no more outliers were detected for any labeled case and all manually inspected chin contours were positioned adequately. Figure 2.11 (right) shows the facial landmarks with contour correction applied.

### 2.4.2 Image stabilization

Participants will often raise their head when asked to raise the eyebrows, lower their head when asked to frown and show similar facial movements. These global rigid facial motions disturb motion estimation by image subtraction and calculating marker motion. Image stabilization is therefore applied to compensate for these rigid global motions by a method similar to He et al. [27], except that here also the scale is taken into account for stabilization.

The method boils down to performing an affine transformation to align a source image  $\mathbf{I}_{src}$  to a destination image  $\mathbf{I}_{dest}$ . The parameters are a  $2 \times 2$  matrix  $\mathbf{A}$  and a  $2 \times 1$  translation vector  $\mathbf{b}$ . The transformation takes old coordinates  $(x', y')$  into novel coordinates  $(x, y)$  by applying the following transformation:

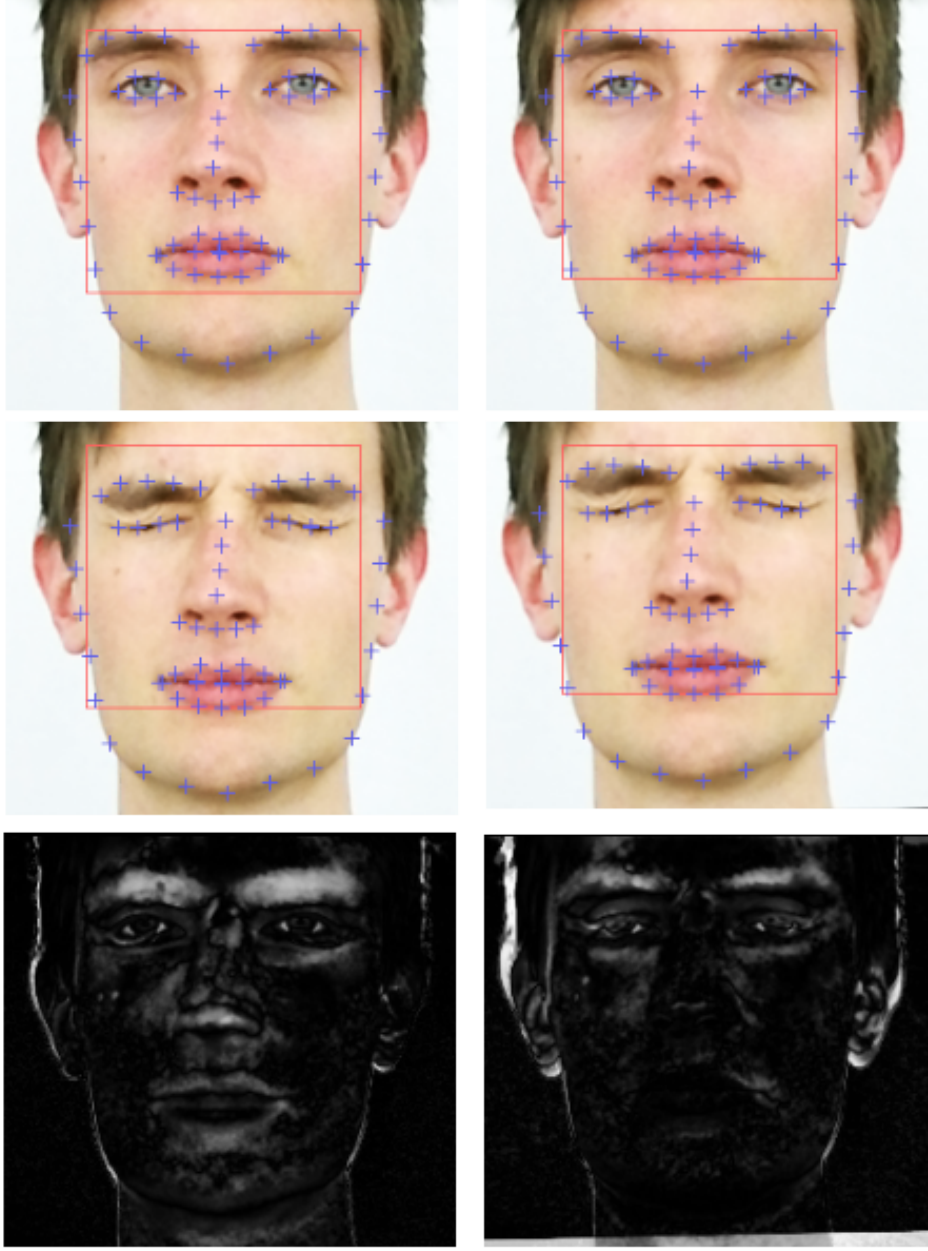
$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & b_1 \\ a_{21} & a_{22} & b_2 \end{bmatrix} \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = [\mathbf{A}|\mathbf{b}] \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix}$$

The problem of finding the parameters for  $\mathbf{A}$  and  $\mathbf{b}$  can be formulated as follows:

$$[\mathbf{A}^*|\mathbf{b}^*] = \arg \min_{[\mathbf{A}|\mathbf{b}]} \sum_i ||\mathbf{I}_{dest}[i] - \mathbf{A}\mathbf{I}_{src}[i]^\tau - \mathbf{b}||^2$$

where  $\mathbf{I}_{src}[i]$  and  $\mathbf{I}_{dest}[i]$  are the  $i$ -th points in the source and destination images respectively. In our case the affine transformations have been limited to rotation, uniform scaling and translation, which greatly simplifies the problem.





**Figure 2.12:** Participant closing the eyes firmly at rest (top) and at maximum expression (middle) and the absolute differences between the two images (bottom). The white pixels from the bottom images indicate the differences between the other two images. The left column shows the images without image stabilization and the right column shows the same images where image stabilization is applied. The red rectangle indicates the area (bounding box) used for calculating the image stabilization parameters.

To find the optimal values for  $\mathbf{A}$  and  $\mathbf{b}$ , the image stabilization function “estimateRigidTransform” from OpenCV 2.0 [54] was used. The input regions for the function were based on the bounding box of all the manually marked landmarks established at section 2.3 of the face minus the facial contour (landmarks 0 up until 17). Here, the bounding box is the smallest rectangle enclosing the selected set of landmarks. Figure 2.12 shows the images at rest (top) and at active conditions (middle) and the absolute differences between the two (bottom). The input bounding boxes used for calculating the image stabilization parameters are shown in red.



In case the OpenCV 2.0 image stabilization failed for the defined bounding box, the crop box was used instead for image stabilization, which resolved the image stabilization for most of the cases. In rare cases, where that also failed, the identity transform was set for the parameters. The second type of stabilization failure only happened for participant 31 when closing the eyes firmly, showing the teeth and puffing the cheeks and for participant 48 while frowning. In all such cases the participants pitched their heads very far up or down on the expressive state, making the difference too great to compensate by image stabilization.

## 2.5 Feature extraction

Potentially relevant features were extracted from the collected video materials to be later analyzed on their contribution as objective measures and usefulness for classification. This section focuses on the description of the extracted features and extraction procedures.

### 2.5.1 Euclidean distances

To calculate the distances between two landmarks  $\mathbf{a}$  and  $\mathbf{b}$  of either 2D or 3D coordinates, the Euclidean distance was calculated:

$$dist(\mathbf{a}, \mathbf{b}) = \sqrt{\sum_i (a_i - b_i)^2} = \|\mathbf{a} - \mathbf{b}\|$$

These distances were calculated between all 68 landmarks resulting in  $\frac{68 \times (68-1)}{2} = 2278$  distances for a single image. Each distance was paired with its contra-lateral counterpart and calculated for both rest and maximum expression images.

### 2.5.2 Triangle areas

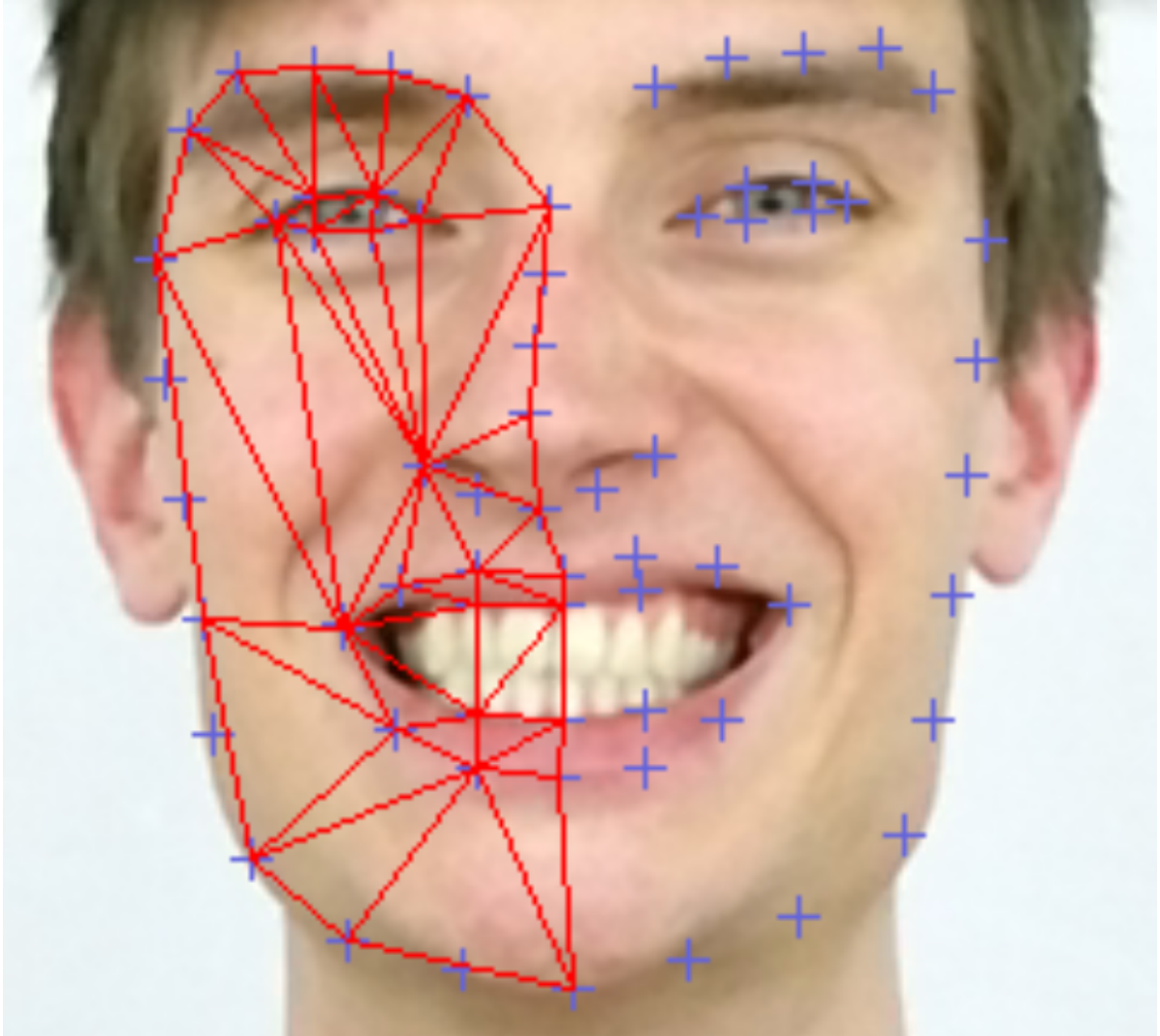
Using three landmarks it is possible to calculate a surface area, which was reported to be an important feature for facial weakness [55]. For three two dimensional landmarks points  $\mathbf{a}, \mathbf{b}, \mathbf{c}$  the triangle area can be calculated by the following formula:

$$area = \left| \frac{a_x(b_y - c_y) + b_x(c_y - a_y) + c_x(a_y - b_y)}{2} \right|$$

Another possibility is to use Heron's formula (written down by Heron in his work *Metrica* ca. 100 BC-100 AD), which works when the lengths of all three sides of the triangle are known. Since all coordinates of the landmarks are known, all distances can be calculated using the Euclidean distance, hence it can be used for calculating triangles between landmarks for both 2D and 3D coordinates. If  $a, b, c$  are the lengths of the sides of a triangle, the Heron's formula is given by:

$$area = \sqrt{p(p-a)(p-b)(p-c)} \quad p = \frac{a+b+c}{2}$$

The triangulation of a set of points  $\mathbf{P}$  is defined as the subdivision of a plane determined by a maximal set of non-crossing straight line edges, i.e. all points in set  $\mathbf{P}$  form as many connections to as many points as possible without crossing any other connections. For a complete triangulation for a set  $\mathbf{P}$  with  $n$  vertices the maximum number of triangles is defined by:  $2 \times n - h - 2$ , where  $h$  is the number of points lying on the boundary of the convex hull of  $\mathbf{P}$  [56]. In the case of the landmarks from Figure 2.9  $n = 68$  and  $h = 27$ , the number of triangles are thus



**Figure 2.13:** The 44 selected triangle areas for the right side of the face.

$2 \times 68 - 27 - 2 = 107$  for a full triangulation. Note however that there are multiple triangulations possible for the 107 triangles.

For our approach we only considered a set of 88 hand picked triangles (44 for each facial side), since enumerating all possible combinations exhaustively would have resulted in an excessive number of features. With the selection the emphasis was placed on having high definition for the mouth and eyes, while having a lower definition for the facial contours, since the former were thought to be more descriptive than the latter. The picked triangles are shown in Figure 2.13. The triangles were mirrored for the left side of the face and calculated for both rest and active conditions.

### 2.5.3 Motion

Motion features estimate the movement of points or the displacement within an area in the face. For this project we employed two type of motion features: a subtraction method and an optical flow method. The subtraction method estimates local motion for a region of the face, but provides no information about the direction of the motion. The optical flow method calculates

the displacement between rest and active states for landmarks. Both methods employ the image stabilization method from section 2.4.2, which reduces the global displacement between the 2D image states. Because of the nature of the image stabilization all the motion features are solely two dimensional.

**Subtraction** To estimate motion not only from landmarks but also for the surface area of the face, we used a subtraction method, which is very similar to the one described by Neely et al. and He et al. [39, 27]. First the images are stabilized in order to match the regions. Then five image regions were determined for the image at rest for each side of the face: for the eyes, the brows, the nose, the mouth and the chin. The areas were determined from their respective landmark positions. The bounding box for the entire face was initially calculated, which is defined as the smallest (non-rotated) rectangle which encloses all the facial landmarks minus the face contour landmarks (markers 0 up until 17).

The eyebrow region was determined by first taking the bounding box of the 5 eyebrow landmarks and subsequently extending the inner region boundary until it was aligned with the  $x$  position of landmark 27 and extending the bottom of region until it was halfway between the bounding box for the eyebrows and the bounding box for the eyes.

The eye region was defined by taking the eye bounding box for the 6 eye landmarks, then extending the top of the region to be halfway between the eye and the eyebrow bounding boxes. Next, the inner region boundary was extended to be halfway between the inner boundary of the eye bounding box and the  $x$  position of landmark 27. The outer side of the region was extended to be halfway the outer side of the eye bounding box and the outer boundary of the face bounding box. Finally, the bottom of the eye region was aligned with the  $y$  position of landmark 28.

The nose region was based on the bounding box of the 7 landmarks of the nose on one side of the face minus landmarks 27 and 28, for which the outer region boundary was extended to be halfway the nose bounding box and the face bounding box.

The mouth region started with the bounding box of the 12 mouth landmarks to one side of the face. The top of the region was then extended to be halfway the top of the mouth bounding box  $y$  position and the bottom of the nose bounding box  $y$  position. The outer side of the nose region was set to be halfway the outer side of the face bounding box and the outer side of the mouth bounding box.

The chin region top was aligned to be at the bottom of the mouth bounding box, the chin region bottom was defined as halfway the bottom of the bounding box for all landmarks and the bottom of the mouth bounding box. The left and right boundaries of the chin region are respectively equal to the left and right boundaries of the mouth bounding box.

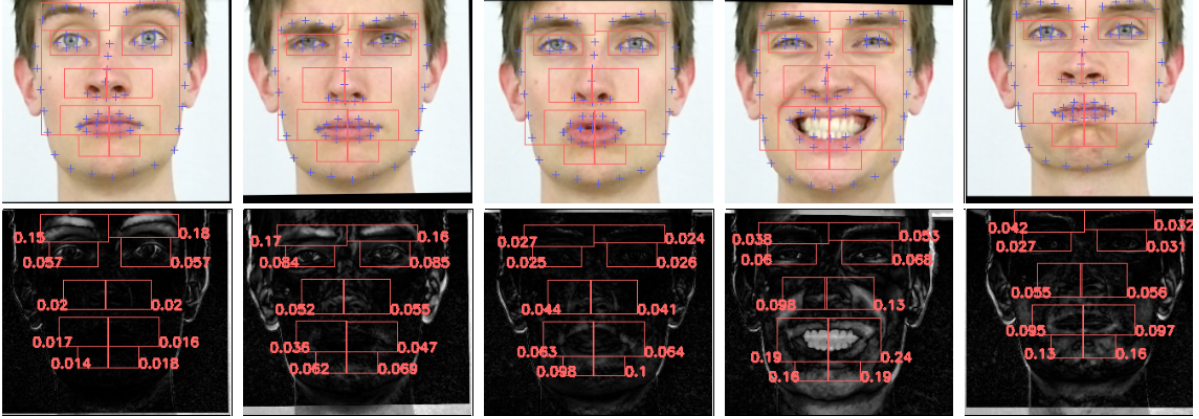
After stabilization of the images and establishing the facial regions, the images were converted to gray scale using OpenCV 2.0 [54] “cvtColor” function with the “COLOR\_RGB2GRAY” flag, so each pixel intensity assumed a value in the interval  $[0, 255]$ . Subsequently the local motion for each facial area  $R$  was calculated using the summed saturated absolute distance  $m_R$  and the normalized summed saturated absolute distance  $\hat{m}_R$  between the rest image  $I_{rest}$  and the stabilized image at maximum expression  $I_{expr}$ :

$$m_R = \sum_{(x,y) \in R} \text{satuate}(|I_{expr}[x,y] - I_{rest}[x,y]|)$$

$$\hat{m}_R = m_R \frac{1}{\sum_{(x,y) \in R} 255}$$

$$\text{satuate}(x) = \begin{cases} 0 & \text{if } x < 0 \\ 255 & \text{if } x > 255 \\ x & \text{otherwise} \end{cases}$$

where  $\text{satuate}(x)$  is a function that clamps the pixel intensities within the  $[0, 255]$  interval. Figure 2.14 shows the 10 estimated regions together with their numerical results for the normalized summed saturated absolute distance values  $\hat{m}_R$ . It also shows task corresponding motion scores. For the final subtraction feature set both the absolute summed pixels  $m_R$  and the region normalized values were calculated.



**Figure 2.14:** Facial regions (red rectangles) for the subtraction method with the normalized summed saturated absolute distance values (bottom row). The top row shows the region fitting using the landmarks. The columns from left to right correspond to the following tasks: raising the eyebrows, frowning, pursing the lips, smiling and puffing the cheeks.

**Optical flow** The subtraction method does not contain directional information for the motion although this information can usually help to discriminate normal motion from abnormal motion [27]. For tracking the direction of motion, usually a set of points are equally distributed on the face in rest and subsequently tracked using the Lucas Kanade algorithm [32, 27]. However, since we only use two frames, motion for the landmarks can be obtained after image stabilization by simply subtracting the landmark at maximum expression from the landmark at rest for both vertical and horizontal displacement and depth displacement if it was available.

#### 2.5.4 Sjögreen features

Sjögreen et al. [57] explored quantitative methods for evaluating lip function and defined a few simple geometric formulas for that purpose based on facial markers. He defined the following set of calculations for estimating lip mobility from 4 markers  $ROC$ ,  $LOC$ ,  $CB$ ,  $LL$  of the mouth corresponding respectively to landmarks 54, 48, 51, 57 from Figure 2.9. They defined the following set of formulas:

$$\begin{aligned}
MW &= ||ROC - LOC|| \\
MW_{left} &= ||LOC - MC|| \\
MW_{right} &= ||ROC - MC|| \\
A &= 100 \left| \frac{MW_{right} - MW_{left}}{MW_{right} + MW_{left}} \right| \\
R &= ||\mathbf{a}_{expr} - \mathbf{b}_{rest}||
\end{aligned}$$

Here  $MW$  is the mouth width,  $MW_{left}$  and  $MW_{right}$  are the mouth widths from respectively the left and right oral commissures to the midline of the face. Here  $MC$  is the interpolated position on the line between  $ROC$  and  $LOC$  crossing the midline of the face, which we determined here by interpolation between the  $CB, LL$  line and the  $ROC, LOC$  line.  $A$  is an asymmetry measure for the face and  $R$ , called the resultant, gives a measure of global displacement between movement and rest conditions.

Since the global displacement  $R$  features were also captured by the motion features from section 2.5.3, they were not included.  $MW_{left}$ ,  $MW_{right}$ ,  $A$  and  $MW$  were calculated for each rest and active condition and were adopted for both 2D and 3D use.

### Asymmetry calculations

Since FSHD and facial weakness are often associated with asymmetrical weakness, a promising approach is the comparison of the left and the right half of the face for participants. A generic approach was taken similar to the work of Azoulay et al. [19] for estimating asymmetry. For both sides of the faces, feature vectors  $\mathbf{l}_{rest}, \mathbf{l}_{expr}$  and  $\mathbf{r}_{rest}, \mathbf{r}_{expr}$  of equal length are calculated, for the face in rest and in maximal expression. By subsequently performing the following calculation a feature vector  $\mathbf{f}_{asymm}$  for the asymmetry between the two face halves for those features is obtained:

$$\mathbf{f}_{asymm} = \frac{\mathbf{l}_{expr}}{\mathbf{l}_{rest}} - \frac{\mathbf{r}_{expr}}{\mathbf{r}_{rest}}$$

This formula takes the difference between facial halves by first normalizing using the resting condition and then by taking the difference between the left and the right sides. If the facial expression is the same on both sides, the features within vector  $\mathbf{f}_{asymm}$  are close to zero, otherwise some of the features are either negative or positive, based on which side is more expressive.

#### 2.5.5 Other features

Age and gender were also added as features.

## 2.6 Ground truth

Grade	Mouth/forehead	Eyes
1	Impossible to initiate movement	Incomplete eye closure
2	Limited movement possible	Eyelashes mostly visible on closure
3	Near complete movement possible	Small rim of eyelashes visible on closure
4	Complete movement possible without effort	Complete eye closure without effort

**Table 2.3:** Translation of the Dutch instructions on the score sheet.

To obtain the ground truth for the facial weakness within specific regions of the face, the following three experts were asked to score specially tailored evaluation videos of the participants.

**Human expert 1 (Ex1)** dr. Carien Beurskens, mime therapist, with much experience on head and neck oncology and facial paralysis.

**Human expert 2 (Ex2)** prof. George Padberg, a prominent FSHD specialist.

**Human expert 3 (Ex3)** Simone Knuijt, speech language pathologist, which specializes in facial weakness.

All experts had several years of experience with assessing facial weakness and were briefly instructed about the research. The experts watched and graded 91 evaluation videos corresponding to each participant by filling in digital score sheets, which are included within appendix B. Each evaluation video was created from the video cuts of the frontal PowerShot camera showing the front of the face of the participant and was cropped to show only the face of the participants, excluding the shoulders when possible and was rotated to show the face in a normal frontal position. All audio from the evaluation videos was removed, the order of the participants was randomized and the participant tags replaced with non informative numbers to avoid providing label information to the experts. All tasks were positioned in the same order: closing the eyes gently, closing the eyes firmly, raising the eyebrows, frowning, pursing the lips, showing the teeth and puffing of the cheeks. Each task was repeated 3 times and was followed by a 4 second blank to allow the experts to note down the scores for both the right and left side of the face for that task on the digital score sheet. It is important to note here that the left and right sides should be considered from the perspective of subject as is medical tradition. The scoring of facial weakness for each task were done on a 4-point scale ranging from total facial paralysis (1) to normal movements without effort (4). Additionally, Table 2.3 shows the (translated) instructions for the grading system used, which the experts were asked to use for scoring each task.

After all scores were gathered, the facial weakness ground truth is defined as the median score between the three experts for each task and side and were used to train the facial weakness classifiers on. Because the weakness classes were very imbalanced, i.e. most participants had none (4) to mild facial weakness (3) and only few had severe to total facial weakness (2 & 1), which gave problems training the facial weakness grading systems, the choice was made to transform the four point scale to a three point scale using the following transformation:  $1 \rightarrow 1, 2 \rightarrow 1, 3 \rightarrow 2, 4 \rightarrow 3$ .

The ground truth for FSHD for each participant were known from DNA tests, but were not known by the experts. In addition to scoring the facial weakness for all tasks, the experts were asked to estimate from the videos if the participant had FSHD or not.

Feature set	tasks	items	sided	states	points	#features
Distances	0-6	1	×	×	2D & 3D	72618
Triangles	0-6	1	×	×	2D & 3D	4212
Subtraction	0-6	5	×		2D	105
Subtraction abs	0-6	5	×		2D	105
Motion	0-6	1	×	×	2D	1218
Motion mid	0-6	1		×	2D	140
Sjögreen MWRL	0-6	2	×	×	2D & 3D	98
Sjögreen A	0-6	2		×	2D & 3D	84
Age	-	1			-	1
Gender	-	1			-	1
FSHD labels	-	1			-	1
Expert Weakness labels	0-6	4	×		-	56
Expert FSHD labels	-	4			-	4

**Table 2.4:** The feature sets and their characteristics.

## 2.7 Combining features

The features described in section 2.5 have several common characteristics that were exploited in order to facilitate generalization and implementation and to provide the features with meaningful descriptive labels. Features can be characterized by feature type, task, iteration, state (at rest or in motion), sidedness (right or left side of the face) and whether they utilized 2D or 3D landmark information. Table 2.4 lists all the extracted feature sets with their characteristics. “Tasks” indicates if the features were calculated for each task, “items” indicates how many different features were calculated within the feature set, “sided” indicates if the features were calculated separately for both parts of the face, “states” indicates if the difference between rest and active conditions was used, and “points” indicates if 2D or 3D type points were used for calculating the features. Finally “#features” gives the total number of features calculated for the feature set for a single participant. All features from Table 2.4 were calculated for 87 participants and subsequently concatenated in one big feature matrix ( $87 \times 78645$ ) together with the ground truth values to be used for later analysis.

To calculate all features and labels for a feature set, a general iterator was implemented, which could be sub-classed in order to implement the feature calculations and automatically generate the corresponding labels. In addition to automatically generating the labels, it also automatically mirrored the landmarks from the right side to the left so only one side of landmark points needed to be defined. Furthermore, the iterator would check if a feature set calculated multiple states and multiple facial sides and would automatically extrapolate asymmetry features and differences based on those calculated values. What features were extrapolated, depended on the sides and states flags. If a feature set did not define both sides and states, it would not extrapolate any extra features. If a feature set used either sides or states, it would extrapolate either the difference between the sides (L - R) or the states (active - rest). When both sides and states were calculated, the method from section 2.5.4 was applied, which first normalizes the active states by the rest states and then subtracts the normalized right from the normalized left side.

The splitting of the features sets “Subtraction”, “Motion” and “Sjögreen” within Table 2.4 arises from a different usage of iterators and hence each has a different implementation. For the “Subtraction” feature set both the absolute and the relative subtractions were treated separately. For the “Motion” feature set the landmarks at the center line add no useful extra information by mirroring the features since they would be the same points and are hence treated separately to avoid redundant features. For the “Sjögreen MWRL” feature set the left and right sides were compared, which was not the case for the “Sjögreen A” feature set.

### 2.7.1 Feature labeling

The iterator was also used to automatically generate labels for each feature within the features sets. The labeling was based on the same feature characteristics as listed in Table 2.4, but includes additional detail. The general format of the labeling is shown in Table 2.5. The label should be read from left to right and was always ordered in the same manner. The value of  $x$  in the label can take any of the values specified below the label row. For the task this is a value between 0 and 6 describing the task for which the feature applies. If the label concerns a general feature that applies to all tasks an underscore token is set. Similarly for the iterations,  $x$  can take a value from 1 to 3 for any of the iterations or an underscore if it applies to all iterations. For the items,  $x$  can take a value greater or equal to 0 or an underscore if it has no item set. For the side there are five different states: it can be a feature that describes no particular side ( $\_$ ), describes either the right (R) or the left (L) side of the face, or can describe an aggregated feature (A or D) that was obtained by combining sides or states or both. For the facial states the feature can apply to the normal rest condition (N), the maximal expression condition (M), or apply to no condition in particular ( $\_$ ). When landmarks are used, the landmark point types used can be set to either 2D (2) or 3D (3). Otherwise, if no landmarks are used, it was set to ( $\_$ ). When any of the 68 landmarks were used, each used point was listed by enumerating the point numbers as a two digit number with the letter ‘p’ prepended (e.g. p00, p01, p02, ... ,p68). Finally, the feature set to which the feature belongs was set as the tag at the end of the generated label. In case a feature set concerned label information, the feature set tag was set to contain the “LABEL” string.

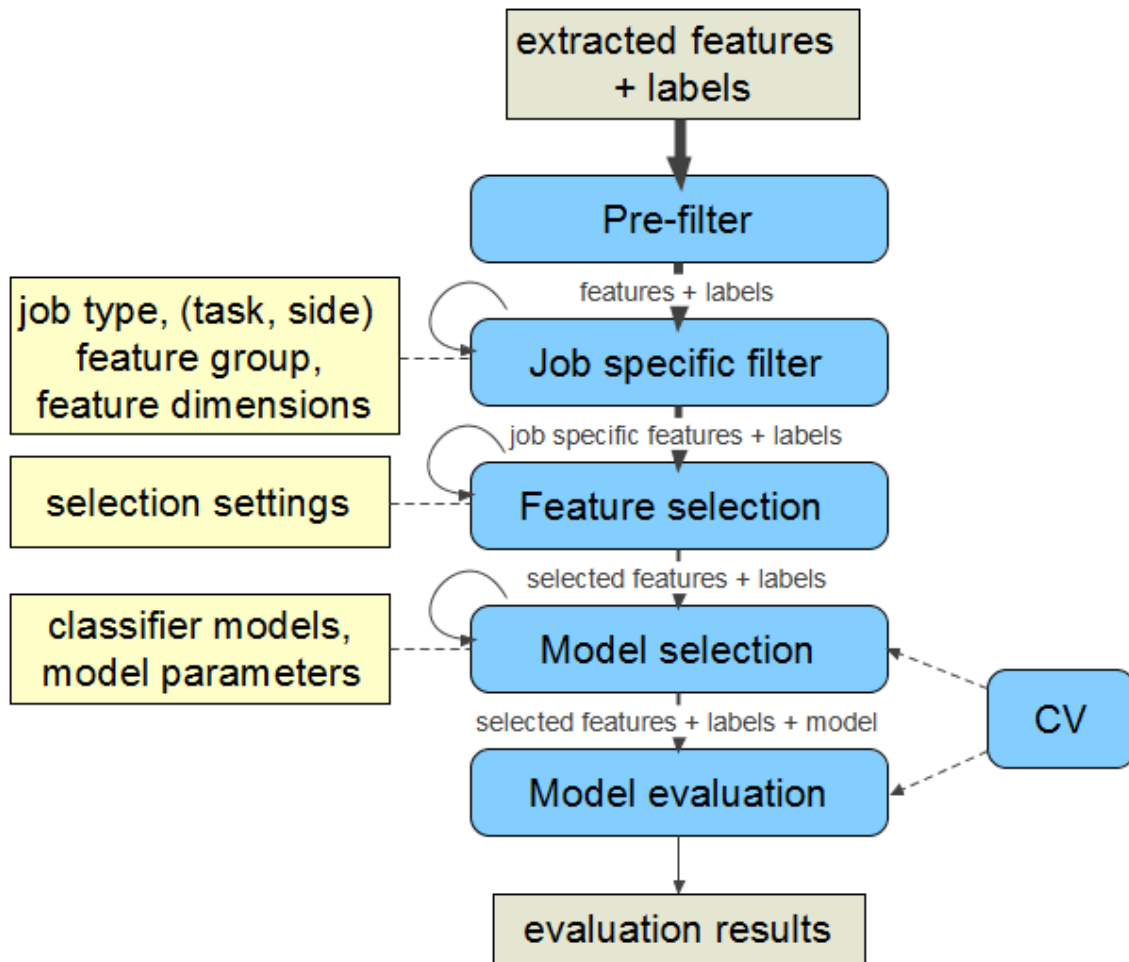
An example label is: “T5\_I1\_n0\_RM3\_p39p41\_dist”, which describes a feature from the distance feature set calculated from the 3D Kinect data between landmarks 39 and 41, which applies to the right side of the face when the participant is in maximal expression while showing the teeth (task 5).

	task	iteration	items	side	state	point type	points	tag
label	T $x$ _	I $x$ _	n $x$ _	$x$	$x$	$x$ _	$x$	$x$
$x$	0-6, _	1-3, _	>=0, _	R,L,A,D,_	N,M,_	2, 3,_	points	text

**Table 2.5:** The label format.



## 2.8 System evaluation pipeline



**Figure 2.15:** Overview of the system evaluation pipeline.

What we are interested in is finding a good *classifier model* and good *feature subset* from all the extracted features from section 2.7. This is of course dependent on what we want to classify: determining if a participant has FSHD or determining the degree of facial weakness. Let us call this factor “job type”. Also when classifying the degree of facial weakness we need to consider the “task” and the “side” to classify. Since there are 7 tasks and 2 sides at least 14 separate classifiers are required for grading facial weakness. Furthermore, since the facial weakness ground truth is ordinal in nature, we should prefer regression over binary classification models. For diagnosing FSHD we are looking for a single binary classifier.

The extracted featureset contains a lot of redundant and irrelevant features. We want to select from the featureset a subset that optimizes the performance of the classifier on predicting the ground truth. This task is called feature selection. In this particular case we are also interested in the effectiveness of different “feature groups”, e.g. how distance features compare with triangle features, how using all features compare with asymmetry features, etc. Furthermore, we are interested in comparing the effect of “feature dimensions” on classifier performance, e.g.

2D features versus 3D features versus both 2D and 3D features. Finally, since even after selecting feature subsets based on group and dimension these still contained many features of which a lot were not that useful. To further reduce the feature subset size we used different feature selection methods with different “selection settings”.

Model selection is concerned with selecting the right “classifier model”. This means selecting between regression and classification models, different types of models and “model parameters”. For diagnosing FSHD different binary classifiers were selected evaluated and for grading facial weakness several regression models were selected and evaluated. The machine learning models were selected based on their interpretability and nature of the problems.

By having predefined sets of features and classification models, the feature selection and model selection essentially become search problems. I.e. both require searching the predefined sets of respectively features and classification models and selecting the most promising combination of candidates. However, feature selection and model selection are closely related, since some classifiers prefer certain kind of features. Since the number of possibilities (i.e. the search space) is very large and simple enumeration of possibilities would take too much computation time, a good search strategy is essential.

Our approach towards selecting the optimal feature subsets, feature selection settings, classifier model and model parameters is by predefining promising candidate sets for each factor and subsequently evaluating each combination. This approach resulted in the system evaluation pipeline depicted in Figure 2.15. The evaluation system takes as input the extracted features and labels from section 2.7 and all the predefined candidate sets for each factor: job type, task, side, feature group, feature dimensions, selection settings, classifier models, model parameters. The evaluation system calculates and returns multiple evaluation results for each combination of factors and writes those results to a text file. There are multiple processing steps within the system evaluation pipeline, starting from the top we have the pre-filter, the job specific filter, the feature selection, the model selection and the model evaluation steps. The pre-filter step is performed only once. It removes known irrelevant features, applies general transformations on the feature data and passes the processed features and labels to the job specific filter. The job specific filter selects features based on the candidates within the “job type”, “task”, “side”, “feature groups” and “feature dimensions” factors and passes the selection to the feature selection step. The job specific filter iterates all possible factor combinations until all have been processed. The feature selection step performs multiple feature selection steps based on the defined “selection settings”, with feature selection method and feature selection parameters. A sub-selection of the features is then passed on to the model selection step, which enumerate all “classifier models”. Model parameter selection is implicitly performed by performing a grid search within a model selection iteration. Finally the selected model, the selected features and labels are passed on to the model evaluation step, which generates various results for that specific combination of features and classification model and writes the evaluation results to a file for later analysis. For both model selection and model evaluation 10 fold cross validation (CV) is used.

In this section the various components of the system evaluation pipeline are described in detail.

### 2.8.1 Pre-filter

The pre-filter takes as input the featureset from section 2.7 and removes irrelevant and near irrelevant features from the extracted featureset, standardizes the data and processes the facial weakness ground truth. The processing steps are in order:

1. Domain knowledge was used to discard all features that only consisting of face contour points. These features were thought to be not relevant for facial weakness nor for diagnosing FSHD.
2. Secondly, a variance threshold filter was used to remove all features having a variance of 0.0475 or less, which have little added value for classification. E.g. a feature with the same values for all participants has a variance of 0 and has no added value for classification. This step roughly reduced the features by one third.
3. The remaining features were standardized.
4. The distribution of classes of facial weakness within participants was heavily skewed, because extreme to severe facial weakness did almost never occur. The problem with this is that regression models requires enough samples for those cases to learn the facial weakness relation, hence the decision was made to collapse those classes into a single class representing both. All the facial weakness ground truth scores were reformatted from a 4 point ordinal scale to a 3 point ordinal scale in the following manner:  $1 \rightarrow 1, 2 \rightarrow 1, 3 \rightarrow 2, 4 \rightarrow 3$ .

### 2.8.2 Job specific filter

The job specific filter first of all considers the “job type” factor, if the job concerns diagnosing FSHD, the “task” and “side” factors were ignored. Otherwise, when grading facial weakness, all 14 combinations between task and side were generated and separately evaluated. Independent of “job type” all combinations between “feature group” and “feature dimensions” were generated and evaluated. The chosen feature groups were:

1. All feature types
2. Only distance features
3. Only motion features
4. Only triangle features
5. Only subtraction features
6. Only asymmetry features

For “feature dimensions” the chosen feature groups were:

1. 2D features only
2. 3D features only
3. 2D & 3D features

When evaluating facial weakness domain knowledge was utilized by discarding all non-task related features. I.e. non-task related features are either irrelevant or redundant and are assumed to not contain any information for predicting the class labels. E.g. when a participant closed the eyes, all other task features like puffing the cheeks and showing the teeth were removed. Note however that no features were discarded for the “side” factor. Thus, features from the entire face were used for learning to grade the facial weakness for a particular side of the face.

### 2.8.3 Feature selection

Feature selection deals with selecting a subset of features from all extracted features so the most relevant and informative combination of features remains for training a machine learning model. The main purpose of feature selection is to remove redundant and irrelevant features, where redundant features provide no more information than other selected features and irrelevant features provide no useful information at all. The removal of irrelevant and redundant features help to enhance model interpretability, shorten model training time, reduce overfitting and improve generalization.

While intuitively it is possible to exhaustively try out every possible subset of features, this is not realistic for huge feature sets. Therefore several techniques have been developed for smarter feature selection, which can be broadly categorized in: wrapper methods, filter methods and embedded methods.

Wrapper methods use a predictive model like SVM to score feature subsets. Each new subset is used to train the model and is tested on a separately stored test set. Using a score like the error rate (number of mistakes) for classification or Mean Square Error (MSE) for regression to evaluate its performance. Wrapper methods are the most computationally intensive, but can usually provide the best performing feature set for a particular model. Exhaustive search is an example of a wrapper method.

Filter methods approximate the usefulness of a feature subset and are faster to compute than wrapper methods. In general they calculate some measure for the feature subset to evaluate the scores. The chosen measure greatly influences the outcome. Some measures are: mutual information, correlation, inter- and intraclass distances or significance tests scores for each class/feature combination [58]. The trade-off for computational speed is that the selected features are generally not specific to a model and hence the obtained subsets are usually inferior to wrapper methods for classification performance. A big advantage of filter methods is that the obtained subset does not rely on the mechanics and assumptions of a machine learning model and can be more informative regarding the feature relations and characteristics.

Embedded methods perform feature selection as integrated part of the model learning. The Lasso method, for example, is a well known approach for constructing a linear model and combined feature selection. It iteratively penalizes the regression coefficients of the model until some become zero and are subsequently removed from the feature set. Performance of embedded methods vary between wrapper and filter methods computation times.

### Approaches

Three different feature selection methods (wrapper) were evaluated: L1-based Support Vector Machine feature selection (L1-SVC) and Randomized Logistic Regression (RLR, also known as Stability Selection) [59] for classification and Randomized Lasso [59] for regression. All methods generally selects a very sparse set of features (i.e.  $< 200$ ) from the job specific feature set, which is convenient for further model selection, model training and later model interpretability. The LinearSVC, RandomizedLogisticRegression and RandomizedLasso implementations from the Scikit-learn [60] Python machine learning library (version 0.16.1) were used for the feature selection methods.

1. A Support Vector Machine Classifier with an L1 penalty (L1-SVC) is known to produce sparse solutions. The L1-SVC was fitted to the data, i.e. coefficients for each feature were trained, based on the feature labels. Subsequently, the features with a coefficient greater than zero were kept. For controlling the sparsity of the selected features the  $C$ -parameter was tuned for the following set of values for  $C$   $\{0.1, 1, 10\}$ .

2. The following parameters were fixed for the Randomized Logistic Regression (RLR) approach: `sample_fraction=0.8`, `n_resampling=200`, `scaling=0.5`, `tol=1e-3`, `selection_threshold=0.25`. For controlling the sparsity of the selected features the  $C$ -parameter was tuned for the following set of values for  $C$   $\{3, 10, 20\}$ .
3. The following parameters were fixed for the Randomized Lasso (RL) approach: `sample_fraction=0.75`, `n_resampling=200`, `scaling=0.5`, `selection_threshold=0.25`. For controlling the sparsity of the selected features the  $\alpha$ -parameter was tuned for the following set of values for  $\alpha$   $\{0.01, 0.005, 0.0025, 0.0001\}$ .

#### 2.8.4 Model selection

For learning the relation between the ground truth (FSHD DNA results or median facial weakness grades)  $\mathbf{y}$  and the extracted features  $\mathbf{X}$  from the data, several different machine learning algorithms were selected and compared. The machine learning algorithms were selected from the Scikit-learn [60] Python machine learning library (version 0.16.1). For classification: Support Vector Classifier (SVC), k-Nearest Neighbors (kNN), Random Forests (RF) and Logistic Regression (LR) were used. Ridge Regression (Ridge), ElasticNet (ENet), k-Nearest Neighbors Regression (kNNR) and Random Forest Regression (RFR) were used for regression.

Each of the models had their own set of hyper-parameters which had to be optimized to find the optimal hyper-parameter set for solving the classification problems. Classification was optimized on accuracy and regression problems were optimized on mean squared error. Optimization was performed by using an exhaustive grid search approach (explicitly enumerating a predefined set of hyper-parameter combinations) and 10-fold cross validation.

The evaluated hyper-parameter sets are listed below for each machine learning model:

##### Classification

For SVC:

```
{'kernel': ['rbf'], 'gamma': [1e-2, 1e-3, 1e-4, 1e-5],
'C': [0.1, 0.5, 1, 10, 100, 1000]},
{'kernel': ['linear'], 'C': [0.1, 0.5, 1, 5, 10, 100, 1000]}
```

For kNN:

```
{'n_neighbors': [1,2,3,5,10], 'weights': ['uniform', 'distance'],
'algorithm': ['auto'], 'metric': ['minkowski']}
```

For LR:

```
{'solver': ['newton-cg', 'lbfgs'], 'C': [0.01, 0.1, 0.5, 1, 5, 10, 100, 1000]},
{'solver': ['liblinear'], 'penalty': ['l2'],
'C': [0.01, 0.1, 0.5, 1, 5, 10, 100, 1000], 'dual': [True, False]},
{'solver': ['liblinear'], 'penalty': ['l1'],
'C': [0.01, 0.1, 0.5, 1, 5, 10, 100, 1000]}
```

For RF:

```
{'n_estimators': [1,10,20,40,80,160], 'criterion': ['gini'],
'max_features': [None], 'bootstrap': [True, False]}
```

## Regression

For Ridge:

```
{'solver': ['svd', 'cholesky', 'lsqr', 'sparse_cg'],  
'tol': [1e-2, 1e-3, 1e-4, 1e-5],  
'alpha': [5.0, 1.0, 0.5, 0.05, 0.005, 0.0005], 'normalize': [True, False]}
```

For ENet:

```
{'l1_ratio': [0.1, 0.25, 0.5, 0.75, 1.0], 'tol': [1e-2, 1e-3, 1e-4, 1e-5],  
'alpha': [5.0, 1.0, 0.5, 0.05, 0.005, 0.0005], 'selection': ['cyclic', 'random'],  
'normalize': [True, False]}
```

For kNNR:

```
{'n_neighbors': [2, 3, 5, 10], 'weights': ['uniform', 'distance'],  
'algorithm': ['auto'], 'metric': ['minkowski']}
```

For RFR:

```
{'n_estimators': [1, 10, 20, 40, 80, 160], 'criterion': ['mse'],  
'max_features': ['auto', None], 'bootstrap': [True, False]}
```

### 2.8.5 Model evaluation

Using the optimal found parameter set for a certain model on the selected features from the feature selection step, the model evaluation step evaluates the tuned model one final time on the selected features using cross-validation and generates relevant evaluation data for further analysis, which was stored to a file. The evaluation data included the following evaluation scores: all factors with their selected levels, the predictions per participant, the selected features by their feature label, the accuracy score, mean squared error, confusion matrix, model parameters. The outcomes of the facial weakness predictions were first rounded to the nearest integer within the [1, 3]-interval, before calculating the evaluation scores.

### 2.8.6 Cross validation

A 10-fold cross validation loop was used for both model selection and model evaluation in order to prevent overfitting and to get a better idea about classifier generalization. For classification the cross validation loop was stratified, i.e. the folds were made by preserving the percentage of samples for each class.

Most machine learning algorithms work best when the number of cases for each class are roughly equal, when this is not the case as with our data then we are dealing with the class imbalance problem (see e.g. Chawla [61] for a thorough explanation of the problem). For regression a special cross validation routine was developed, which implements oversampling to overcome the class imbalance problem. In general with oversampling more samples are drawn from the minority class to compensate for the lack of cases. This can be achieved by either synthesizing new data or by duplicating cases at random from the initial cases of the minority class. In our approach the latter approach was implemented. To avoid overfitting oversampling was part of the cross validation, i.e. on each cross validation step the data was split into a train and a test set, then oversampling was applied on the train set and subsequently the model was trained on the oversampled train set, finally the trained model was evaluated on the test set.

This resulted in better classification outcomes for the regression while still providing honest estimates.

Standardization was performed on the pre-filter step and not within the cross-validation loops on purpose, since the number of cases were small (87 participants). If standardization was performed within the cross-validation loop it occasionally happened that all feature values having a high standard deviation (i.e. outliers) would fall within the test-fold, which resulted in bad standardization parameters. In such a case the standard deviation would be heavily underestimated, which as a result negatively influenced classification performance. In theory, when the number of cases is large enough, this should be less of an issue and standardization should be placed inside the cross-validation loop.

## 2.9 Analyses

The system evaluation pipeline was used to calculate results for all combinations of predefined settings for 30 iterations for classifying FSHD and for 62 iterations for predicting facial weakness. This resulted in large data files on which several statistical analyses were performed in order to provide answers to the research questions. First tests were performed for identifying the best classifier model, feature selection settings, feature dimensions and feature group combination for training the computer systems based on classification accuracy. These analyses were performed using two repeated measures MANOVA tests within IBM SPSS Statistics (Version 20.0.0).

To further evaluate the selected features by the systems the 10% most frequently selected features were visualized for both predicting FSHD and predicting facial weakness to evaluate the relevance of the identified features and relate those the disease.

The best system scores were taken for all 4 classifiers and 4 regressors per task and side, selected on accuracy. The predicted weakness grades for all 87 participants were  $4 \times 14 \times 87$  ordinal scores within the  $[1, 3]$  interval and the FSHD prediction were  $4 \times 87$  single binary scores. These scores would subsequently be compared with the expert predictions and the ground truths.

The FSHD predictions of the system were compared with the human expert predictions and the FSHD ground truth to evaluate the ability of the experts and systems for diagnosing FSHD from facial characteristics using a MatLab implementation of Barnard’s exact test [62]. The weakness scores were compared on inter-rater agreement using Krippendorff’s Alpha via the SPSS Kalpha implementation [63].

To test the relation between the FSHD and facial weakness the pooled minimal expert ground truth is introduced, which is defined as the minimum over all expert ground truths per task and side, yielding a single prediction per participant. A Pearson product-moment correlation coefficient was computed between the pooled minimal expert ground truth and the FSHD ground truth to investigate the strength of their relation.

### 2.9.1 Identifying best model and feature combination

The aim of these tests is to identify the best combination of features, feature selection method and machine learning model for predicting either the facial weakness within participants (for a specific task and side) or for predicting whether a participant has FSHD. The following repeated measures MANOVA was performed on 30 iterations for the FSHD diagnosing computer systems:

### **Dependent variables**

- Accuracy FSHD prediction (quantitative,  $4 \times 6 \times 3 \times 6 = 432$  measures per iteration)

### **Independent variables**

- Between-subject factor classifier model (fixed factor categorical):
  - k-Nearest Neighbors (kNN)
  - Random forest (RF)
  - Support vector classifier (SVC)
  - Logistic Regression (LR)
- Within-subject factor feature group (fixed factor categorical):
  - All feature types
  - Only distance features
  - Only motion features
  - Only triangle features
  - Only subtraction features
  - Only asymmetry features
- Within-subject factor feature dimensions (fixed factor categorical):
  - 2D
  - 3D
  - 2D & 3D
- Within-subject factor selection settings (fixed factor categorical):
  - L1-SVC  $C = 0.1$
  - L1-SVC  $C = 1$
  - L1-SVC  $C = 10$
  - RLR  $C = 3$
  - RLR  $C = 10$
  - RLR  $C = 20$

A similar yet more complex analysis has been performed for the systems predicting the facial weakness scores. The following analysis was performed using a repeated measures MANOVA on 62 iterations:

### **Dependent variables**

- Accuracy facial weakness prediction (quantitative,  $7 \times 2 \times 4 \times 6 \times 3 \times 4 = 4032$  measures per iteration)

### **Independent variables**

- Between-subject factor task (fixed factor categorical):



- Closing the eyes gently
- Closing the eyes firmly
- Raising the eyebrows
- Frowning
- Pursing the lips
- Showing the teeth
- Puffing the cheeks
- Between-subject factor side (fixed factor categorical):
  - Right side
  - Left side
- Between-subject factor classifier model (fixed factor categorical):
  - Ridge Regression (Ridge)
  - ElasticNet (ENet)
  - k-Nearest Neighbors Regression (kNNR)
  - Random Forest Regression (RFR)
- Within-subject factor feature group (fixed factor categorical):
  - All feature types
  - Only distance features
  - Only motion features
  - Only triangle features
  - Only subtraction features
  - Only asymmetry features
- Within-subject factor feature dimensions (fixed factor categorical):
  - 2D
  - 3D
  - 2D & 3D
- Within-subject factor selection settings (fixed factor categorical):
  - RL  $\alpha = 0.01$
  - RL  $\alpha = 0.005$
  - RL  $\alpha = 0.0025$
  - RL  $\alpha = 0.0001$

The goal of the two tests was to find the best combination of features, feature selection method and machine learning model to train a classifier. However, the latter test also tries to identify if scoring sides matters and if there are differences between grading the tasks.

### 2.9.2 Comparison system and experts on FSHD diagnosis

The goal of this test is to validate the obtained system scores on diagnosing FSHD with the human expert scores and the DNA ground truth. Pair-wise comparisons have been performed using Barnard's exact test [62] between all the following experts (including the ground truth as expert):

- FSHD ground truth
- Human expert 1
- Human expert 2
- Human expert 3
- System expert SVC
- System expert LR
- System expert kNN
- System expert RF

### 2.9.3 Comparison system and experts on facial weakness grading

The goal of this test is to validate the obtained system scores on grading facial weakness with the human expert scores (the current golden standard) and the synthesized ground truth. Krippendorff's Alpha Reliability Estimate [63] has been calculated for the following groups, for all 7 tasks and all 2 sides:

- Human expert 1 - Human expert 2 - Human expert 3
- Human expert 1 - Human expert 2
- Human expert 1 - Human expert 3
- Human expert 2 - Human expert 3
- Human expert 1 - Ground truth
- Human expert 1 - Ground truth
- Human expert 2 - Ground truth
- System expert Ridge - System expert ENet - System expert kNNR - System expert RFR
- System expert Ridge - Ground truth
- System expert ENet - Ground truth
- System expert kNNR - Ground truth
- System expert RFR - Ground truth

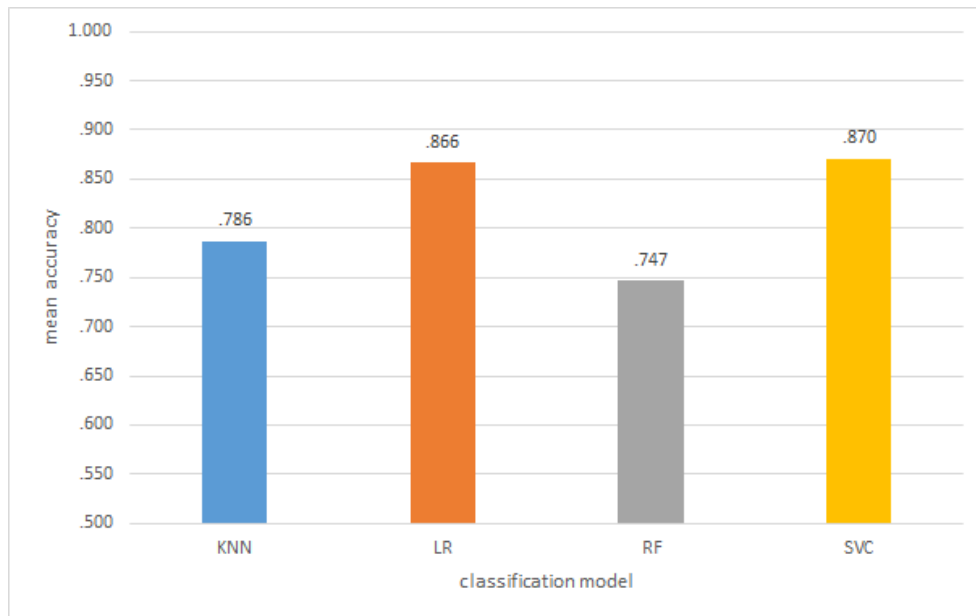
## Chapter 3

# Results

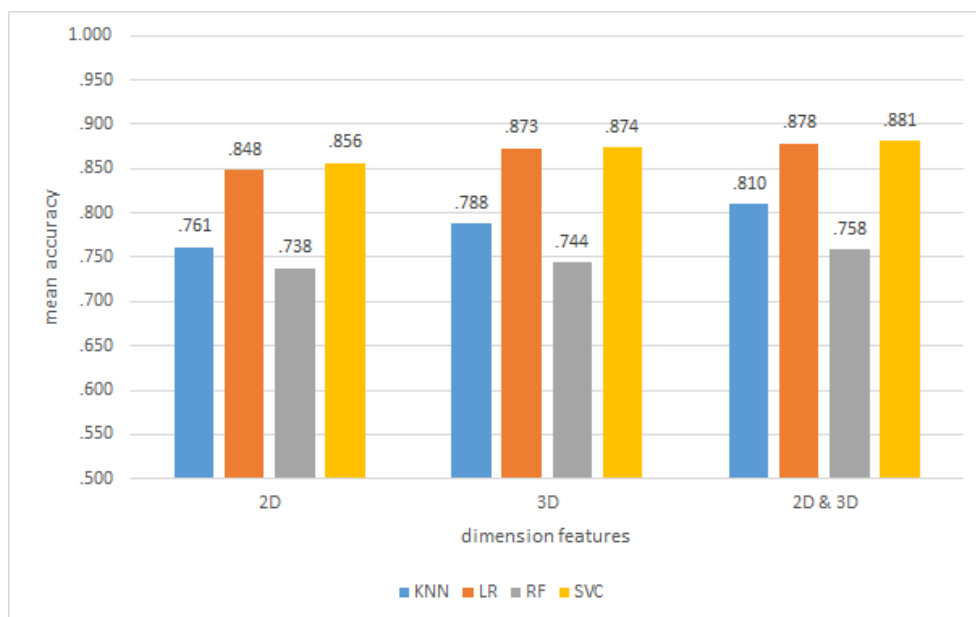
### 3.1 FSHD grading system evaluation results

A 4 (Classifier model) x 6 (Feature group) x 3 (Feature dimensions) x 6 (Feature selection settings) repeated measures MANOVA was conducted in the statistical software package SPSS with classifier model as between-subject factor and feature group, feature dimensions and feature selection settings as within-subject factors. The levels of classifier model were: Random Forest (RF), K-Nearest Neighbors (KNN), Support Vector Classifier (SVC), Logistic Regression (LR). The levels of feature group were: all, distances, motions, triangles, subtractions, only asymmetries. The levels of feature dimensions were: 2D, 3D, 2D&3D and the levels of feature selection settings were: L1-SVC C=0.1, L1-SVC C=1, L1-SVC C=10, stability selection C=3, stability selection C=10, stability selection C=20. The dependent variable was classifier performance measured as the accuracy of predicting the FSHD ground truth for the 87 participants. The classifier model, feature group, feature dimensions and feature selection settings main effects and all their interactions were all found to be highly significant ( $p < 0.001$ ) with moderate to strong effect sizes ( $\eta^2 > 0.580$ ). The exact analysis statistics can be found in Appendix C.1.

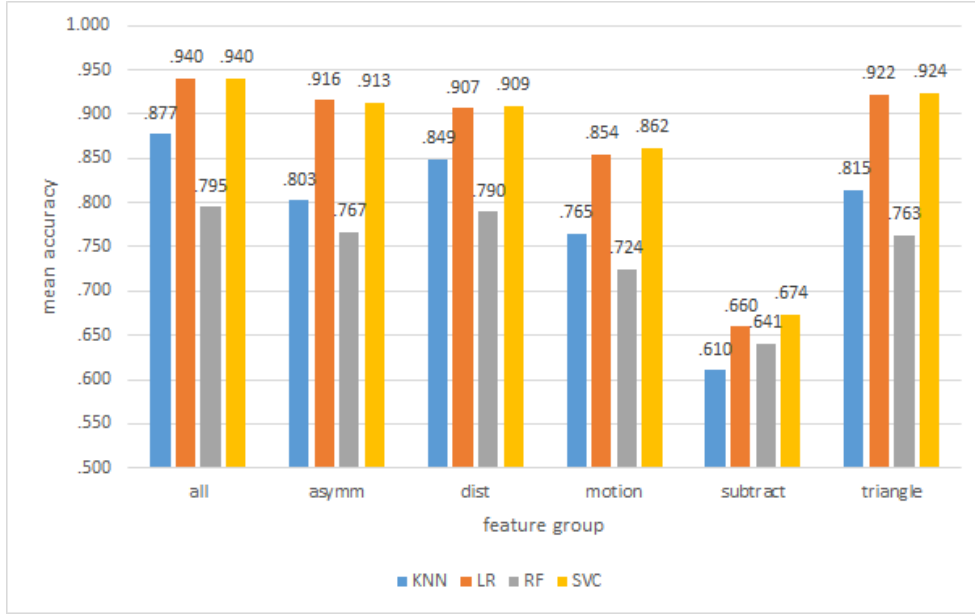
Post hoc pairwise tests were performed with Bonferroni correction for the classifier model between-subject factor, which also showed high significance ( $p < 0.001$ ) with strong effects between classifier models on classification performance ( $\eta^2 > 0.700$ ). Figure 3.1 shows the average accuracies between models, which reveal that SVC was the best classification model, but was closely followed by LR. Figure 3.2 shows the feature groups used per model and reveals that selecting from all features yielded the best results, but selecting only from distances, triangles or asymmetry features also gave good results. Weaker performance was obtained when only selecting from motion features or when only selecting from subtraction features. Figure 3.3 shows the mean accuracy versus the dimension features used. Selecting from both 2D and 3D features gave the highest performance, followed by selecting only from 3D features and then by selecting only from 2D features. Finally, Figure 3.4 shows the accuracy between models for the feature selection settings. The stability selection approach overall outperforms the L1-SVC feature selection method. For L1-SVC best performance was found for C=0.1, while for stability selection C=20 performed better for LR, SVC and KNN. For RF C=3 performed best.



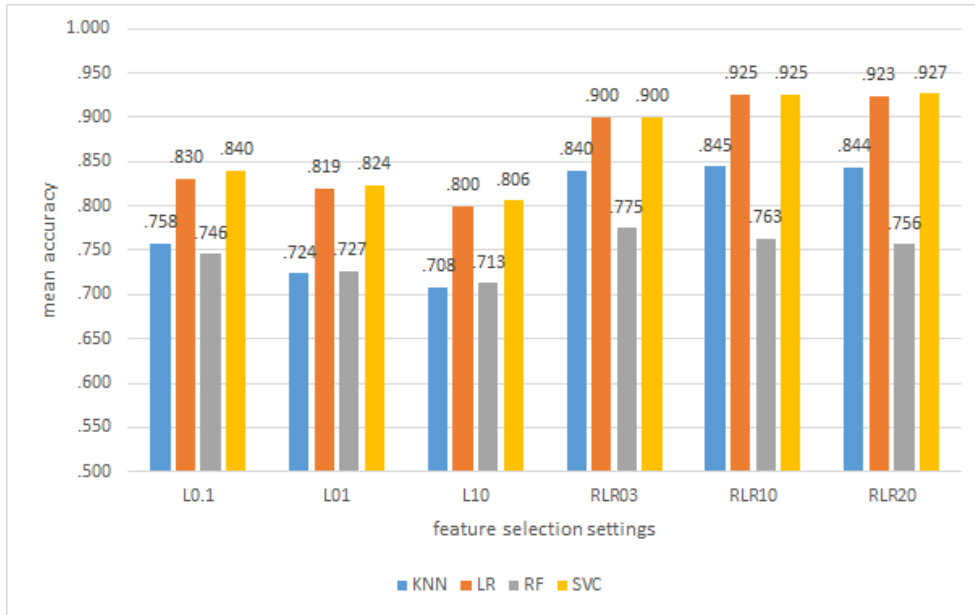
**Figure 3.1:** System performance between classification models. Note that the y-axis starts from chance level (0.5).



**Figure 3.2:** System performance of models between feature groups. Note that the y-axis starts from chance level (0.5).



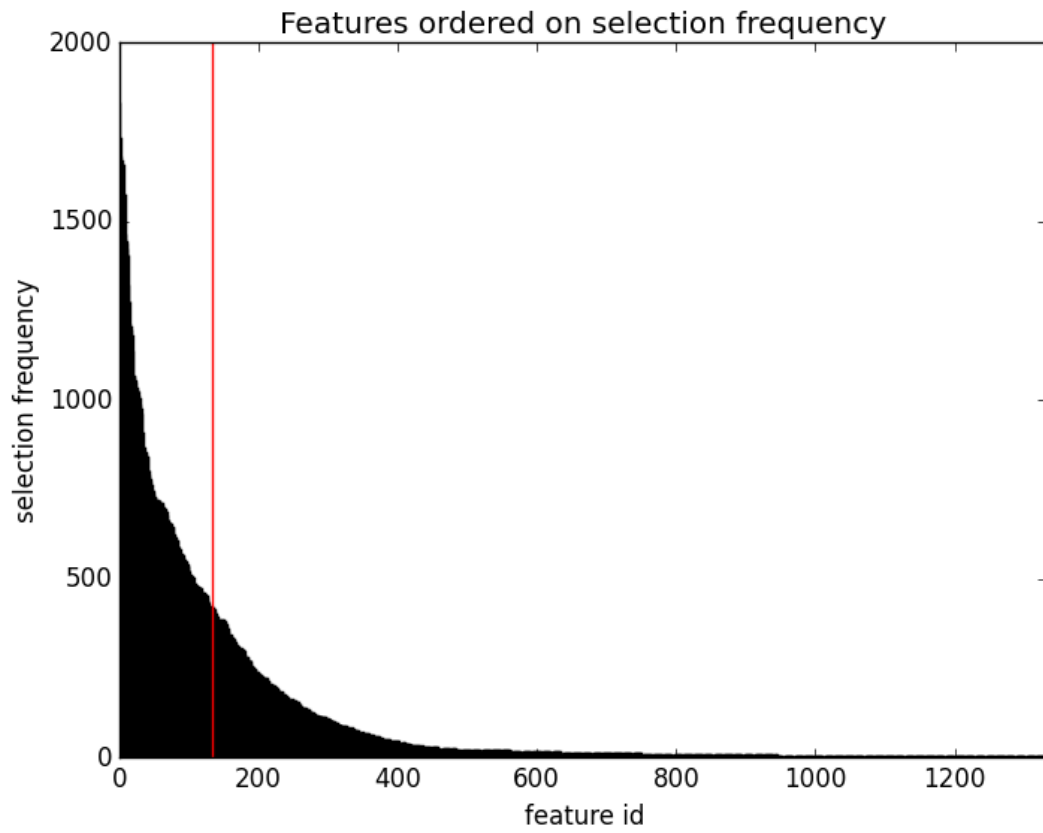
**Figure 3.3:** System performance of models between selected feature dimensions. Note that the y-axis starts from chance level (0.5).



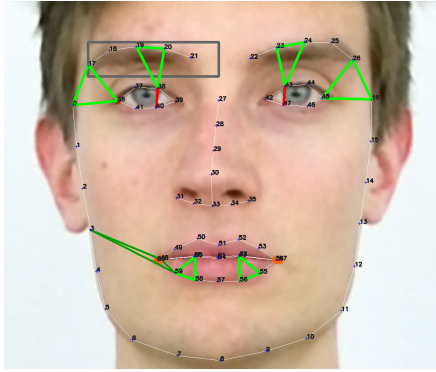
**Figure 3.4:** System performance of models between feature selection settings. Note that the y-axis starts from chance level (0.5).

### 3.1.1 Selected features

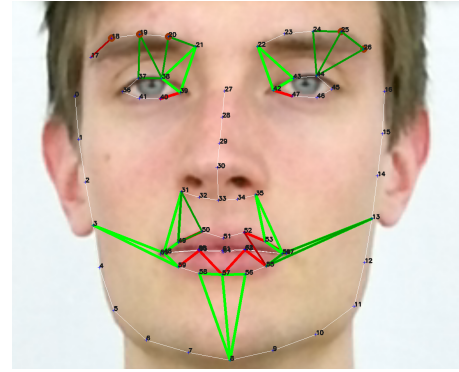
Figure 3.5 shows the resulting selection frequencies for all unique features that were selected from all different training runs of each evaluated system combination which included all features. Out of 1332 uniquely identified features over all different training runs 133 features were selected by taking the 10% most selected features, which are listed in Appendix C.2. The percentage was determined to yield a small subset for proper visualization.



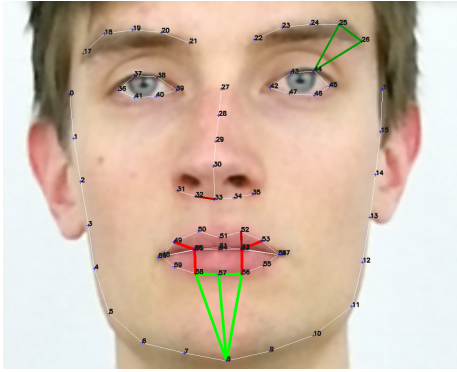
**Figure 3.5:** Feature selection frequencies of all 1332 selected features over all runs for training FSHD models ordered by selection frequency. The cutoff value lies at the 10% most selected features resulting in a set of 133 features.



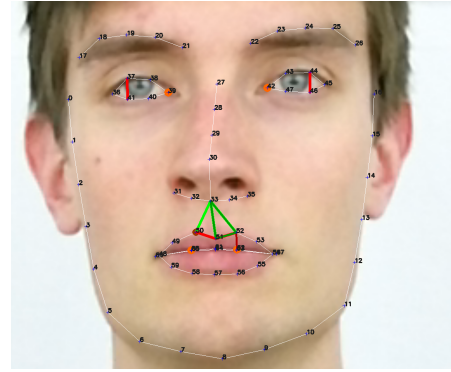
(a) Closing the eyes gently (9 features)



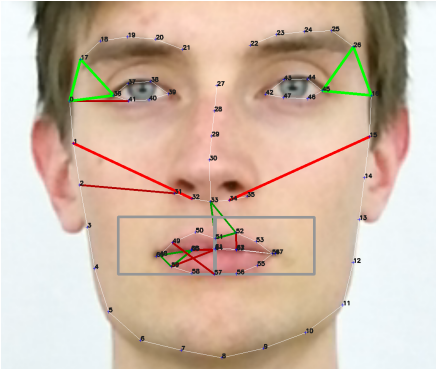
(b) Closing the eyes firmly (23 features)



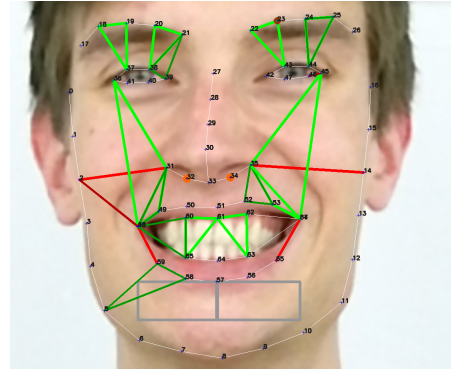
(c) Raising the eyebrows (7 features)



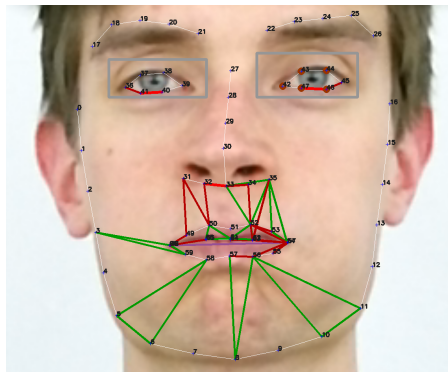
(d) Frowning (8 features)



(e) Pursing the lips (15 features)



(f) Exposing the teeth (21 features)



(g) Puffing the cheeks (50 features)

	asymmetric	symmetric
distance		
triangle		
subtraction		
motion		
Sjögreen		

**Figure 3.6:** Visualization of the 133 most frequently selected facial features for diagnosing FSHD per task.

### 3.2 Comparison expert FSHD predictions

This section lists the FSHD classification performance results of the three human experts and the developed systems that performed best. The reported scores by the human experts can be found in Appendix B. The best systems per model were the systems with the highest accuracies. The following list gives the system experts with their identification codes:

**System expert SVC (SVC)** Support Vector Classifier

Parameters: {kernel: rbf, C: 1, gamma: 0.01, probability: true}, only triangles, 2D & 3D, Random Logistic Regression with C=20, Random seed = 1

**System Expert KNN (KNN)** K-Nearest Neighbors Classifier

Parameters: {n\_neighbors: 5, metric: minkowski, weights: uniform, algorithm: auto}, all, 2D & 3D, Random Logistic Regression with C=20, Random seed = 1

**System Expert LR (LR)** Logistic Regression Classifier

Parameters: {C: 0.5, solver: newton-cg}, only triangles, 2D & 3D, Random Logistic Regression with C=20, Random seed = 1

**System Expert RF (RF)** Random Forest Classifier

Parameters: {n\_estimators: 40, bootstrap: true, criterion: gini}, all, 2D & 3D, Random Logistic Regression with C=3, Random seed = 2

Expert	Ex1	Ex2	Ex3	SVC	KNN	LR	RF
Accuracy	0.667	0.781	0.759	1.000	1.000	1.000	0.887

**Table 3.1:** Accuracies of FSHD predictions for each expert when compared to the FSHD ground truth.

Table 3.2 shows the Barnard’s exact test results for comparing the human experts and the systems experts with each other and the ground truth on FSHD predictions for each participant ( $df = 1, N = 87$ ). Barnard’s exact test was conducted using a MatLab implementation [62]. The Barnard’s exact test revealed that expert 3 was the only expert to predict FSHD significantly different from the FSHD ground truth ( $p = 0.0013, n = 0.9201, w = 3.2946$ ). Furthermore, the Barnard’s exact tests revealed that expert 1 and expert 3 predicted FSHD significantly different from all other experts with  $p < 0.004$ .



	Ex1	Ex2	Ex3	SVC	KNN	LR	RF
FSHD	p=0.3861	p=0.1413	<b>p=0.0013*</b>	p=1.0000	p=1.0000	p=1.0000	p=0.3853
	w=0.6595	w=1.1264	w=3.2946	w=0	w=0	w=0	w=0.6697
	n=0.0201	n=0.7701	n=0.9201	n=0.4701	n=0.4701	n=0.4701	n=0.0701
Ex1		<b>p=0.0000*</b>	<b>p=0.0000*</b>	<b>p=0.0000*</b>	<b>p=0.0000*</b>	<b>p=0.0000*</b>	<b>p=0.0000*</b>
		w=6.6802	w=6.9553	w=6.6478	w=6.6478	w=6.6478	w=6.6815
		n=0.7201	n=0.5201	n=0.6901	n=0.6901	n=0.6901	n=0.7501
Ex2			<b>p=0.0033*</b>	p=0.3525	p=0.3525	p=0.3525	p=0.4281
			w=3.1416	w=0.7178	w=0.7178	w=0.7178	w=0.5588
			n=0.9501	n=0.0401	n=0.0401	n=0.0401	n=0.0401
Ex3				<b>p=0.0013*</b>	<b>p=0.0013*</b>	<b>p=0.0013*</b>	<b>p=0.0013*</b>
				w=3.2946	w=3.2946	w=3.2946	w=3.2946
				n=0.9201	n=0.9201	n=0.9201	n=0.9201
SVC					p=1.0000	p=1.0000	p=0.3853
					w=0	w=0	w=0.6697
					n=0.4701	n=0.4701	n=0.0701
KNN						p=1.0000	p=0.3853
						w=0	w=0.6697
						n=0.4701	n=0.0701
LR							p=0.3853
							w=0.6697
							n=0.0701

**Table 3.2:** Cross table with comparisons of the FSHD predictions from human experts, expert systems and the FSHD ground truth using Barnard’s exact test. The following statistics are reported: p-values (p), wald statistics (w) and nuisance parameters (n). The FSHD category represents the known FSHD ground truth. \* Indicates a significant difference on FSHD predictions between experts.

### 3.3 Correlation between FSHD and facial weakness

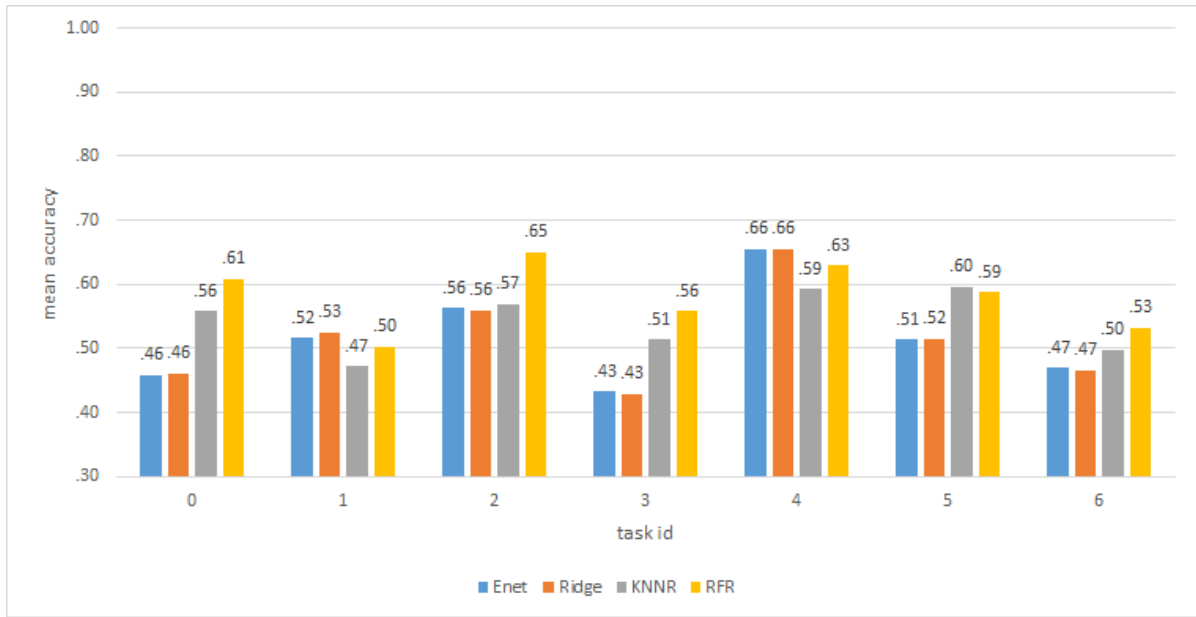
A Pearson product-moment correlation coefficient was computed to assess the relationship between the FSHD ground truth and the minimal median pooled expert facial weakness scores. There was a significant yet moderate negative correlation between FSHD and minimal median pooled expert facial weakness scores,  $r = -0.464, n = 87, p < 0.01$ . This result supports the idea that FSHD increases the chance of having a lower overall facial weakness score.

### 3.4 Facial weakness grading system evaluation results

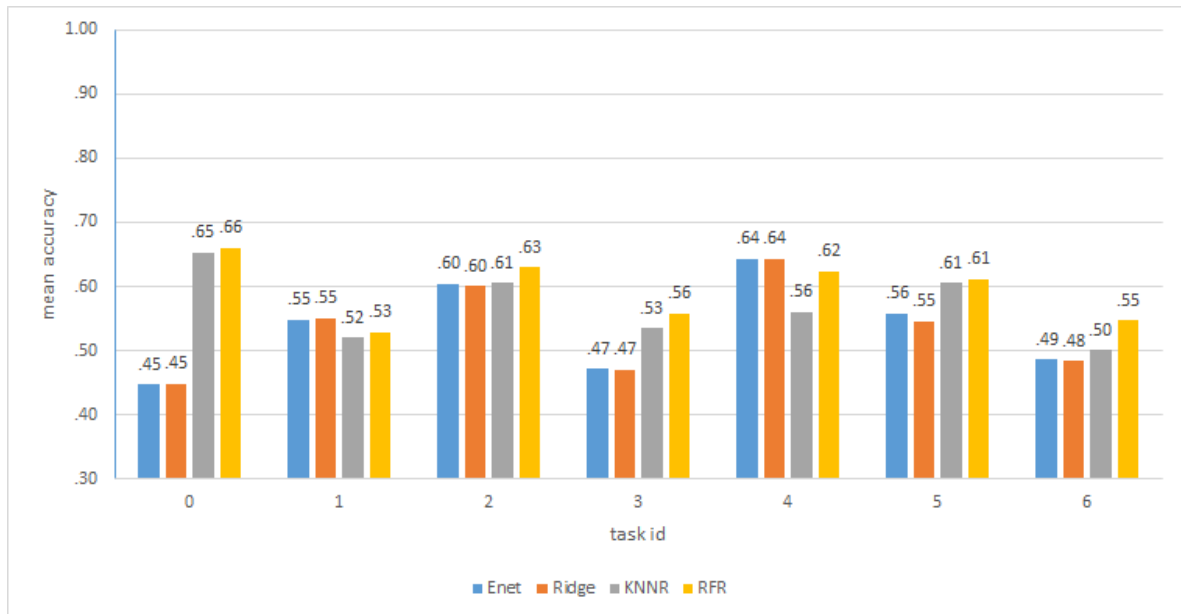
A 7 (Task) x 2 (Side) x 4 (Regression model) x 6 (Feature group) x 3 (Feature dimensions) x 4 (Feature selection settings) repeated measures MANOVA was conducted in the statistical software package SPSS with task, side and regression model as between-subject factors and feature group, feature dimensions and feature selection settings as within-subject factors. The levels of tasks were: closing the eyes gently (0), closing the eyes firmly (1), raising the eyebrows (2), frowning (3), pursing the lips (4), showing the teeth (5) and puffing the cheeks (6). The levels of the side factor were the right and left side of the face. The levels of regression model were: Ridge regression, ElasticNet regression (ENet), K-Nearest Neighbors regression (KNNR) and Random Forest Regression (RFR). The levels of feature group were: all, distances, motions,

triangles, subtractions and only asymmetries. The levels of feature dimensions were: 2D, 3D, 2D&3D and the levels of feature selection settings were: RL  $\alpha = .01$ , RL  $\alpha = .005$ , RL  $\alpha = .0025$  and RL  $\alpha = .0001$ . The dependent variable was classifier performance measured as the accuracy of predicting the weakness ground truth for the 87 participants. The task, side, regression model, feature group, feature dimensions and feature selection settings main effects and all their interactions were all found to be highly significant ( $p < 0.001$ ). However, the effect size varied greatly. For the main effects (task, feature group, feature dimensions, selection method) the effect sizes were great ( $\eta^2 > 0.900$ ), but for the side main effect the effect size was moderate ( $\eta^2 = .478$ ). Hence, all interactions involving side were mostly moderate to low in effect size ( $\eta^2 < 0.550$ ). Most interaction effects were also moderate to low in effect size, except for the feature group and feature selection method interaction ( $\eta^2 = .743$ ), the feature group, feature selection method and feature dimensions interaction ( $\eta^2 = .684$ ), the feature dimensions and task interaction ( $\eta^2 = .700$ ), the feature group and task interaction ( $\eta^2 = .740$ ) and the task and regression model interaction ( $\eta^2 = .918$ ). The listing of the analysis output can be found in Appendix C.3.

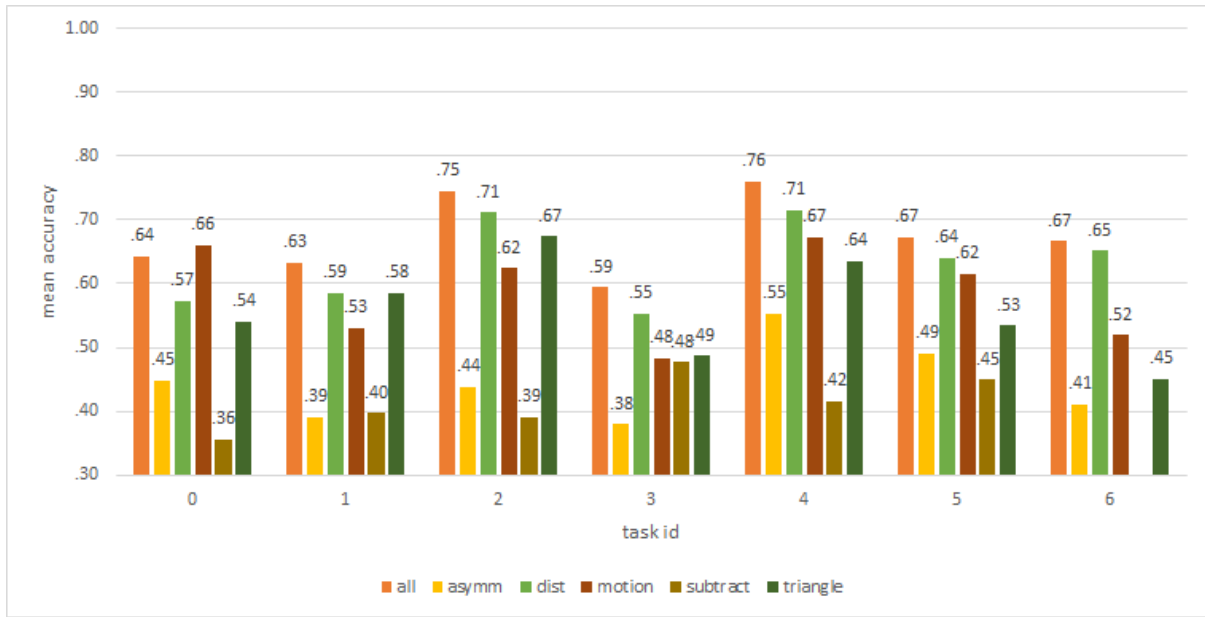
Post hoc pairwise tests were performed with Bonferroni correction for each level within all factors. The majority of the tests reveal high significance between levels, except between the frowning (task 3) and the puffing of the cheeks (task 6) tasks and between the Ridge and ElasticNet regression models. Figure 3.7 and Figure 3.8 show the system model performance between tasks and regression models for respectively the right and the left side of the face. A significance difference between sides was found (where the left side dominates the right side on performance) and also for all its interactions with other factors. However, in general the effect is not that strong. In general RFR and KNNR outperform ENet and Ridge models. However, for closing the eyes firmly (task 1) and for pursing the lips (task 4) it is the other way around. Finally, pursing the lips (task 4), raising the eyebrows (task 2) and showing the teeth (5) produced better systems on average than the other tasks. Figure 3.9 shows the system performance per different selected feature group and per task. It can be observed that selecting from all features produced the best systems, closely followed by distance features, then motion and triangle features. Selecting from only asymmetry features or from subtraction features produced the worst performing systems on average. Figure 3.10 shows the system performance per different selected dimensions and per task. Selecting only from 3D features produced the worst systems, followed by selecting only from 2D features and selecting from both 2D and 3D features performed best. Figure 3.11 shows the different feature selection methods, and clearly shows that an alpha parameter of  $\alpha = .005$  produces the best performing systems on average.



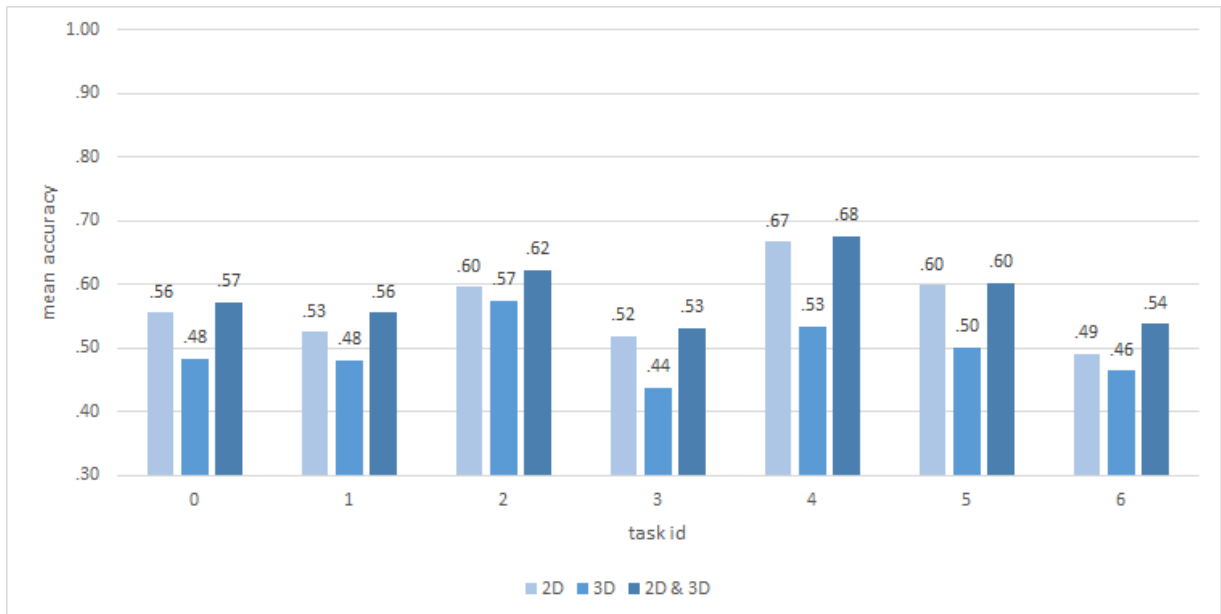
**Figure 3.7:** System facial weakness classification performance between tasks for the right side of the face. Note that the y-axis starts near chance level (0.333).



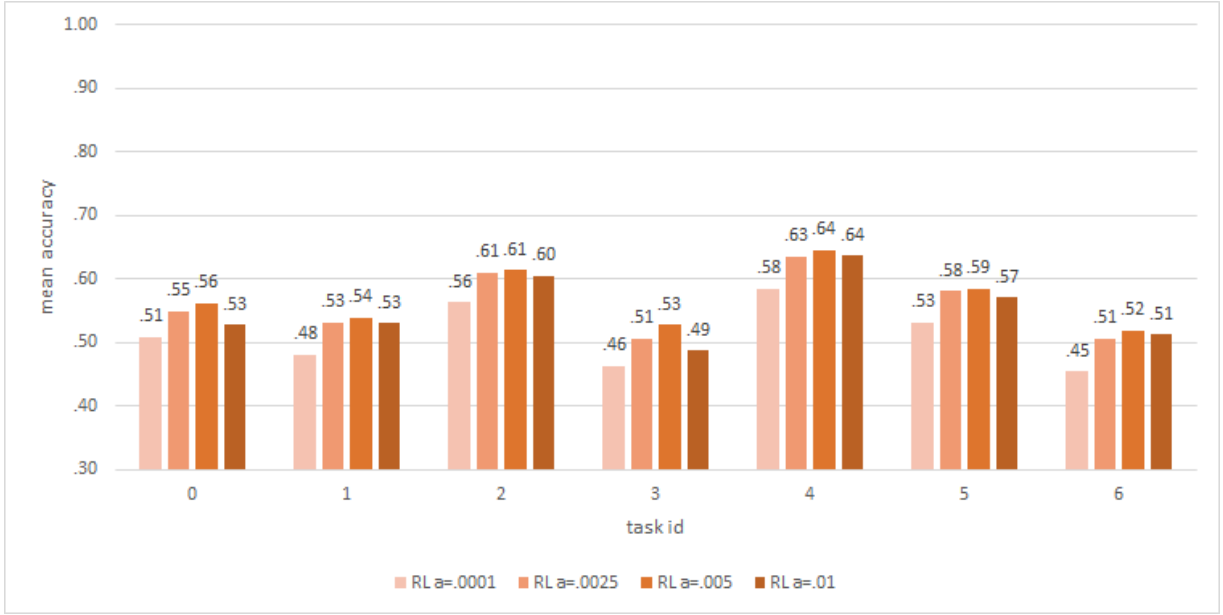
**Figure 3.8:** System facial weakness classification performance between tasks for the left side of the face. Note that the y-axis starts near chance level (0.333).



**Figure 3.9:** System facial weakness classification performance between tasks for each feature group. Note that the y-axis starts near chance level (0.333).



**Figure 3.10:** System facial weakness classification performance between tasks for the feature dimension groups. Note that the y-axis starts near chance level (0.333).



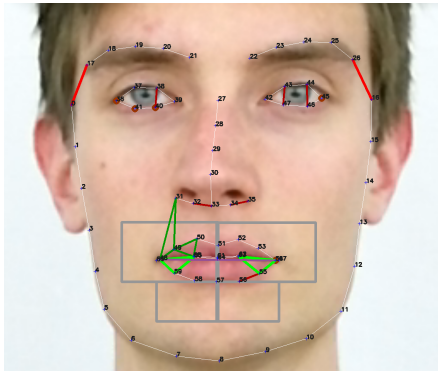
**Figure 3.11:** System facial weakness classification performance between tasks for the feature selection settings. Note that the y-axis starts near chance level (0.333).

### 3.4.1 Selected features

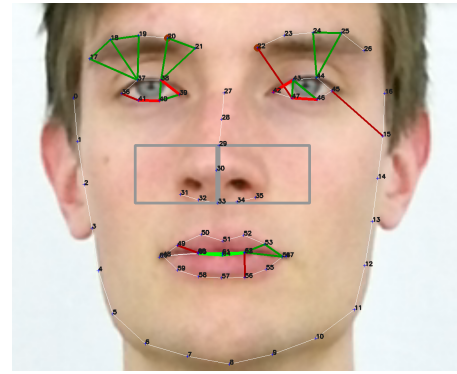
Table 3.3 shows the number of selected features frequencies for predicting weakness scores per task and side taken over all of the iterations for the all-features condition. For each task and side the 20 most frequently chosen features were selected and visualized for further interpretation. See Figure 3.12 and Figure 3.13 for a visualization of those features for respectively the right and left side of the face. For a full listing of the selected features for the right and left side of the face, see Appendix C.6.

task id	task	#features R	#features L
0	closing the eyes gently	208	199
1	closing the eyes firmly	210	219
2	raising the eyebrows	224	203
3	frowning	203	201
4	pursing the lips	186	167
5	showing the teeth	154	165
6	puffing the cheeks	206	199

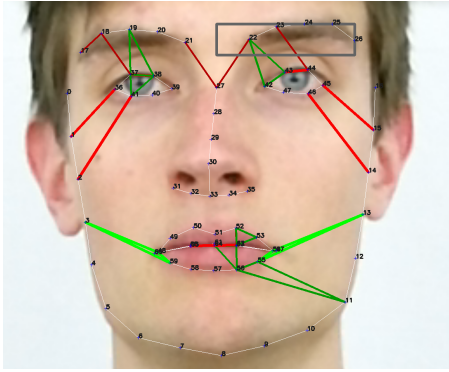
**Table 3.3:** Number of uniquely selected features for predicting the weakness score per task and side.



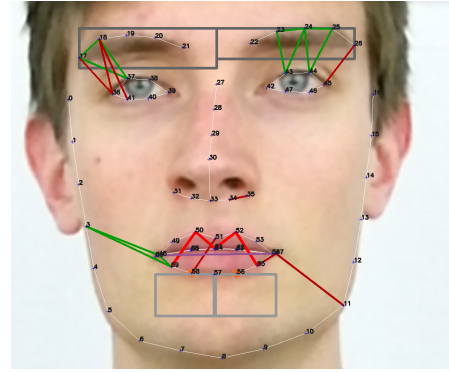
(a) Closing the eyes gently (20 features)



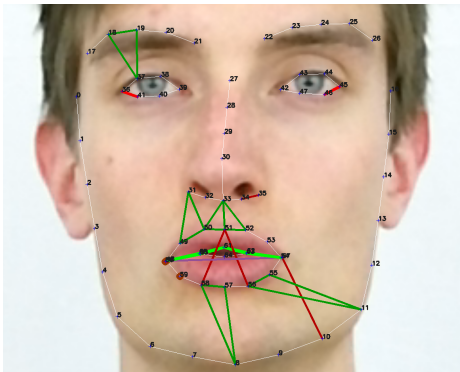
(b) Closing the eyes firmly (20 features)



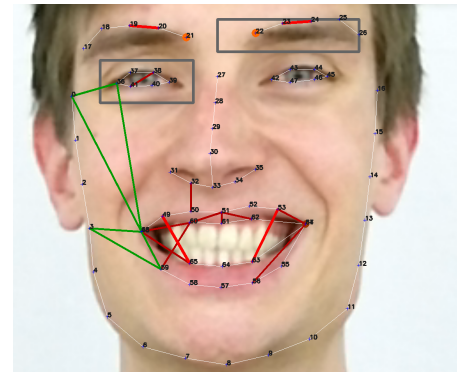
(c) Raising the eyebrows (20 features)



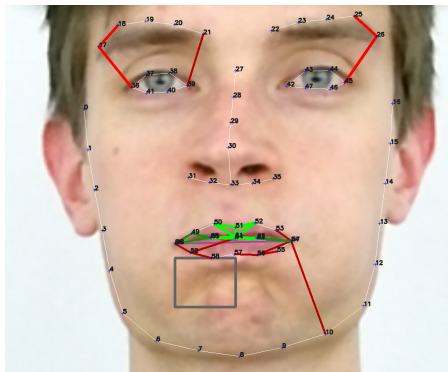
(d) Frowning (20 features)



(e) Pursing the lips (20 features)



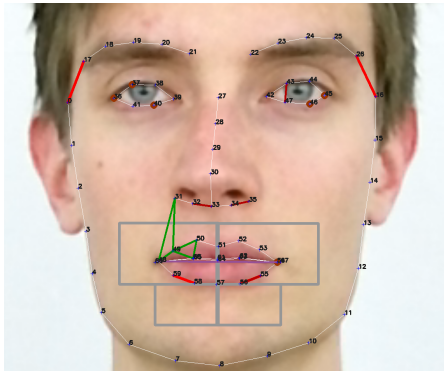
(f) Exposing the teeth (20 features)



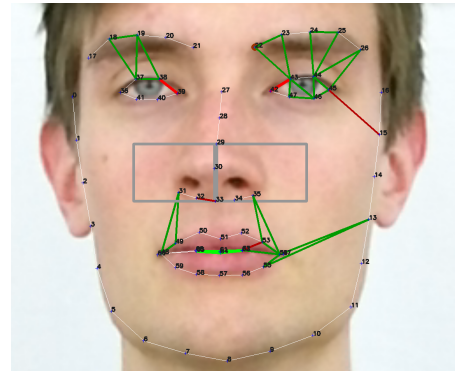
(g) Puffing the cheeks (20 features)

	asymmetric	symmetric
distance		
triangle		
subtraction		
motion		
Sjögreen		

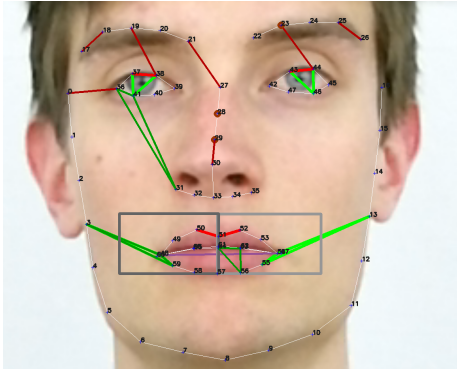
**Figure 3.12:** Visualization of the 20 most frequently selected facial features for estimating facial weakness per task for the right side of the face.



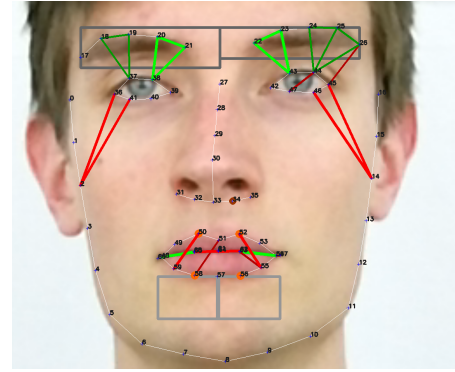
(a) Closing the eyes gently (20 features)



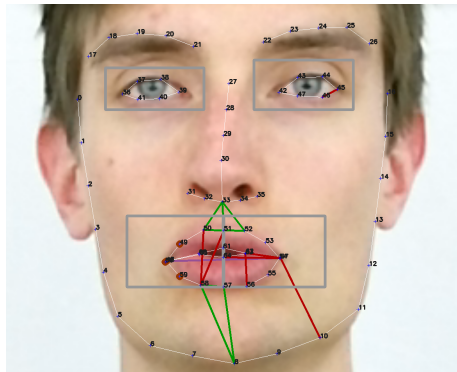
(b) Closing the eyes firmly (20 features)



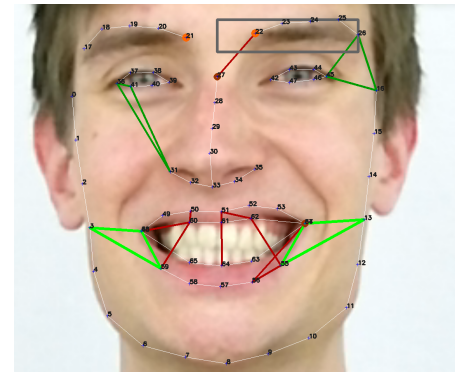
(c) Raising the eyebrows (20 features)



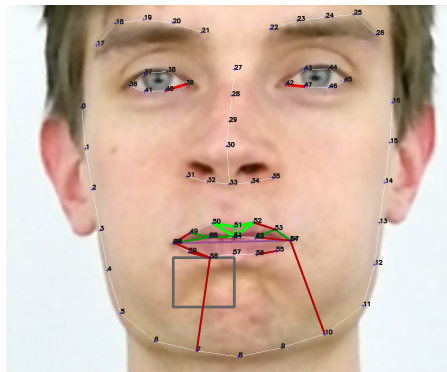
(d) Frowning (20 features)



(e) Pursing the lips (20 features)



(f) Exposing the teeth (20 features)



(g) Puffing the cheeks (20 features)

	asymmetric	symmetric
distance		
triangle		
subtraction		
motion		
Sjögreen		

**Figure 3.13:** Visualization of the 20 most frequently selected facial features for estimating facial weakness per task for the left side of the face.



### 3.5 Comparison expert facial weakness scores

	0		1		2		3		4		5		6	
	L	R	L	R	L	R	L	R	L	R	L	R	L	R
Ex1-Ex2-Ex3	-0.20	-0.18	0.63	0.62	0.50	0.57	0.49	0.41	0.75	0.73	0.53	0.60	0.69	0.70
Ex1-Ex2	-0.68	-0.72	0.59	0.58	0.38	0.48	0.42	0.31	0.79	0.75	0.47	0.61	0.66	0.65
Ex1-Ex3	0.64	0.77	0.78	0.81	0.56	0.56	0.48	0.44	0.76	0.77	0.51	0.56	0.70	0.69
Ex2-Ex3	-0.69	-0.68	0.53	0.48	0.55	0.69	0.57	0.45	0.71	0.67	0.63	0.64	0.71	0.76
Ex1-GT	0.93	0.88	0.94	0.97	0.70	0.65	0.64	0.65	0.92	0.91	0.65	0.76	0.80	0.79
Ex2-GT	-0.64	-0.66	0.66	0.63	0.71	0.83	0.76	0.66	0.86	0.83	0.81	0.84	0.85	0.87
Ex3-GT	0.73	0.90	0.84	0.84	0.85	0.88	0.83	0.83	0.86	0.87	0.84	0.81	0.90	0.92
S1-S2-S3-S4	0.64	0.73	0.74	0.67	0.78	0.70	0.63	0.61	0.86	0.86	0.65	0.63	0.80	0.89
S1-GT	0.70	0.77	0.83	0.89	0.92	0.91	0.75	0.72	0.94	0.94	0.84	0.79	0.88	0.91
S2-GT	0.65	0.80	0.83	0.83	0.92	0.91	0.75	0.75	0.93	0.94	0.84	0.77	0.87	0.89
S3-GT	0.75	0.86	0.85	0.73	0.88	0.79	0.76	0.77	0.89	0.90	0.74	0.79	0.85	0.89
S4-GT	0.75	0.80	0.76	0.66	0.77	0.71	0.67	0.76	0.87	0.85	0.63	0.68	0.79	0.84

**Table 3.4:** Krippendorff’s Alpha Reliability Estimates between expert groups and the weakness ground truth scores. The columns represent the task and side for each reliability estimate. The tasks are listed by their task identification number.

In this section the facial weakness classification performance results of the three human experts (Ex1, Ex2, Ex3) and the developed systems that performed best are compared on agreement. The reported scores by the human experts can be found in Appendix B. The best systems were separately trained per model, task and side and were obtained by taking the systems for that category with the highest accuracy. The best systems with their parameters per task, side and model are listed in Appendix C.4.

Krippendorff’s Alpha Reliability Estimate has been calculated using the Kalpha SPSS implementation [63]. Krippendorff’s Alpha measures inter-rater reliability where an alpha value of 1.0 implies exact agreement between observers and where alpha values near 0.0 or below zero indicate respectively grading agreement obtained at chance level and grading agreement below chance level.

Krippendorff’s Alpha has been calculated for all 7 tasks and 2 sides of the face while comparing different expert groups: all three human experts (Ex1, Ex2, Ex3), all four system experts (Ridge Regression, ElasticNet, KNN-Regression, Random Forest Regression) (S1, S2, S3, S4), the human experts pair-wise (Ex1-Ex2, Ex1-Ex3, Ex2-Ex3) and the experts pair-wise with the training ground truth (Ex1-GT, Ex2-GT, Ex3-GT, S1-GT, S2-GT, S3-GT, S4-GT). The units for all comparisons equal the number of participants. The statistics were calculated for ordinal data using 5000 bootstrap samples. The exact statistics are listed in Appendix C.5. Table 3.4 gives a more comprehensive overview of the calculated reliability estimates.

In general, the agreement on grading the left and right sides did not vary more than a maximum of  $\pm 0.15$ , which suggests a difference but roughly similar agreement on grading between experts for each side of the face. Furthermore, the trained computer systems tend to be in higher agreement with each other than the agreement among human experts. Overall, the human experts are in moderate to high agreement with the ground truth estimate. However, on closing the eyes gently one expert assigned the same grade to each participant, what resulted in below chance agreement with the other human raters and hence also the ground truth. This finding reflects the property of the facial weakness ground truth (median of all expert scores)



to deal with outliers. The computer systems are all in moderate to high agreement with the ground truth ( $\alpha > 0.63$ ).

There was a very low agreement between the human experts on the closing the eyes gently (task 0) for both the right and the left side of the face, what can be attributed to deviating scores from human expert 2 with respect to other experts. Expert 2 in this case graded all scores of 2 on the 4 point scale for each participant for this task, while the other experts had moderate to high agreement ( $\alpha = 0.64$  and  $\alpha = 0.77$ ) with each other on this task. Pursing the lips (task 4) and Puffing the cheeks (task 6) were the tasks for which the experts reached the highest agreement, which were also the tasks for which the computer systems showed the highest agreement between each other and between the ground truth. Closing the eyes firmly (task 1) was also graded with moderate to high agreement. The other tasks: raising the eyebrows, frowning and showing the teeth were graded with low to moderate agreement.

## Chapter 4

# Discussion

Let us start by repeating the research question: *“is it possible to develop a computerized markerless facial weakness grading system that gives an objective measure for facial weakness for FSHD patients which has comparable performance to the current golden standard?”*.

In light of the results from section 3, the research question can be confirmed, since facial weakness grading systems have been developed with comparable performance to the current golden standard. Computerized diagnosis of FSHD from the extracted facial features can be performed by the developed systems above human expert-level, which was shown by the comparison of the expert FSHD predictions from section 3.2. Also, the results from section 3.5 show that the developed facial weakness grading systems have high agreement with the facial weakness ground truth, which is based on the current golden standard.

However, the number of evaluated participants was rather small (87 participants). Hence, the trained models might be limited in their ability to generalize to other and larger datasets. Also, the median between expert predictions was taken as the facial weakness ground truth, but on most tasks the experts were found to only moderately agree with each other. Hence, more effort should be spent to obtain a higher agreement among experts in order to obtain a more reliable standard for facial weakness.

### 4.1 Expert FSHD predictions

The FSHD grading system performance results suggest that diagnosing FSHD from the extracted features is possible with nearly perfect accuracy. The best trained FSHD grading system experts (KNN, LR, RF and SVC) all perform much better than the three human experts on predicting whether a participant has FSHD.

The comparisons between experts showed that one human expert scored significantly different from the ground truth. This finding is thought to be related to the grading approach of the expert: i.e. the expert only predicted FSHD for participants that were clear cases and preferred to predict no FSHD on the border cases. It was also found that each human expert predicted significantly different from the other human experts, which indicates significant disagreement between human experts on when to diagnose FSHD. However, this is of little clinical relevance, since DNA tests can be performed to provide a clear diagnosis. But, if easy to use FSHD grading systems would become common, they could aid human experts with a diagnosis.

Only one expert did not score significantly different from the best trained computer experts. It appears that this resulted from the fact that all experts had experience with grading facial weakness, but only that particular expert had also much experience with FSHD.

## 4.2 Weakness grading

The results from section 3.3 reveal a clear yet moderate correlation between having FSHD and the pooled minimum facial weakness ground truth. This correlation basically indicates that if FSHD is present, there is a fair chance of experts also assigning some degree of facial weakness for any of the tasks. This finding is in line with the expectations, except that the correlation is not that strong. The moderate effect might be explained by the varying involvement of facial weakness within FSHD patients. Another explanation might be that the pooled minimum facial weakness ground truth is not representative as an overall facial weakness for a participant. This might be the case, since it is not directly assigned by the human experts, but rather calculated from the facial gradings of the human experts per task.

For the comparison between experts from section 3.5, the greatest concern is the moderate to weak agreement between the human experts on how to grade facial weakness. According to the literature an agreement of at least 0.80 should be achieved among experts [64] to ensure the derivation of a reliable ground truth from those scores. However, the agreement scores were found to be significantly lower than that. Even if we assume a reasonable lower agreement boundary  $> 0.70$ , then there was only reasonable agreement found for pursing the lips and for puffing the cheeks and not for the other tasks. Exceptionally bad agreement (agreement below chance level) was found on the first task (closing the eyes gently), because one expert gave all participants for that task a grade of 2, which lead to high disagreement with the other human experts and the ground truth. The latter finding vouches for using the median to calculate the facial weakness ground truth, since it can deal with such outliers.

On most tasks high agreement between the ground truth and the human experts was found, which also supports the use of the median for calculating the ground truth. The fact that not all tasks were found to be in high agreement can be attributed to the disagreement among the human experts.

The expert comparison further revealed that the system experts were in moderate to high overall agreement with each other and were also individually in reasonable to high overall agreement with the ground truth. These findings support the taken approach for obtaining and training of the facial weakness systems.

The systems currently selected as the best for grading facial weakness were selected based on accuracy. However, accuracy does not tell how much the outcomes differ from the ground truth. For example two systems with similar accuracy could both predict wrong for a specific participant. Assume that the ground truth was found to be 1 and the first system graded a 2 and the second system assigned a 3. In such a case the first system, which is closer to the ground truth, should be preferred. However, accuracy does not capture this difference treating both cases similar, thus the inferior system has equal probability of being selected. Hence, the selection criteria could be changed to a measure like Krippendorff’s Alpha [63] to measure the ordinal agreement between the ground truth and the system prediction, which does take such differences into account.

A big problem with the current results for training the weakness grading systems is the lack of participants in combination with the class imbalance problem. The participants with severe facial weakness are the minority class, which means that they are less present within the collected data. To deal with this problem, the original facial weakness 4-point scale has been reduced to a 3-point scale, merging the two classes with the highest degrees of facial weakness, and the minority classes were oversampled (using duplication) for the training folds. In order to increase the sensitivity of the facial grading (i.e. in order to add more points to the facial grading scale) much more data should be collected. Furthermore, sampling techniques like

SMOTE [65] could be used to replace or improve the oversampling method.

Another point of discussion is to what extent the proposed method in this thesis provides an objective measure. For training the grading systems, subjective expert gradings were used. Hence, the objectivity of the golden standard and the obtained system can be questioned. The trained systems are by nature objective, since they are programs that once trained will always yield the same prediction for a certain input. However, since the ground truth for training facial weakness is subjective in nature, it seems that the classifiers trained on such a ground truth will objectively predict a subjective measure. Hence, researchers have to reach consensus about how to grade facial weakness and to establish acceptable standards for facial weakness grading systems (and humans).

## 4.3 Evaluation of features

The system evaluation pipeline from section 2.8 in combination with the labeling system from section 2.7.1 provide a set of tools to identify and examine what features are relevant for both classification tasks. Initial results using the system evaluation pipeline show the potential for investigating the relevance of features.

### 4.3.1 FSHD features

The results from section 3.1 reveal the general features that work and do not work when diagnosing FSHD, what models are best suited for the classification task and which tasks are relevant for diagnosing FSHD.

The best models for predicting FSHD are the LR and SVC models, but the KNN and RF models perform good as well. Advantages over the LR and SVC models are also that these methods weigh the features using coefficients, so the influence of individual features can be examined more easily by examining the learned model parameters. The best systems were found by making a selection from all extracted features including both 2D and 3D features, although using only 3D features outperforms using only 2D features and using only distance, triangle or asymmetry features also works very well. It seems that using 3D data improves classification accuracy for diagnosing FSHD. Finally, for Randomized Logistic Regression with  $C = 10$  were found to be the best settings for the feature selection component within the system evaluation pipeline. This finding is not that strange, since the L1-SVC method is known to only select a single feature from a group of highly correlated features. This poses a problem, since two highly correlated features can both be very informative with respect to the class labels. This problem is solved by the stability selection method, which uses randomization techniques to reestimate the selected features multiple times and selecting the most frequently selected features.

From the actual selected features which are assumed to be the most relevant for diagnosing FSHD, the majority of the features were related to puffing the cheeks, closing the eyes firmly, exposing the teeth and pursing the lips. The finding that the most relevant features coincide with these tasks, is in line with the expectations, since these tasks have the most clear visual facial deformations with clear differences between FSHD and controls. More surprising was the finding that features around the mouth were also very predictive for tasks related to the eyes and that features around the eyes were also very predictive for tasks related to the mouth. There are two likely explanations for this: first of all it is possible that it is just a coincidental result because the results are based on a relative small number of participants. However, another more interesting interpretation is that the found features capture compensation mechanisms

of FSHD patients. With compensation mechanism we mean muscular contractions of muscle groups unrelated to the task in order to compensate for the lack of muscular strength in task-related muscle groups. E.g. an FSHD with severe difficulty of showing the teeth might also more likely squint their eyes in order to successfully perform the task.

### 4.3.2 Facial weakness features

The results from section 3.4 give more insight into the features related to facial weakness. It appears that the ENet and Ridge regressors do not perform significantly different from each other and that the kNNR and the RFR perform the best overall. The fact that ENet and Ridge perform similarly is easily understood, since both rely on the same mechanics where the latter is a special case of the former. The difference between the two is the way the regularization terms are weighed, Ridge uses only quadratic regularization term and ENet additionally uses a normal regularization term. On some tasks kNNR and RFR perform very well or even better than kNNR or RFR, in such cases those models should be preferred, since they are more easily understood in terms of coefficient weighing. Best overall performance was found for raising the eyebrows, pursing the lips and showing the teeth. A significant yet moderate difference was found between sides of the face, probably because the extracted features were in general more descriptive for left sided facial weakness within the participants and is thought to be a bias within the evaluated participants. Increasing the number of participants should gradually factor out this difference. For features selection, Randomized Lasso and  $\alpha = .005$  gave the highest overall accuracy within grading systems.

Selection from all feature groups performed best overall, followed by: distance features, motion features, triangle features, asymmetry features and finally subtraction features respectively. Note that the order of importance for the feature groups was found to be different to the order for diagnosing FSHD. The most notable feature group differences between grading facial weakness with respect to diagnosing FSHD were the finding that motion features performed much better and the finding that asymmetry features performed much worse. The fact that motion features performed much better for grading facial weakness might just be because of the different nature of the two classification tasks, where facial weakness is more expressed in local displacements captured by the motion features. The fact that selecting only from asymmetry features performs that much worse on facial weakness grading was unexpected, since it seems logical that in order to grade the facial weakness on one side of the face the contra-lateral side of the face should be used as a reference. The cause of this finding is currently not well understood.

Selecting from both 2D and 3D features yields the best overall performance, followed by: selecting only from 2D features and selecting only from 3D features. Of interest here is the observation that selecting from 3D measures performs worse than selecting only from 2D measures. This finding could be explained by measurement error from the Kinect 2 sensor or the possibility that the 2D features are coincidentally more correlated with the class labels, due to an annotation bias. Still, using both 2D and 3D gave the best performance, hence adding 3D features improves grading facial weakness.

Looking at the visualization of the most selected facial features, there does not seem to be a clear focus on either the left or right side of the face. Furthermore, similar to the most selected features for diagnosing FSHD, tasks related to the mouth also involve features of the eye and tasks related to the mouth also involve features of the eyes. This also indicates that compensation mechanism might be captured by the features and used for predicting facial weakness. The finding that the selected facial features do not have a clear preference for either

side of the face might be due to the fact that both sides of the face need to be evaluated in order to assess an asymmetry. This is further supported by the reported number of asymmetry features among the most selected features, despite the earlier finding that they are not that informative on their own. For closing the eyes, many features around the eyes capture the disability of the participant with facial weakness to completely close the eyes. For exposing the teeth, the finding that triangles are selected near the mouth corners and connected to the face contour, seem to capture the characteristic inability of FSHD patients to raise them. For the other tasks, the feature relations with FSHD and facial weakness are not that clear. For future work, increasing the number of participants might help to expose such relations for the other tasks.

### 4.3.3 Subtraction and motion features

Both subtraction and motion features do not perform that well on both FSHD diagnosis and facial weakness prediction, with the exception of motion features for weakness prediction. This was to be expected, since participants tend to move their heads (mostly up and down) during tasks, what greatly influences the resulting subtraction and motion feature values. Although an image stabilization technique has been implemented similarly to the work of He et al. [27] to compensate for these rigid motions, head fixation techniques should greatly improve the reliability of these features [39]. Hence, if subtraction and motion features are to be reliability acquired, it is recommended to focus the acquisition method on acquiring such features.

## 4.4 Methodological limitations

### 4.4.1 Landmark and timeframe annotation

A big drawback of the current approach is the tedious procedure of first identifying timeframes of the participant at rest and at maximal expression from videos and then to manually label 68 landmarks within the two timeframes for each task. Careful labeling all 7 tasks for a participant takes approximately 40 – 80 minutes, even after pre-fitting the landmarks on the faces using DRMF [52]. Not only does the procedure take a lot of time, but it also introduces an additional bias by having a human observer annotate the landmarks and timeframes.

An obvious solution to remove the burden from the experimenter and to reduce the human observer subjectivity, is to use or develop a computer program to automatically identify the timeframes and perform facial landmark localization. To identify the relevant timeframes from a video stream, some form of feature extraction over the video stream of a participant performing a task is required, i.e. a feature profile over time. In general, landmark localization methods can be applied on a video stream of a participant performing a task either during recording or during playback, what results in velocity profiles of landmarks over time. From the velocity profile it should be possible to determine the location of the time frames at maximal expression and at rest for each task, using for example methods as described by e.g. [19, 27]. Hence, the core mechanic for automating timeframe identification is the creation of feature profiles over time, which can be accomplished using landmark localization.

Besides facial landmark localization (also known as face alignment), there is also facial landmark tracking. Facial landmark localization takes as input an image and tries to precisely fit a predefined set of facial landmarks on that image. Facial landmark tracking takes an image and a predefined set of marker positions and updates the positions of the markers given the image. Landmark localization is usually more computationally expensive than landmark

tracking and hence landmark fitting is often only performed on the first frame(s) of video and succeeded by landmark tracking methods for the other video frames.

One of the simplest and most common approaches for facial landmark tracking used within facial paralysis research, is the use of physical markers positioned at the landmark locations on the face of the participant. Physical markers with reflective properties can be used to facilitate easy facial landmark tracking by giving the markers a distinct color from the face. Using color intensity thresholding methods, it is possible to extract and track the landmark positions from video, which are relatively easy to implement. On the downside it might be tedious and difficult to consequently position many facial markers on the correct facial locations on the face, introducing a positioning bias. Also, depending on the materials, physical markers might hinder natural facial expressions and put physical constraints on the positioning and number of markers that can be used, e.g. it is not feasible to position physical markers on the pupils of a participant and when 90 markers are used around the mouth it becomes difficult to uniquely track those and recognize each separate marker.

Reliable facial landmark tracking without physical markers is technically a lot harder to accomplish, since the ground truth for the landmark positions has to be learned from data or has to be modeled beforehand. Both are difficult to accomplish, since there are virtually no FSHD facial databases with landmark annotations available and the effect of FSHD on the facial muscles is highly variable.

The use of existing software that is not specifically created for the task can be problematic, which the initial pilot experiments with software like FaceReader [28] showed. FaceReader is simply not developed nor suited for tracking faces for people with FSHD, i.e. it has too little useful landmarks for the eyes, the tracking is not reliable enough and the underlying model has probably been trained on mostly symmetric faces, which results in improper tracking of asymmetric faces. There are however many other face tracking software libraries and methods available that seem more suitable. A general problem linked to all of these methods is that they need to be trained and validated on FSHD patients before finding a place in an automated marker-free grading system for facial weakness for FSHD. The usage of 3D data like depth stream from the Kinect sensor can improve the tracking significantly. A modern landmark localization approach for facial therapy applications using a Kinect was reported by Lanz et al. [66], using 3D tracking of facial landmark positions, which were reported to be invariant to facial expression.

Feature profiles over time are key to successful automatic timeframe annotation and hence should be incorporated in a fully automated grading system. An additional benefit of using feature profiles over time is that those could potentially be informative for predicting FSHD or for grading muscular weakness and hence these profiles could be added as additional features to indicate progress over time. This can be beneficial, since the features from the current approach lack motion based features describing the transition between rest and maximal expression. For example, the trembling of muscles and characteristic contraction of muscles during state transition are currently missed.

#### 4.4.2 Data acquisition

Only 87 participants could be used at the time for training and evaluation of the grading systems so care should be taken before generalizing the results to the entire population of people with FSHD. For this particular set of participants good results were obtained, but it should be investigated if the results do also scale with the number of cases. The currently identified relevant facial features might still include artifacts introduced by for example manual

labeling, which are selected because they are very correlated with facial weakness or FSHD. With many more participants these artifacts should slowly be factored out so only the relevant features to facial weakness and FSHD will remain. It is to be expected that the accuracy of the FSHD diagnosis systems will go down a bit, since it is overfitted on the current dataset. Furthermore, the system performance for predicting the facial weakness should improve, since more participants would introduce more minority class samples and as such make it easier to deal with the consequences of the class imbalance problem.

Because of the time consuming nature of the labeling, only 91 cases have been labeled by hand (from which 4 cases have been excluded because of indecisive FSHD DNA results) for a single iteration for the analyses within this thesis. Yet a lot of additional participants have been recorded using the same setup and hence more data can be labeled to validate the method on a larger group. This data includes two additional unlabeled intra-participant iterations that could potentially be used to improve classifier stability and also allow for testing intra-participant system performance.

## **Auditory cues**

The use of simple audio cues for synchronization of multiple videos and automatic identification of the tasks within video streams works very well and provides a convenient way for semi-automated video cutting. However, it was found that sometimes participants did not perform the tasks exactly after the audio cue onset and started somewhat too early or too late, which made the audio cues fall out of sync with the actual performance of the tasks. For the presented work this implied more manual searching for the position of the timeframes at rest and at maximal expression for cut participant videos, since the rest timeframe would not be at the beginning of the video. The problem is thought to be solvable by instructing the participants more explicitly to only start performing the task just after the signal has stopped playing.

It is very important during recording that a participant keeps looking at the camera while performing the tasks and also directly after performing the tasks. That is they should keep looking at the camera: before, during and *after* performing a task, until all three iterations of a task have been recorded and the participant assumed the resting facial expression. However, in the participant-database of recorded videos many participants would immediately look towards the instructor after the last task was performed and as such they did not assume their natural rest state on the final iteration. Therefore, additional emphasis should be placed on instructing participants to keep looking at the camera in front of them until the signal is given to relax.

Proper instructions are key to obtaining better task videos where participants keep looking at the camera. More auditory cues could be used, besides the one at task onset: one for the participant to indicate moving back to rest state after assuming maximal expression for a while, and one for the participant to know when to relax (release their focus from the camera) after three tasks have been performed in succession.

## **Cameras**

The current approach relies on the data captured with the Kinect 2 camera and does not use the videos from the other three PowerShot cameras. Hence, removing the Powershot cameras provides a simple way to reduce the complexity of the setup. The choice for the Kinect 2 over the PowerShot cameras is simply the addition of depth information, which has been shown to improve classification performance. The Kinect 2 was not designed for only capturing facial depth data and there is better recording equipment for that task specifically. However, the



advantage of the Kinect 2 and the PowerShot cameras are that they come at consumer level prices, which facilitates clinical adoption.

### **Lighting setup**

The necessity for the lighting equipment should be further investigated. The only features that currently are influenced by lighting are the subtraction features, which work with pixel illumination intensities. The subtraction features require even distribution of light intensity on the face and thus other lighting setups might be suitable as well, but the subtraction features were not selected very often by the feature selection. When no subtraction features are used, the current lighting setup might be unnecessary and simple room illumination might be sufficient. However, the current system lacks a working landmark localization method which is likely to require proper illumination to work [28].

### **Kinect video cutting**

Kinect 2 stream recordings are not really suited for cutting, i.e. there are no default cutting tools delivered with the SDK and the self-written cutting tool generates files that do not start from the audio cue. Hence, it is recommended for future work to use meta-data annotations for the audio cues and place those directly in the full-length raw Kinect recording instead of cutting the Kinect recording in task-related parts. This would help to keep the annotations synchronous with the other recorded videos. The developed cutting tools might still be useful to reduce the overall file size of the raw Kinect recordings, by removing the start frames and the end frames for which no tasks are recorded.

### **Head stabilization**

Participants would very often jerk their heads upwards when performing tasks like raising the eyebrows, or lower their heads when frowning, resulting in different orientation of the head between and within participants. The problem with these rigid head motions is that they can lead to crude 2D distance measurements. For example imagine that we measured the distance between the nose tip and the upper lip of a participant at rest as a distance from the pixels when facing the camera directly. When measuring the same distance again while the participant keeps his expression exactly the same but tilts his head down the measured 2D distance with respect to the camera becomes smaller. This is of course undesirable, since the true value of interest has in fact not changed between conditions and should therefore be the same. 3D measurements take into account the depth of the face and hence suffer less from this problem, i.e. the 2D distance (x and y dimensions) do also change, but the z dimension should change proportionally, so the measured distance remains the same. However, severe head motions (e.g. tilting the head completely up to tilting the head completely down) might occlude some facial attributes and hinder a depth camera from properly obtaining the depth values for those, in which case the values should be interpolated, which might result in error.

To perform proper 2D geometric measurements, the orientation of the face of the participant should be fixated with respect to the camera. This could be accomplished for example by fixating the head on a brace [36, 42] (restricting head movement), or by putting a camera enabled helmet on the participant (correct the camera for head movements) [34].

The image stabilization method from section 2.4.2 is able to correct some of the head rotations (roll of the head) and stationary movements into the y-x dimensions (translation) and z-dimension (uniform scaling), but does not deal well with pitch and yaw rotations since those

also change the perspective of the face. Furthermore, the method is limited to 2D images only and hence is unsuitable for stabilizing 3D head poses.

### **Landmark normalization**

Heads vary in shape and size. Hence, the obtained landmarks are not invariant over participants. Furthermore, the current approach does not employ a normalization step where the landmarks are scaled to a uniform shape and size for each head, before calculation of the features. This could be done for example by taking two or more reference landmarks and using those to calculate relative distances. It could be interesting to investigate if adding normalization can improve system performance.

### **4.4.3 Analyses**

The Repeated Measures MANOVA analyses used for identifying the relevant features and feature selection methods for both classifying FSHD and predicting facial weakness are currently the most suitable statistical analyses. However, certain assumptions like sphericity of the data are likely to be violated and should be corrected for. Hence, these assumptions should be taken in consideration for future work.

## **4.5 Future directions**

The created participant-database offers a lot of potential for future research. There are a lot of unused data from which additional features could be extracted. E.g. currently only the time-frames of the face at rest and at maximal expression are used for comparison, but the transitions between the two (from rest to extreme and from extreme to rest) could also be informative for the degree of facial weakness. For example, there are cases where FSHD patients show visible trembling of the muscles or have difficulty to maintain maximal expression. Extracting this information from the videos could possibly provide useful information to improve classification.

An important direction for developing a fully automated grading system for facial weakness is a reliable marker-less landmark localization system. There are many methods that claim high robustness, but most of the available methods rely on databases of healthy and symmetrical faces, which hinders using such methods for tracking subtle facial weakness and asymmetry. It is likely that if such a method will be employed, it requires additional calibration, which must be incorporated within the acquisition process.

A problem with the current approach is the lack of data for identifying the combined relevance of features for classification of facial weakness. A potentially future approach for collecting video data from FSHD patients is using a mobile phone application or a framework like ResearchKit for administering data collecting applications. Mobile applications have the advantage of being easy to distribute for researchers and can be used to reach a great variety of patients at their homes. Additionally, having the participants performing the data acquisition themselves saves the researchers time. A disadvantage of collecting data in this manner is that a lot of factors like head stabilization and lighting control are a lot harder to control than recording people at the clinic.

The presented facial grading systems should be sufficiently validated and should be made more usable for application in FSHD facial weakness research. The presented work provides ample of opportunities for improvement, extension and future research. Therefore, this work should be regarded as a starting point for developing a computerized standard for objectively grading facial weakness within FSHD.

## Chapter 5

# Conclusions

The first conclusion is that it is possible to develop a computerized marker-less facial weakness grading system that gives an objective measure for facial weakness for FSHD patients that has comparable performance to the current golden standard.

The second conclusion is that it is possible to develop computer systems that can diagnose FSHD from facial features above human expert level.

Furthermore, passing all sources of extracted facial features to the feature selection method and using stability selection, produces the best feature training sets for obtaining facial grading systems and FSHD diagnosis systems with the overall highest accuracies. The distance, triangle and motion features are selected the most for the facial weakness grading systems. The distance, triangle and asymmetry features are selected the most for the FSHD diagnosis systems. Combining 3D features with 2D features improves outcome facial weakness classification and FSHD diagnosis.

# References

- [1] J. C. Deenen, H. Arnts, S. M. van der Maarel, G. W. Padberg, J. J. Verschuuren, E. Bakker, S. S. Weinreich, A. L. Verbeek, and B. G. van Engelen, “Population-based incidence and prevalence of facioscapulohumeral dystrophy,” *Neurology*, vol. 83, no. 12, pp. 1056–1059, 2014.
- [2] G. Padberg, P. Lunt, M. Koch, and M. Fardeau, “Diagnostic criteria for facioscapulohumeral muscular dystrophy,” *Neuromuscular Disorders*, vol. 1, no. 4, pp. 231–234, 1991.
- [3] G. W. A. M. Padberg *et al.*, *Facioscapulohumeral disease*. Faculty of Medicine, Leiden University Medical Center (LUMC), Leiden University, 1982.
- [4] J. Gabriels, M.-C. Beckers, H. Ding, A. De Vriese, S. Plaisance, S. Van der Maarel, G. Padberg, R. Frants, J. Hewitt, D. Collen, *et al.*, “Nucleotide sequence of the partially deleted d4z4 locus in a patient with fshd identifies a putative gene within each 3.3 kb element,” *Gene*, vol. 236, no. 1, pp. 25–32, 1999.
- [5] L. Snider, L. N. Geng, R. J. Lemmers, M. Kyba, C. B. Ware, A. M. Nelson, R. Tawil, G. N. Filippova, S. M. van der Maarel, S. J. Tapscott, *et al.*, “Facioscapulohumeral dystrophy: incomplete suppression of a retrotransposed gene,” *PLoS genetics*, vol. 6, no. 10, p. e1001181, 2010.
- [6] V. Voet, G. Bleijenberg, J. Hendriks, I. de Groot, G. Padberg, B. van Engelen, and A. Geurts, “[both aerobic exercise and cognitive-behavioral therapy reduce fatigue in fshd: an rct].,” *Nederlands tijdschrift voor geneeskunde*, vol. 159, pp. A8806–A8806, 2014.
- [7] C. J. Linstrom, “Objective facial motion analysis in patients with facial nerve dysfunction,” *The Laryngoscope*, vol. 112, no. 7, pp. 1129–1147, 2002.
- [8] S. McGrenary, B. F. O’Reilly, and J. J. Soraghan, “Objective grading of facial paralysis using artificial intelligence analysis of video data,” in *Computer-Based Medical Systems, 2005. Proceedings. 18th IEEE Symposium on*, pp. 587–592, IEEE, 2005.
- [9] M. J. Brenner and J. G. Neely, “Approaches to grading facial nerve function,” in *Seminars in plastic surgery*, vol. 18, p. 13, Thieme Medical Publishers, 2004.
- [10] M. May, “Facial paralysis, peripheral type: a proposed method of reporting,” *The Laryngoscope*, vol. 80, no. 3, pp. 331–390, 1970.
- [11] J. W. House, “Facial nerve grading systems,” *The Laryngoscope*, vol. 93, no. 8, pp. 1056–1069, 1983.
- [12] J. W. House and D. E. Brackmann, “Facial nerve grading system,” *Otolaryngology Head and Neck Surgery*, vol. 93, no. 2, pp. 146–147, 1985.

- [13] D. E. Brackmann and D. M. Barrs, "Assessing recovery of facial function following acoustic neuroma surgery," *Otolaryngology Head and Neck Surgery*, vol. 92, no. 1, pp. 88–93, 1984.
- [14] T. S. Kang, J. T. Vrabec, N. Giddings, and D. J. Terris, "Facial nerve grading systems (1985–2002): beyond the house-brackmann scale," *Otology & neurotology*, vol. 23, no. 5, pp. 767–771, 2002.
- [15] S. Burres and U. Fisch, "The comparison of facial grading systems," *Archives of Otolaryngology–Head & Neck Surgery*, vol. 112, no. 7, pp. 755–758, 1986.
- [16] G. E. Murty, J. P. Diver, P. J. Kelly, G. O'Donoghue, and P. J. Bradley, "The nottingham system: objective assessment of facial nerve function in the clinic," *Otolaryngology–head and neck surgery*, vol. 110, no. 2, pp. 156–161, 1994.
- [17] B. G. Ross, G. Fradet, and J. M. Nedzelski, "Development of a sensitive clinical facial grading system," *Otolaryngology–head and neck surgery*, no. 3, pp. 380–386, 1996.
- [18] J. T. Vrabec, D. D. Backous, H. R. Djalilian, P. W. Gidley, J. P. Leonetti, S. J. Marzo, D. Morrison, M. Ng, M. J. Ramsey, B. M. Schaitkin, *et al.*, "Facial nerve grading system 2.0," *Otolaryngology-Head and Neck Surgery*, vol. 140, no. 4, pp. 445–450, 2009.
- [19] O. Azoulay, Y. Ater, L. Gersi, Y. Glassner, O. Bryt, and D. Halperin, "Mobile application for diagnosis of facial palsy,"
- [20] S. He, J. J. Soraghan, and B. F. O'Reilly, "Objective grading of facial paralysis using local binary patterns in video processing," in *Engineering in Medicine and Biology Society, 2008. EMBS 2008. 30th Annual International Conference of the IEEE*, pp. 4805–4808, IEEE, 2008.
- [21] P. C. Johnson, H. Brown, W. M. Kuzon Jr, R. Balliet, J. L. Garrison, and J. Campbell, "Simultaneous quantitation of facial movements: the maximal static response assay of facial nerve function.," *Annals of plastic surgery*, vol. 32, no. 2, pp. 171–179, 1994.
- [22] M. Isono, K. Murata, H. Tanaka, M. Kawamoto, and H. Azuma, "An objective evaluation method for facial mimic motion," *Otolaryngology–Head and Neck Surgery*, vol. 114, no. 1, pp. 27–31, 1996.
- [23] P. Dulguerov, F. Marchal, D. Wang, and C. Gysin, "Review of objective topographic facial nerve evaluation methods.," *Otology & Neurotology*, vol. 20, no. 5, pp. 672–678, 1999.
- [24] M. Frey, P. Giovanoli, H. Gerber, M. Slameczka, and E. Stüssi, "Three-dimensional video analysis of facial movements: a new method to assess the quantity and quality of the smile.," *Plastic and reconstructive surgery*, vol. 104, no. 7, pp. 2032–2039, 1999.
- [25] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.
- [26] A. E. Hoerl and R. W. Kennard, "Ridge regression: Biased estimation for nonorthogonal problems," *Technometrics*, vol. 12, no. 1, pp. 55–67, 1970.
- [27] S. He, J. J. Soraghan, and B. F. O'Reilly, "Biomedical image sequence analysis with application to automatic quantitative assessment of facial paralysis," *Journal on Image and Video Processing*, vol. 2007, no. 3, p. 2, 2007.

- [28] “Noldus facereader.” <http://www.noldus.com/human-behavior-research/products/facereader>. Accessed: 2015-02-12.
- [29] D. A. Wood, G. B. Hughes, M. Secic, and T. L. Good, “Objective measurement of normal facial movement with video microscaling,” *Otology & Neurotology*, vol. 15, no. 1, pp. 61–65, 1994.
- [30] G. S. Wachtman, J. F. Cohn, J. M. VanSwearingen, and E. K. Manders, “Automated tracking of facial features in patients with facial neuromuscular dysfunction,” *Plastic and reconstructive surgery*, vol. 107, no. 5, pp. 1124–1133, 2001.
- [31] J. J.-J. Lien, T. Kanade, J. F. Cohn, and C.-C. Li, “Detection, tracking, and classification of action units in facial expression,” *Robotics and Autonomous Systems*, vol. 31, no. 3, pp. 131–146, 2000.
- [32] B. D. Lucas, T. Kanade, *et al.*, “An iterative image registration technique with an application to stereo vision,” in *IJCAI*, vol. 81, pp. 674–679, 1981.
- [33] M. M. Gross, C.-A. Trotman, and K. S. Moffatt, “A comparison of three-dimensional and two-dimensional analyses of facial motion,” *The Angle Orthodontist*, vol. 66, no. 3, pp. 189–194, 1996.
- [34] K. Mishima, T. Yamada, A. Ohura, and T. Sugahara, “Production of a range image for facial motion analysis: a method for analyzing lip motion,” *Computerized Medical Imaging and Graphics*, vol. 30, no. 1, pp. 53–59, 2006.
- [35] H. Popat, S. Richmond, L. Benedikt, D. Marshall, and P. L. Rosin, “Quantitative analysis of facial movement—a review of three-dimensional imaging techniques,” *Computerized Medical Imaging and Graphics*, vol. 33, no. 5, pp. 377–383, 2009.
- [36] J. G. Neely, J. Y. Cheung, M. Wood, J. Byers, and A. Rogerson, “Computerized quantitative dynamic analysis of facial motion in the paralyzed and synkinetic face,” *Otology & Neurotology*, vol. 13, no. 2, pp. 97–107, 1992.
- [37] J. G. Neely, J. F. Jekel, and J. Y. Cheung, “Variations in maximum amplitude of facial expressions between and within normal subjects,” *Otolaryngology–Head and Neck Surgery*, vol. 110, no. 1, pp. 60–63, 1994.
- [38] C. J. Moran and J. G. Neely, “Patterns of facial nerve synkinesis,” *The Laryngoscope*, vol. 106, no. 12, pp. 1491–1496, 1996.
- [39] J. G. Neely, A. H. Joaquin, L. A. Kohn, and J. Y. Cheung, “Quantitative assessment of the variation within grades of facial paralysis,” *The Laryngoscope*, vol. 106, no. 4, pp. 438–442, 1996.
- [40] T. D. Helling and J. Gail Neely, “Validation of objective measures for facial paralysis,” *The Laryngoscope*, vol. 107, no. 10, pp. 1345–1349, 1997.
- [41] H. Scriba, S. J. Stoeckli, D. Veraguth, A. Pollak, and U. Fisch, “Objective evaluation of normal facial function,” *Annals of otology, rhinology & laryngology*, vol. 108, no. 7, pp. 641–644, 1999.

- [42] V. Meier-Gallati, H. Scriba, and U. Fisch, "Objective scaling of facial nerve function based on area analysis (oscar)," *Otolaryngology-Head and Neck Surgery*, vol. 118, no. 4, pp. 545–550, 1998.
- [43] L. Wang and D.-C. He, "Texture classification using texture spectrum," *Pattern Recognition*, vol. 23, no. 8, pp. 905–910, 1990.
- [44] T. Ojala, M. Pietikäinen, and D. Harwood, "Performance evaluation of texture measures with classification based on kullback discrimination of distributions," in *Pattern Recognition, 1994. Vol. 1-Conference A: Computer Vision & Image Processing., Proceedings of the 12th IAPR International Conference on*, no. 1, pp. 582–585, 1994.
- [45] T. Ojala, M. Pietikäinen, and D. Harwood, "A comparative study of texture measures with classification based on featured distributions," *Pattern recognition*, vol. 29, no. 1, pp. 51–59, 1996.
- [46] M. S. Karlan, "Contour analysis in plastic and reconstructive surgery," *Archives of Otolaryngology*, vol. 105, no. 11, pp. 670–679, 1979.
- [47] P. Ekman and W. V. Friesen, "Measuring facial movement," *Environmental psychology and nonverbal behavior*, vol. 1, no. 1, pp. 56–75, 1976.
- [48] K. Yuen, I. Inokuchi, M. Maeta, S.-I. Kawakami, and Y. Masuda, "Evaluation of facial palsy by moiré topography index," *Otolaryngology-Head and Neck Surgery*, vol. 117, no. 5, pp. 567–572, 1997.
- [49] J. F. Cohn, A. J. Zlochower, J. Lien, and T. Kanade, "Automated face analysis by feature point tracking has high concurrent validity with manual faces coding," *Psychophysiology*, vol. 36, no. 01, pp. 35–43, 1999.
- [50] Z. Zhang, "Microsoft kinect sensor and its effect," *MultiMedia, IEEE*, vol. 19, no. 2, pp. 4–10, 2012.
- [51] L. Cohen, "Time-frequency distributions-a review," *Proceedings of the IEEE*, vol. 77, no. 7, pp. 941–981, 1989.
- [52] A. Asthana, S. Zafeiriou, S. Cheng, and M. Pantic, "Robust discriminative response map fitting with constrained local models," in *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pp. 3444–3451, IEEE, 2013.
- [53] H. Gonzalez-Jorge, B. Riveiro, E. Vazquez-Fernandez, J. Martínez-Sánchez, and P. Arias, "Metrological evaluation of microsoft kinect and asus xtion sensors," *Measurement*, vol. 46, no. 6, pp. 1800–1806, 2013.
- [54] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.
- [55] P. Dulguerov, D. Wang, T. V. Perneger, F. Marchal, and W. Lehmann, "Videomimicography: the standards of normal revised," *Archives of Otolaryngology-Head & Neck Surgery*, vol. 129, no. 9, pp. 960–965, 2003.
- [56] H. Edelsbrunner, T. S. Tan, and R. Waupotitsch, "An  $O(n^2 \log n)$  time algorithm for the minmax angle triangulation," *SIAM Journal on Scientific and Statistical Computing*, vol. 13, no. 4, pp. 994–1008, 1992.

- [57] L. Sjögren, A. Lohmander, and S. Kiliaridis, “Exploring quantitative methods for evaluation of lip function,” *Journal of Oral Rehabilitation*, vol. 38, no. 6, pp. 410–422, 2011.
- [58] I. Guyon and A. Elisseeff, “An introduction to variable and feature selection,” *The Journal of Machine Learning Research*, vol. 3, pp. 1157–1182, 2003.
- [59] N. Meinshausen and P. Bühlmann, “Stability selection,” *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 72, no. 4, pp. 417–473, 2010.
- [60] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, “Scikit-learn: Machine learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.
- [61] N. V. Chawla, “Data mining for imbalanced datasets: An overview,” in *Data Mining and Knowledge Discovery Handbook*, pp. 875–886, Springer, 2010.
- [62] A. Trujillo-Ortiz, R. Hernandez-Walls, A. Castro-Perez, L. R.-C. N. Ramos-Delgado, and R. Garcia-Sanchez, “Barnardtest: Barnard’s exact probability test,” *A MATLAB file.[WWW document]*. URL <http://www.mathworks.com>, 2004.
- [63] A. F. Hayes and K. Krippendorff, “Answering the call for a standard reliability measure for coding data,” *Communication methods and measures*, vol. 1, no. 1, pp. 77–89, 2007.
- [64] K. De Swert, “Calculating inter-coder reliability in media content analysis using krippendorff’s alpha,” *Center for Politics and Communication*, 2012.
- [65] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “Smote: synthetic minority over-sampling technique,” *Journal of artificial intelligence research*, pp. 321–357, 2002.
- [66] C. Lanz, J. Denzler, and H.-M. Gross, “Robust landmark localization for facial therapy applications,”
- [67] F. Bellard, M. Niedermayer, *et al.*, “FFmpeg,” <http://ffmpeg.org>, 2014.
- [68] Microsoft, “Kinect 2 SDK,” <https://www.microsoft.com/en-us/kinectforwindows>, 2014.



## Appendix A

### Consent form

**Verklaring ten behoeve van digitale foto en/of videoregistratie**

Ondergetekende verklaart hierbij bereid te zijn mee te werken aan een digitale foto en/of videoregistratie, welke zal worden gemaakt ten behoeve van onderwijs en/of onderzoek.

Hij/zij verklaart bekend te zijn met de opzet en het karakter van de registraties en er geen bezwaar tegen te hebben dat deze registraties in hun geheel, gedeeltelijk of bewerkt, onder verantwoordelijkheid van een medicus of persoon die zich gebonden voelt door een medisch beroepsgeheim, worden vertoond in het kader van de opleiding van medici en/of paramedici of van medisch wetenschappelijk onderzoek.

Hij/zij weet dat de registraties niet via open-net zullen worden uitgezonden.

Hij/zij behoudt zich het recht voor deze toestemming te herroepen binnen een maand na de registratie binnen welke tijd hij/zij desgevraagd in de gelegenheid zal worden gesteld het resultaat van de audiovisuele registraties te bezien.

Datum: ..... Naam: .....

Plaats: ..... Geboortedatum: .....

Handtekening: .....

---

Patientensticker + reg.nr.

Nummer opname:

Diagnose:

Begin opname: .....

Einde opname: .....

## Appendix B

# Expert score sheets

The following pages are related to the expert score sheets and are in order of appearance:

1. Truncated empty expert scoresheet template, with the participant tags replaced by anonymous movie numbers (1 page, Dutch)
2. Facial weakness scores and FSHD predictions from the experts obtained from the filled-in scoresheet templates (2 pages)



tag	Carlen Beurskens												George Padberg												Simone Knütt																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																
	0			1			2			3			4			5			6			0			1			2			3			4			5			6																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																	
	l	r	F	l	r	F	l	r	F	l	r	F	l	r	F	l	r	F	l	r	F	l	r	F	l	r	F	l	r	F	l	r	F	l	r	F																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																					
e1	3	3	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3	2	3

[illegible]

# Appendix C

## Analyses listings

### C.1 FSHD system evaluation

Effect		Value	F	Hypothesis df	Error df	p	$\eta^2$
feature_group	Pillai's Trace	.995	4477.287 <sup>b</sup>	5.000	112.000	.000	.995
	Wilks' Lambda	.005	4477.287 <sup>b</sup>	5.000	112.000	.000	.995
	Hotelling's Trace	199.879	4477.287 <sup>b</sup>	5.000	112.000	.000	.995
	Roy's Largest Root	199.879	4477.287 <sup>b</sup>	5.000	112.000	.000	.995
feature_group * models	Pillai's Trace	1.858	37.118	15.000	342.000	.000	.619
	Wilks' Lambda	.010	90.309	15.000	309.584	.000	.787
	Hotelling's Trace	18.456	136.161	15.000	332.000	.000	.860
	Roy's Largest Root	11.873	270.706 <sup>c</sup>	5.000	114.000	.000	.922
feature_dimensions	Pillai's Trace	.979	2629.811 <sup>b</sup>	2.000	115.000	.000	.979
	Wilks' Lambda	.021	2629.811 <sup>b</sup>	2.000	115.000	.000	.979
	Hotelling's Trace	45.736	2629.811 <sup>b</sup>	2.000	115.000	.000	.979
	Roy's Largest Root	45.736	2629.811 <sup>b</sup>	2.000	115.000	.000	.979
feature_dimensions * models	Pillai's Trace	1.342	78.785	6.000	232.000	.000	.671
	Wilks' Lambda	.078	98.945b	6.000	230.000	.000	.721
	Hotelling's Trace	6.444	122.440	6.000	228.000	.000	.763
	Roy's Largest Root	5.458	211.061 <sup>c</sup>	3.000	116.000	.000	.845
selection_settings	Pillai's Trace	.993	3413.230 <sup>b</sup>	5.000	112.000	.000	.993
	Wilks' Lambda	.007	3413.230 <sup>b</sup>	5.000	112.000	.000	.993
	Hotelling's Trace	152.376	3413.230 <sup>b</sup>	5.000	112.000	.000	.993
	Roy's Largest Root	152.376	3413.230 <sup>b</sup>	5.000	112.000	.000	.993
selection_settings * models	Pillai's Trace	1.812	34.790	15.000	342.000	.000	.604
	Wilks' Lambda	.009	92.089	15.000	309.584	.000	.790
	Hotelling's Trace	26.198	193.279	15.000	332.000	.000	.897
	Roy's Largest Root	23.020	524.855 <sup>c</sup>	5.000	114.000	.000	.958
feature_group * feature_dimensions	Pillai's Trace	.983	780.495b	8.000	109.000	.000	.983
	Wilks' Lambda	.017	780.495b	8.000	109.000	.000	.983
	Hotelling's Trace	57.284	780.495b	8.000	109.000	.000	.983
	Roy's Largest Root	57.284	780.495b	8.000	109.000	.000	.983

Effect		Value	F	Hypothesis df	Error df	p	$\eta^2$
feature_group * feature_dimensions * models	Pillai's Trace	1.981	26.958	24.000	333.000	.000	.660
	Wilks' Lambda	.012	48.286	24.000	316.734	.000	.774
	Hotelling's Trace	15.349	68.856	24.000	323.000	.000	.837
	Roy's Largest Root	10.069	139.706 <sup>c</sup>	8.000	111.000	.000	.910
feature_group * selection_settings	Pillai's Trace	.968	110.046b	25.000	92.000	.000	.968
	Wilks' Lambda	.032	110.046b	25.000	92.000	.000	.968
	Hotelling's Trace	29.904	110.046b	25.000	92.000	.000	.968
	Roy's Largest Root	29.904	110.046b	25.000	92.000	.000	.968
feature_group * selection_settings * models	Pillai's Trace	2.024	7.802	75.000	282.000	.000	.675
	Wilks' Lambda	.010	13.744	75.000	275.890	.000	.788
	Hotelling's Trace	22.496	27.195	75.000	272.000	.000	.882
	Roy's Largest Root	19.845	74.616 <sup>c</sup>	25.000	94.000	.000	.952
feature_dimensions * selection_settings	Pillai's Trace	.945	182.603b	10.000	107.000	.000	.945
	Wilks' Lambda	.055	182.603b	10.000	107.000	.000	.945
	Hotelling's Trace	17.066	182.603b	10.000	107.000	.000	.945
	Roy's Largest Root	17.066	182.603b	10.000	107.000	.000	.945
feature_dimensions * selection_settings * models	Pillai's Trace	1.740	15.047	30.000	327.000	.000	.580
	Wilks' Lambda	.022	27.935	30.000	314.742	.000	.719
	Hotelling's Trace	12.110	42.656	30.000	317.000	.000	.801
	Roy's Largest Root	8.744	95.309 <sup>c</sup>	10.000	109.000	.000	.897
feature_group * feature_dimensions * selection_settings	Pillai's Trace	.982	107.020b	40.000	77.000	.000	.982
	Wilks' Lambda	.018	107.020b	40.000	77.000	.000	.982
	Hotelling's Trace	55.595	107.020b	40.000	77.000	.000	.982
	Roy's Largest Root	55.595	107.020b	40.000	77.000	.000	.982
feature_group * feature_dimensions * selection_settings * models	Pillai's Trace	2.201	5.444	120.000	237.000	.000	.734
	Wilks' Lambda	.002	13.742	120.000	231.597	.000	.876
	Hotelling's Trace	36.920	23.280	120.000	227.000	.000	.925
	Roy's Largest Root	19.512	38.536 <sup>c</sup>	40.000	79.000	.000	.951

**Table C.1:** Multivariate test statistics for the effects of classifier model, feature selection settings, feature dimensions and feature group on FSHD model performance.

<sup>b</sup> Exact statistic

<sup>c</sup> The statistic is an upper bound on F that yields a lower bound on the significance level.

Source	Type III Sum of Squares	df	Mean Square	F	p	$\eta^2$
Intercept	8660.130	1	8660.130	3662116.834	.000	1.000
classifier_model	36.171	3	12.057	5098.540	.000	.992
Error	.274	116	.002			

**Table C.2:** Tests of between-subjects effects on FSHD classification accuracy.



## C.2 FSHD features

feature label	#	feature label	#
T4_I1_n0_R_2_p48p59_dist	1832	T6_I1_n0_D_2_p48p54_Sjogreen_MWA	804
T0_I1_n1_A_2_p66p67_motion2d	1732	T0_I1_n1_A_2_p48p54_motion2d	804
T6_I1_n0_L_2_p42_motion2d	1684	T1_I1_n0_LM2_p52p55_dist	792
T1_I1_n0_R_3_p17p18_dist	1672	T4_I1_n0_L_2_p33p52p51_triangle	780
T5_I1_n0_A_2_p31p48p49p35p54p53_triangle	1664	T4_I1_n0_R_2_p48p65p59_triangle	776
T6_I1_n0_R_3_p50p60p61_triangle	1660	T1_I1_n0_R_3_p19p37p38_triangle	764
T6_I1_n0_R_2_p59p66_dist	1580	T1_I1_n0_A_3_p39p40p42p47_dist	752
T5_I1_n0_A_2_p48p59p54p55_dist	1576	T5_I1_n4_A___subtractabs	744
T1_I1_n0_A_3_p59p65p55p63_dist	1464	T5_I1_n0_A_2_p60p61p65p62p61p63_triangle	740
T6_I1_n0_L_3_p54p63_dist	1444	T6_I1_n0_L_2_p43_motion2d	728
T1_I1_n0_L_3_p26p25_dist	1440	T5_I1_n0_L_2_p35p53p52_triangle	728
T1_I1_n0_A_3_p60p57p62p57_dist	1424	T1_I1_n0_R_2_p19_motion2d	720
T6_I1_n0_L_3_p54p53_dist	1408	T6_I1_n0_LM2_p35p52_dist	720
T1_I1_n0_L_2_p25p24p44_triangle	1324	T5_I1_n0_A_3_p02p31p14p35_dist	720
T1_I1_n0_R_3_p20p21p38_triangle	1276	T6_I1_n0_LM2_p34p52_dist	720
T1_I1_n0_LM2_p53p67_dist	1216	T0_I1_n1_R_2_p36_motion2d	720
T6_I1_n0_R_3_p05p06p58_triangle	1204	T5_I1_n0_L_3_p25p24p44_triangle	720
T6_I1_n0_L_3_p11p10p56_triangle	1196	T1_I1_n0_A_2_p03p48p59p13p54p55_triangle	716
T6_I1_n0_R_2_p48p59_dist	1180	T6_I1_n0_A_3_p32p33p34p33_dist	712
T1_I1_n0_A_3_p49p50p53p52_dist	1140	T3_I1_n1_A_2_p60p62_motion2d	712
T5_I1_n1_A_2_p32p34_motion2d	1068	T3_I1_n0_LN2_p52p62_dist	712
T4_I1_n0_A_3_p01p32p15p34_dist	1068	T6_I1_n0_L_3_p62p64_dist	708
T6_I1_n0_L_3_p53p67_dist	1056	T1_I1_n0_A_2_p31p48p49p35p54p53_triangle	700
T4_I1_n0_LN2_p52p63_dist	1052	T2_I1_n0_LN2_p52p63_dist	696
T3_I1_n1_A_2_p39p42_motion2d	1036	T6_I1_n1_D_2_p48p54_Sjogreen_MWA	696
T4_I1_n0_RM2_p00p41_dist	1032	T4_I1_n0_RM2_p59p64_dist	692
T6_I1_n0_L_3_p54p55_dist	1028	T6_I1_n0_L_2_p54p55_dist	688
T2_I1_n0_A_3_p08p57p58p08p57p56_triangle	1028	T5_I1_n0_RM2_p31p48p49_triangle	680
T2_I1_n0_R_3_p32p33_dist	1012	T1_I1_n0_L_2_p25_motion2d	668
T4_I1_n0_R_3_p60p64_dist	1004	T3_I1_n0_R_2_p50_motion2d	668
T6_I1_n0_LM2_p45p46_dist	988	T5_I1_n0_A_3_p20p21p38p23p22p43_triangle	664
T6_I1_n0_L_3_p64p63_dist	976	T2_I1_n0_L_3_p26p25p44_triangle	656
T6_I1_n0_L_2_p08p57p56_triangle	964	T6_I1_n0_LM2_p35p33p52_triangle	656
T6_I1_n0_R_3_p49p66_dist	908	T1_I1_n0_A_3_p08p57p58p08p57p56_triangle	652
T5_I1_n0_RM2_p02p48_dist	872	T6_I1_n1___M2_p48p54_Sjogreen_MWA	644
T6_I1_n0_LM2_p52p67_dist	868	T4_I1_n3_R___subtractabs	644
T0_I1_n0_R___subtractabs	856	T5_I1_n0_A_3_p31p36p48p35p45p54_triangle	644
T5_I1_n0_A_3_p18p19p37p25p24p44_triangle	856	T0_I1_n0_A_3_p58p65p59p56p63p55_triangle	624
T6_I1_n0_L_3_p54p62_dist	852	T5_I1_n0_R_2_p48p60p65_triangle	620
T6_I1_n0_L_2_p55p67_dist	844	T6_I1_n0_LM2_p52p62p61_triangle	616
T4_I1_n0_R_3_p64p65_dist	844	T4_I1_n3_A___subtractabs	612

feature label	#	feature label	#
T5_I1_n0_R_3_p21p38p39_triangle	608	T6_I1_n0_LN2_p52p63_dist	488
T5_I1_n0_RM2_p31p66_dist	608	T6_I1_n0_LM2_p56p57_dist	484
T4_I1_n0_RM2_p02p31_dist	592	T4_I1_n0_R_3_p49p57_dist	484
T4_I1_n0_A_3_p00p17p36p16p26p45_triangle	592	T6_I1_n0_RM2_p31p49_dist	484
T5_I1_n0_R_2_p31p48p49_triangle	584	T6_I1_n0_LM2_p35p62_dist	480
T0_I1_n0_A_3_p00p17p36p16p26p45_triangle	584	T5_I1_n0_L_2_p23_motion2d	480
T6_I1_n0_A_3_p40p41p47p46_dist	572	T6_I1_n0_RM2_p32p50_dist	480
T6_I1_n0_RM2_p36p41_dist	568	T1_I1_n0_R_3_p31p49p50_triangle	480
T1_I1_n0_L_2_p26_motion2d	568	T0_I1_n0_R_2_p03p48p59_triangle	472
T6_I1_n0_LM2_p35p53_dist	564	T6_I1_n0_RM2_p31p50_dist	472
T6_I1_n0_L_3_p35p54p53_triangle	560	T1_I1_n0_A_3_p21p38p39p22p43p42_triangle	468
T1_I1_n0_L_2_p13p54p55_triangle	552	T2_I1_n0_A_3_p60p58p62p56_dist	460
T5_I1_n0_R_2_p05p58p59_triangle	548	T6_I1_n0_LM2_p35p63_dist	460
T6_I1_n0_L_2_p44_motion2d	548	T6_I1_n0_L_2_p47_motion2d	460
T1_I1_n0_R_2_p18_motion2d	544	T1_I1_n0_L_3_p26p25p44_triangle	456
T6_I1_n1_A__subtractabs	540	T1_I1_n0_R_2_p20_motion2d	456
T6_I1_n0_L_2_p46_motion2d	536	T3_I1_n0_A_3_p50p51p52p51_dist	452
T2_I1_n0_LN2_p52p62_dist	536	T0_I1_n0_A_3_p38p40p43p47_dist	452
T6_I1_n0_LM2_p54p52_dist	524	T3_I1_n0_L_3_p33p52p51_triangle	448
T6_I1_n0_R_2_p03p48p59_triangle	512	T6_I1_n0_LM2_p53p52_dist	440
T5_I1_n0_LM2_p45p46_dist	512	T4_I1_n0_RN2_p02p31_dist	436
T3_I1_n0_A_3_p33p50p51p33p52p51_triangle	512	T6_I1_n0_L_3_p63p67_dist	428
T0_I1_n0_A_3_p19p20p38p24p23p43_triangle	508	T5_I1_n0_RM2_p02p66_dist	424
T2_I1_n0_A_3_p49p65p53p63_dist	504	T6_I1_n0_L_3_p34p33_dist	424
T6_I1_n0_L_3_p54p56_dist	504	T3_I1_n0_A_3_p37p41p44p46_dist	424
T6_I1_n0_R_3_p65p66_dist	492		

**Table C.3:** Listing of the selected 133 features by their labels in order of selection frequency

### C.3 Weakness grading system evaluation

Effect		Value	F	Hypothesis df	Error df	p	$\eta^2$
feature_group	Pillai's Trace	.992	88118,243b	5.000	3412.000	.000	.992
	Wilks' Lambda	.008	88118,243b	5.000	3412.000	.000	.992
	Hotelling's Trace	129.130	88118,243b	5.000	3412.000	.000	.992
	Roy's Largest Root	129.130	88118,243b	5.000	3412.000	.000	.992
feature_group * taskid	Pillai's Trace	2.983	842.136	30.000	17080.000	.000	.597
	Wilks' Lambda	.001	1992.837	30.000	13650.000	.000	.740
	Hotelling's Trace	22.647	2574.488	30.000	17052.000	.000	.819
	Roy's Largest Root	9.661	5500,522c	6.000	3416.000	.000	.906
feature_group * side	Pillai's Trace	.465	593,977b	5.000	3412.000	.000	.465
	Wilks' Lambda	.535	593,977b	5.000	3412.000	.000	.465
	Hotelling's Trace	.870	593,977b	5.000	3412.000	.000	.465
	Roy's Largest Root	.870	593,977b	5.000	3412.000	.000	.465
feature_group * classifier_model	Pillai's Trace	1.329	543.304	15.000	10242.000	.000	.443
	Wilks' Lambda	.052	1207.074	15.000	9419.431	.000	.627
	Hotelling's Trace	11.013	2504.218	15.000	10232.000	.000	.786
	Roy's Largest Root	10.314	7042,342c	5.000	3414.000	.000	.912
feature_group * taskid * side	Pillai's Trace	1.040	149.545	30.000	17080.000	.000	.208
	Wilks' Lambda	.274	173.972	30.000	13650.000	.000	.228
	Hotelling's Trace	1.672	190.123	30.000	17052.000	.000	.251
	Roy's Largest Root	.982	559,098c	6.000	3416.000	.000	.495
feature_group * taskid * classifier_model	Pillai's Trace	2.709	224.455	90.000	17080.000	.000	.542
	Wilks' Lambda	.011	279.368	90.000	16557.068	.000	.592
	Hotelling's Trace	8.741	331.207	90.000	17052.000	.000	.636
	Roy's Largest Root	4.011	761,230c	18.000	3416.000	.000	.800
feature_group * side * classifier_model	Pillai's Trace	.134	32.027	15.000	10242.000	.000	.045
	Wilks' Lambda	.870	32.452	15.000	9419.431	.000	.045
	Hotelling's Trace	.144	32.751	15.000	10232.000	.000	.046
	Roy's Largest Root	.094	64,133c	5.000	3414.000	.000	.086
feature_group * taskid * side * classifier_model	Pillai's Trace	.753	33.653	90.000	17080.000	.000	.151
	Wilks' Lambda	.434	34.539	90.000	16557.068	.000	.154
	Hotelling's Trace	.930	35.258	90.000	17052.000	.000	.157
	Roy's Largest Root	.348	66,107c	18.000	3416.000	.000	.258
feature_dimensions	Pillai's Trace	.978	76626,265b	2.000	3415.000	.000	.978
	Wilks' Lambda	.022	76626,265b	2.000	3415.000	.000	.978
	Hotelling's Trace	44.876	76626,265b	2.000	3415.000	.000	.978
	Roy's Largest Root	44.876	76626,265b	2.000	3415.000	.000	.978
feature_dimensions * taskid	Pillai's Trace	1.204	862.046	12.000	6832.000	.000	.602
	Wilks' Lambda	.090	1330,054b	12.000	6830.000	.000	.700
	Hotelling's Trace	6.858	1950.976	12.000	6828.000	.000	.774
	Roy's Largest Root	6.341	3609,998c	6.000	3416.000	.000	.864

Effect		Value	F	Hypothesis df	Error df	p	$\eta^2$
feature_dimensions * side	Pillai's Trace	.095	178,658b	2.000	3415.000	.000	.095
	Wilks' Lambda	.905	178,658b	2.000	3415.000	.000	.095
	Hotelling's Trace	.105	178,658b	2.000	3415.000	.000	.095
	Roy's Largest Root	.105	178,658b	2.000	3415.000	.000	.095
feature_dimensions * classifier_model	Pillai's Trace	.731	655.652	6.000	6832.000	.000	.365
	Wilks' Lambda	.277	1024,787b	6.000	6830.000	.000	.474
	Hotelling's Trace	2.583	1469.724	6.000	6828.000	.000	.564
	Roy's Largest Root	2.572	2928,792c	3.000	3416.000	.000	.720
feature_dimensions * taskid * side	Pillai's Trace	.373	130.438	12.000	6832.000	.000	.186
	Wilks' Lambda	.641	141,913b	12.000	6830.000	.000	.200
	Hotelling's Trace	.540	153.571	12.000	6828.000	.000	.213
	Roy's Largest Root	.497	283,238c	6.000	3416.000	.000	.332
feature_dimensions * taskid * classifier_model	Pillai's Trace	.868	145.521	36.000	6832.000	.000	.434
	Wilks' Lambda	.288	163,568b	36.000	6830.000	.000	.463
	Hotelling's Trace	1.925	182.580	36.000	6828.000	.000	.490
	Roy's Largest Root	1.583	300,342c	18.000	3416.000	.000	.613
feature_dimensions * side * classifier_model	Pillai's Trace	.022	12.438	6.000	6832.000	.000	.011
	Wilks' Lambda	.978	12,500b	6.000	6830.000	.000	.011
	Hotelling's Trace	.022	12.561	6.000	6828.000	.000	.011
	Roy's Largest Root	.022	24,800c	3.000	3416.000	.000	.021
feature_dimensions * taskid * side * classifier_model	Pillai's Trace	.550	71.991	36.000	6832.000	.000	.275
	Wilks' Lambda	.522	72,839b	36.000	6830.000	.000	.277
	Hotelling's Trace	.777	73.689	36.000	6828.000	.000	.280
	Roy's Largest Root	.501	95,151c	18.000	3416.000	.000	.334
selection_settings	Pillai's Trace	.906	10993,535b	3.000	3414.000	.000	.906
	Wilks' Lambda	.094	10993,535b	3.000	3414.000	.000	.906
	Hotelling's Trace	9.660	10993,535b	3.000	3414.000	.000	.906
	Roy's Largest Root	9.660	10993,535b	3.000	3414.000	.000	.906
selection_settings * taskid	Pillai's Trace	.436	96.699	18.000	10248.000	.000	.145
	Wilks' Lambda	.592	109.077	18.000	9656.735	.000	.160
	Hotelling's Trace	.641	121.486	18.000	10238.000	.000	.176
	Roy's Largest Root	.557	316,987c	6.000	3416.000	.000	.358
selection_settings * side	Pillai's Trace	.030	34,912b	3.000	3414.000	.000	.030
	Wilks' Lambda	.970	34,912b	3.000	3414.000	.000	.030
	Hotelling's Trace	.031	34,912b	3.000	3414.000	.000	.030
	Roy's Largest Root	.031	34,912b	3.000	3414.000	.000	.030
selection_settings * classifier_model	Pillai's Trace	.997	566.378	9.000	10248.000	.000	.332
	Wilks' Lambda	.238	741.669	9.000	8308.930	.000	.380
	Hotelling's Trace	2.221	842.286	9.000	10238.000	.000	.425
	Roy's Largest Root	1.627	1852,066c	3.000	3416.000	.000	.619
selection_settings * taskid * side	Pillai's Trace	.180	36.371	18.000	10248.000	.000	.060
	Wilks' Lambda	.826	37.423	18.000	9656.735	.000	.062
	Hotelling's Trace	.202	38.353	18.000	10238.000	.000	.063
	Roy's Largest Root	.153	87,276c	6.000	3416.000	.000	.133

Effect		Value	F	Hypothesis df	Error df	p	eta <sup>2</sup>
selection_settings * taskid *	Pillai's Trace	.617	49.145	54.000	10248.000	.000	.206
	Wilks' Lambda	.476	53.349	54.000	10173.190	.000	.219
classifier_model	Hotelling's Trace	.916	57.885	54.000	10238.000	.000	.234
	Roy's Largest Root	.672	127,473c	18.000	3416.000	.000	.402
selection_settings * side *	Pillai's Trace	.023	8.948	9.000	10248.000	.000	.008
	Wilks' Lambda	.977	8.968	9.000	8308.930	.000	.008
classifier_model	Hotelling's Trace	.024	8.974	9.000	10238.000	.000	.008
	Roy's Largest Root	.015	16,951c	3.000	3416.000	.000	.015
selection_settings * taskid * side *	Pillai's Trace	.272	18.944	54.000	10248.000	.000	.091
	Wilks' Lambda	.744	19.699	54.000	10173.190	.000	.094
classifier_model	Hotelling's Trace	.324	20.471	54.000	10238.000	.000	.097
	Roy's Largest Root	.245	46,421c	18.000	3416.000	.000	.197
feature_group * feature_dimensions	Pillai's Trace	.979	20098,223b	8.000	3409.000	.000	.979
	Wilks' Lambda	.021	20098,223b	8.000	3409.000	.000	.979
	Hotelling's Trace	47.165	20098,223b	8.000	3409.000	.000	.979
	Roy's Largest Root	47.165	20098,223b	8.000	3409.000	.000	.979
feature_group * feature_dimensions * taskid	Pillai's Trace	2.789	370.617	48.000	20484.000	.000	.465
	Wilks' Lambda	.003	792.139	48.000	16777.772	.000	.621
	Hotelling's Trace	20.069	1424.590	48.000	20444.000	.000	.770
	Roy's Largest Root	12.161	5189,894c	8.000	3414.000	.000	.924
feature_group * feature_dimensions * side	Pillai's Trace	.207	111,570b	8.000	3409.000	.000	.207
	Wilks' Lambda	.793	111,570b	8.000	3409.000	.000	.207
	Hotelling's Trace	.262	111,570b	8.000	3409.000	.000	.207
	Roy's Largest Root	.262	111,570b	8.000	3409.000	.000	.207
feature_group * feature_dimensions * classifier_model	Pillai's Trace	1.310	330.701	24.000	10233.000	.000	.437
	Wilks' Lambda	.095	517.119	24.000	9887.738	.000	.544
	Hotelling's Trace	5.549	787.838	24.000	10223.000	.000	.649
	Roy's Largest Root	4.769	2033,403c	8.000	3411.000	.000	.827
feature_group * feature_dimensions * taskid * side	Pillai's Trace	.982	83.506	48.000	20484.000	.000	.164
	Wilks' Lambda	.300	96.972	48.000	16777.772	.000	.182
	Hotelling's Trace	1.530	108.624	48.000	20444.000	.000	.203
	Roy's Largest Root	.917	391,299c	8.000	3414.000	.000	.478
feature_group * feature_dimensions * taskid *	Pillai's Trace	2.340	78.443	144.000	27328.000	.000	.292
	Wilks' Lambda	.042	93.501	144.000	25094.804	.000	.326
classifier_model	Hotelling's Trace	4.593	108.687	144.000	27258.000	.000	.365
	Roy's Largest Root	1.815	344,528c	18.000	3416.000	.000	.645
feature_group * feature_dimensions * side *	Pillai's Trace	.115	17.012	24.000	10233.000	.000	.038
	Wilks' Lambda	.888	17.179	24.000	9887.738	.000	.039
classifier_model	Hotelling's Trace	.122	17.326	24.000	10223.000	.000	.039
	Roy's Largest Root	.080	34,050c	8.000	3411.000	.000	.074
feature_group * feature_dimensions * taskid * side *	Pillai's Trace	1.154	31.995	144.000	27328.000	.000	.144
	Wilks' Lambda	.268	34.167	144.000	25094.804	.000	.152
classifier_model	Hotelling's Trace	1.525	36.079	144.000	27258.000	.000	.160
	Roy's Largest Root	.588	111,519c	18.000	3416.000	.000	.370

Effect		Value	F	Hypothesis df	Error df	p	$\eta^2$
feature_group * selection_settings	Pillai's Trace	.743	655,030b	15.000	3402.000	.000	.743
	Wilks' Lambda	.257	655,030b	15.000	3402.000	.000	.743
	Hotelling's Trace	2.888	655,030b	15.000	3402.000	.000	.743
	Roy's Largest Root	2.888	655,030b	15.000	3402.000	.000	.743
feature_group * selection_settings * taskid	Pillai's Trace	1.565	80.134	90.000	20442.000	.000	.261
	Wilks' Lambda	.156	83.380	90.000	19138.137	.000	.267
	Hotelling's Trace	2.246	84.856	90.000	20402.000	.000	.272
	Roy's Largest Root	.591	134,248c	15.000	3407.000	.000	.371
feature_group * selection_settings * side	Pillai's Trace	.128	33,199b	15.000	3402.000	.000	.128
	Wilks' Lambda	.872	33,199b	15.000	3402.000	.000	.128
	Hotelling's Trace	.146	33,199b	15.000	3402.000	.000	.128
	Roy's Largest Root	.146	33,199b	15.000	3402.000	.000	.128
feature_group * selection_settings * classifier_model	Pillai's Trace	.755	76.309	45.000	10212.000	.000	.252
	Wilks' Lambda	.388	84.231	45.000	10107.256	.000	.270
	Hotelling's Trace	1.220	92.232	45.000	10202.000	.000	.289
	Roy's Largest Root	.828	187,935c	15.000	3404.000	.000	.453
feature_group * selection_settings * taskid * side	Pillai's Trace	.470	19.303	90.000	20442.000	.000	.078
	Wilks' Lambda	.604	19.937	90.000	19138.137	.000	.081
	Hotelling's Trace	.542	20.493	90.000	20402.000	.000	.083
	Roy's Largest Root	.249	56,667c	15.000	3407.000	.000	.200
feature_group * selection_settings * taskid * classifier_model	Pillai's Trace	1.828	26.344	270.000	51240.000	.000	.122
	Wilks' Lambda	.109	30.852	270.000	39420.635	.000	.138
	Hotelling's Trace	2.813	35.418	270.000	51002.000	.000	.158
	Roy's Largest Root	1.218	231,157c	18.000	3416.000	.000	.549
feature_group * selection_settings * side * classifier_model	Pillai's Trace	.199	16.104	45.000	10212.000	.000	.066
	Wilks' Lambda	.809	16.561	45.000	10107.256	.000	.068
	Hotelling's Trace	.225	17.017	45.000	10202.000	.000	.070
	Roy's Largest Root	.167	37,965c	15.000	3404.000	.000	.143
feature_group * selection_settings * taskid * side * classifier_model	Pillai's Trace	1.016	13.787	270.000	51240.000	.000	.068
	Wilks' Lambda	.330	14.669	270.000	39420.635	.000	.071
	Hotelling's Trace	1.215	15.298	270.000	51002.000	.000	.075
	Roy's Largest Root	.298	56,637c	18.000	3416.000	.000	.230
feature_dimensions * selection_settings	Pillai's Trace	.385	355,542b	6.000	3411.000	.000	.385
	Wilks' Lambda	.615	355,542b	6.000	3411.000	.000	.385
	Hotelling's Trace	.625	355,542b	6.000	3411.000	.000	.385
	Roy's Largest Root	.625	355,542b	6.000	3411.000	.000	.385
feature_dimensions * selection_settings * taskid	Pillai's Trace	.727	78.465	36.000	20496.000	.000	.121
	Wilks' Lambda	.434	87.211	36.000	14981.511	.000	.130
	Hotelling's Trace	.972	92.010	36.000	20456.000	.000	.139
	Roy's Largest Root	.482	274,563c	6.000	3416.000	.000	.325
feature_dimensions * selection_settings * side	Pillai's Trace	.008	4,473b	6.000	3411.000	.000	.008
	Wilks' Lambda	.992	4,473b	6.000	3411.000	.000	.008
	Hotelling's Trace	.008	4,473b	6.000	3411.000	.000	.008
	Roy's Largest Root	.008	4,473b	6.000	3411.000	.000	.008

Effect		Value	F	Hypothesis df	Error df	p	$\eta^2$
feature_dimensions * selection_settings * classifier_model	Pillai's Trace	.367	79.332	18.000	10239.000	.000	.122
	Wilks' Lambda	.650	88.152	18.000	9648.250	.000	.134
	Hotelling's Trace	.512	96.985	18.000	10229.000	.000	.146
	Roy's Largest Root	.457	259,767c	6.000	3413.000	.000	.314
feature_dimensions * selection_settings * taskid * side	Pillai's Trace	.254	25.206	36.000	20496.000	.000	.042
	Wilks' Lambda	.766	26.046	36.000	14981.511	.000	.043
	Hotelling's Trace	.280	26.500	36.000	20456.000	.000	.045
	Roy's Largest Root	.150	85,551c	6.000	3416.000	.000	.131
feature_dimensions * selection_settings * taskid * classifier_model	Pillai's Trace	.765	27.726	108.000	20496.000	.000	.127
	Wilks' Lambda	.427	28.975	108.000	19555.828	.000	.132
	Hotelling's Trace	.953	30.073	108.000	20456.000	.000	.137
	Roy's Largest Root	.363	68,925c	18.000	3416.000	.000	.266
feature_dimensions * selection_settings * side * classifier_model	Pillai's Trace	.050	9.727	18.000	10239.000	.000	.017
	Wilks' Lambda	.950	9.756	18.000	9648.250	.000	.017
	Hotelling's Trace	.052	9.777	18.000	10229.000	.000	.017
	Roy's Largest Root	.032	18,118c	6.000	3413.000	.000	.031
feature_dimensions * selection_settings * taskid * side * classifier_model	Pillai's Trace	.330	11.047	108.000	20496.000	.000	.055
	Wilks' Lambda	.709	11.210	108.000	19555.828	.000	.056
	Hotelling's Trace	.360	11.349	108.000	20456.000	.000	.057
	Roy's Largest Root	.142	26,897c	18.000	3416.000	.000	.124
feature_group * feature_dimensions * selection_settings	Pillai's Trace	.684	306,090b	24.000	3393.000	.000	.684
	Wilks' Lambda	.316	306,090b	24.000	3393.000	.000	.684
	Hotelling's Trace	2.165	306,090b	24.000	3393.000	.000	.684
	Roy's Largest Root	2.165	306,090b	24.000	3393.000	.000	.684
feature_group * feature_dimensions * selection_settings * taskid	Pillai's Trace	1.636	53.078	144.000	20388.000	.000	.273
	Wilks' Lambda	.136	56.093	144.000	19837.322	.000	.283
	Hotelling's Trace	2.494	58.734	144.000	20348.000	.000	.294
	Roy's Largest Root	.876	124,060c	24.000	3398.000	.000	.467
feature_group * feature_dimensions * selection_settings * side	Pillai's Trace	.108	17,081b	24.000	3393.000	.000	.108
	Wilks' Lambda	.892	17,081b	24.000	3393.000	.000	.108
	Hotelling's Trace	.121	17,081b	24.000	3393.000	.000	.108
	Roy's Largest Root	.121	17,081b	24.000	3393.000	.000	.108
feature_group * feature_dimensions * selection_settings * classifier_model	Pillai's Trace	.889	59.544	72.000	10185.000	.000	.296
	Wilks' Lambda	.290	72.321	72.000	10140.787	.000	.338
	Hotelling's Trace	1.875	88.333	72.000	10175.000	.000	.385
	Roy's Largest Root	1.547	218,817c	24.000	3395.000	.000	.607
feature_group * feature_dimensions * selection_settings * taskid * side	Pillai's Trace	.928	25.896	144.000	20388.000	.000	.155
	Wilks' Lambda	.355	26.724	144.000	19837.322	.000	.159
	Hotelling's Trace	1.166	27.459	144.000	20348.000	.000	.163
	Roy's Largest Root	.398	56,350c	24.000	3398.000	.000	.285
feature_group * feature_dimensions * selection_settings * taskid * classifier_model	Pillai's Trace	2.439	22.271	432.000	61380.000	.000	.136
	Wilks' Lambda	.054	25.357	432.000	49061.543	.000	.149
	Hotelling's Trace	3.560	27.944	432.000	61040.000	.000	.165
	Roy's Largest Root	.859	122,068c	24.000	3410.000	.000	.462

Effect		Value	F	Hypothesis df	Error df	p	eta <sup>2</sup>
feature_group *	Pillai's Trace	.242	12.423	72.000	10185.000	.000	.081
feature_dimensions	Wilks' Lambda	.771	12.789	72.000	10140.787	.000	.083
* selection_settings	Hotelling's Trace	.279	13.163	72.000	10175.000	.000	.085
* side *	Roy's Largest Root	.201	28,498c	24.000	3395.000	.000	.168
classifier_model	Pillai's Trace	1.223	10.354	432.000	61380.000	.000	.068
feature_group *	Wilks' Lambda	.264	10.966	432.000	49061.543	.000	.071
feature_dimensions	Hotelling's Trace	1.457	11.437	432.000	61040.000	.000	.075
* selection_settings	Roy's Largest Root	.345	48,964c	24.000	3410.000	.000	.256
* taskid * side *							
classifier_model							

**Table C.4:** Multivariate test statistics for the effects of classifier model, feature selection settings, feature dimensions and feature group on weakness classification performance.

<sup>b</sup> Exact statistic

<sup>c</sup> The statistic is an upper bound on F that yields a lower bound on the significance level.

Source	Type III Sum of Squares	df	Mean Square	F	p	eta <sup>2</sup>
Intercept	75248.894	1	75248.894	9819929.252	.000	1.000
taskid	532.747	6	88.791	11587.177	.000	.953
side	24.007	1	24.007	3132.844	.000	.478
classifier_model	164.265	3	54.755	7145.488	.000	.863
taskid * side	15.082	6	2.514	328.030	.000	.366
taskid * classifier_model	291.570	18	16.198	2113.870	.000	.918
side * classifier_model	1.329	3	.443	57.797	.000	.048
taskid * side * classifier_model	28.802	18	1.600	208.814	.000	.524
Error	26.176	3416	.008			

**Table C.5:** Tests of between-subjects effects on facial weakness classification accuracy.



## C.4 Best weakness grading systems

model	taskid	side	accuracy	parameters
Ridge	0	R	0.874	normalize: False, alpha: 5.0, tol: 0.01, solver: cholesky
	1	R	0.907	normalize: False, alpha: 5.0, tol: 0.01, solver: cholesky
	2	R	0.919	normalize: False, alpha: 5.0, tol: 0.01, solver: sparse_cg
	3	R	0.805	normalize: False, alpha: 5.0, tol: 0.001, solver: svd
	4	R	0.955	normalize: True, alpha: 0.5, tol: 0.01, solver: sparse_cg
	5	R	0.828	normalize: True, alpha: 0.05, tol: 0.01, solver: lsqr
	6	R	0.872	normalize: True, alpha: 0.5, tol: 1e-05, solver: svd
	0	L	0.816	normalize: False, alpha: 5.0, tol: 0.0001, solver: cholesky
	1	L	0.850	normalize: True, alpha: 0.05, tol: 0.01, solver: svd
	2	L	0.919	normalize: False, alpha: 5.0, tol: 0.01, solver: sparse_cg
	3	L	0.839	normalize: True, alpha: 0.05, tol: 0.0001, solver: sparse_cg
	4	L	0.930	normalize: True, alpha: 0.05, tol: 1e-05, solver: lsqr
	5	L	0.896	normalize: False, alpha: 5.0, tol: 1e-05, solver: sparse_cg
	6	L	0.862	normalize: True, alpha: 0.5, tol: 1e-05, solver: sparse_cg
ENet	0	R	0.875	normalize: True, l1_ratio: 0.75, tol: 0.01, alpha: 0.0005, random_state: 35
	1	R	0.886	normalize: False, l1_ratio: 0.1, tol: 0.01, alpha: 0.05, random_state: 8
	2	R	0.907	normalize: False, l1_ratio: 0.25, tol: 0.01, alpha: 0.005, random_state: 61
	3	R	0.783	normalize: False, l1_ratio: 0.25, tol: 0.01, alpha: 0.05, random_state: 36
	4	R	0.954	normalize: True, l1_ratio: 0.1, tol: 1e-05, alpha: 0.0005, random_state: 45
	5	R	0.816	normalize: True, l1_ratio: 0.25, tol: 0.001, alpha: 0.0005, random_state: 68
	6	R	0.862	normalize: True, l1_ratio: 0.1, tol: 0.01, alpha: 0.005, random_state: 60
	0	L	0.805	normalize: False, l1_ratio: 0.25, tol: 0.001, alpha: 0.05, random_state: 27
	1	L	0.827	normalize: False, l1_ratio: 0.1, tol: 0.01, alpha: 0.05, random_state: 8
	2	L	0.918	normalize: False, l1_ratio: 0.1, tol: 1e-05, alpha: 0.05, random_state: 1
	3	L	0.849	normalize: True, l1_ratio: 0.25, tol: 0.01, alpha: 0.0005, random_state: 29
	4	L	0.930	normalize: False, l1_ratio: 0.1, tol: 0.01, alpha: 0.005, random_state: 60
KNNR	5	L	0.896	normalize: True, l1_ratio: 0.25, tol: 0.001, alpha: 0.0005, random_state: 43
	6	L	0.862	normalize: True, l1_ratio: 0.1, tol: 0.0001, alpha: 0.005, random_state: 23
	0	R	0.884	n_neighbors: 2, weights: distance
	1	R	0.781	n_neighbors: 3, weights: distance
	2	R	0.874	n_neighbors: 1, weights: distance
	3	R	0.803	n_neighbors: 10, weights: distance
	4	R	0.907	n_neighbors: 10, weights: uniform
	5	R	0.852	n_neighbors: 5, weights: distance
	6	R	0.839	n_neighbors: 3, weights: uniform
	0	L	0.884	n_neighbors: 3, weights: uniform
	1	L	0.860	n_neighbors: 10, weights: distance
	2	L	0.896	n_neighbors: 5, weights: distance
	3	L	0.805	n_neighbors: 10, weights: distance
	4	L	0.896	n_neighbors: 10, weights: distance
	5	L	0.839	n_neighbors: 10, weights: distance
	6	L	0.827	n_neighbors: 10, weights: distance

model	taskid	side	accuracy	parameters
RFR	0	R	0.875	n_estimators: 80, bootstrap: False, random_state: 12
	1	R	0.725	n_estimators: 120, bootstrap: True, random_state: 64
	2	R	0.827	n_estimators: 20, bootstrap: False, random_state: 36
	3	R	0.805	n_estimators: 20, bootstrap: True, random_state: 6
	4	R	0.852	n_estimators: 40, bootstrap: True, random_state: 54
	5	R	0.819	n_estimators: 80, bootstrap: False, random_state: 21
	6	R	0.792	n_estimators: 120, bootstrap: True, random_state: 27
	0	L	0.897	n_estimators: 80, bootstrap: False, random_state: 47
	1	L	0.783	n_estimators: 40, bootstrap: True, random_state: 46
	2	L	0.826	n_estimators: 20, bootstrap: True, random_state: 27
	3	L	0.769	n_estimators: 80, bootstrap: True, random_state: 38
	4	L	0.884	n_estimators: 20, bootstrap: True, random_state: 26
	5	L	0.794	n_estimators: 120, bootstrap: True, random_state: 19
	6	L	0.793	n_estimators: 80, bootstrap: False, random_state: 21

**Table C.6:** Best systems for grading facial weakness for a particular task and side of the face. The selection criterium was classification accuracy on the facial weakness ground truth after rounding the outcomes to the nearest integer within the 1-3 domain.

## C.5 Weakness grading expert reliability

Taskid	Side	Group	Alpha	LL95%CI	UL95%CI	Units	Observrs	Pairs
0	L	Ex1-Ex2-Ex3	-.1974	-.3328	-.0579	87	3	261
0	L	Ex1-Ex2	-.6795	-.8441	-.5056	87	2	87
0	L	Ex1-Ex3	.6447	.4129	.8366	87	2	87
0	L	Ex2-Ex3	-.6878	-.8391	-.5227	87	2	87
0	L	Ex1-GT	.9298	.8172	.0000	87	2	87
0	L	Ex2-GT	-.6414	-.8145	.4603	87	2	87
0	L	Ex3-GT	.7298	.5609	.8987	87	2	87
0	L	S1-S2-S3-S4	.6408	.5343	.7457	87	4	522
0	L	S1-GT	.7024	.5235	.8564	87	2	87
0	L	S2-GT	.6502	.4703	.8103	87	2	87
0	L	S3-GT	.7500	.5706	.9006	87	2	87
0	L	S4-GT	.7485	.5650	.9010	87	2	87
0	R	Ex1-Ex2-Ex3	-.1836	-.3220	-.0414	87	3	261
0	R	Ex1-Ex2	-.7210	-.8934	-.5465	87	2	87
0	R	Ex1-Ex3	.7739	.6152	.9181	87	2	87
0	R	Ex2-Ex3	-.6762	-.8417	-.4955	87	2	87
0	R	Ex1-GT	.8848	.7611	.9755	87	2	87
0	R	Ex2-GT	-.6567	-.8372	.4712	87	2	87
0	R	Ex3-GT	.9015	.8030	.9754	87	2	87
0	R	S1-S2-S3-S4	.7349	.6426	.8211	87	4	522
0	R	S1-GT	.7718	.6194	.9013	87	2	87
0	R	S2-GT	.7959	.6653	.9091	87	2	87
0	R	S3-GT	.8649	.7408	.9638	87	2	87
0	R	S4-GT	.8009	.6459	.9263	87	2	87
1	L	Ex1-Ex2-Ex3	.6330	.5494	.7089	87	3	261
1	L	Ex1-Ex2	.5918	.4604	.7111	87	2	87
1	L	Ex1-Ex3	.7804	.6674	.8779	87	2	87
1	L	Ex2-Ex3	.5287	.3825	.6622	87	2	87
1	L	Ex1-GT	.9449	.8903	.9888	87	2	87
1	L	Ex2-GT	.6632	.5345	.7711	87	2	87
1	L	Ex3-GT	.8432	.7491	.9216	87	2	87
1	L	S1-S2-S3-S4	.7365	.6633	.8047	87	4	522
1	L	S1-GT	.8305	.7341	.9195	87	2	87
1	L	S2-GT	.8264	.7302	.9101	87	2	87
1	L	S3-GT	.8505	.7657	.9310	87	2	87
1	L	S4-GT	.7604	.6526	.8615	87	2	87

Taskid	Side	Group	Alpha	LL95%CI	UL95%CI	Units	Observers	Pairs
1	R	Ex1-Ex2-Ex3	.6234	.5393	.7011	87	3	261
1	R	Ex1-Ex2	.5812	.4447	.7050	87	2	87
1	R	Ex1-Ex3	.8077	.6992	.9033	87	2	87
1	R	Ex2-Ex3	.4815	.3356	.6195	87	2	87
1	R	Ex1-GT	.9716	.9316	.9958	87	2	87
1	R	Ex2-GT	.6329	.5103	.7385	87	2	87
1	R	Ex3-GT	.8383	.7413	.9191	87	2	87
1	R	S1-S2-S3-S4	.6742	.5865	.7562	87	4	522
1	R	S1-GT	.8869	.8048	.9598	87	2	87
1	R	S2-GT	.8322	.7162	.9325	87	2	87
1	R	S3-GT	.7280	.5917	.8471	87	2	87
1	R	S4-GT	.6605	.5180	.7877	87	2	87
2	L	Ex1-Ex2-Ex3	.4986	.3873	.6029	87	3	261
2	L	Ex1-Ex2	.3836	.1866	.5726	87	2	87
2	L	Ex1-Ex3	.5632	.4077	.7062	87	2	87
2	L	Ex2-Ex3	.5461	.3594	.7137	87	2	87
2	L	Ex1-GT	.7030	.5679	.8223	87	2	87
2	L	Ex2-GT	.7142	.5560	.8510	87	2	87
2	L	Ex3-GT	.8540	.7527	.9407	87	2	87
2	L	S1-S2-S3-S4	.7765	.6935	.8566	87	4	522
2	L	S1-GT	.9232	.8464	.9816	87	2	87
2	L	S2-GT	.9150	.8361	.9780	87	2	87
2	L	S3-GT	.8770	.7852	.9560	87	2	87
2	L	S4-GT	.7695	.6434	.8802	87	2	87
2	R	Ex1-Ex2-Ex3	.5680	.4706	.6637	87	3	261
2	R	Ex1-Ex2	.4847	.3239	.6407	87	2	87
2	R	Ex1-Ex3	.5554	.4038	.6988	87	2	87
2	R	Ex2-Ex3	.6893	.5119	.8455	87	2	87
2	R	Ex1-GT	.6494	.5138	.7743	87	2	87
2	R	Ex2-GT	.8343	.7056	.9409	87	2	87
2	R	Ex3-GT	.8848	.7733	.9755	87	2	87
2	R	S1-S2-S3-S4	.7043	.6056	.7956	87	4	522
2	R	S1-GT	.9084	.8188	.9782	87	2	87
2	R	S2-GT	.9060	.8141	.9775	87	2	87
2	R	S3-GT	.7906	.6555	.9067	87	2	87
2	R	S4-GT	.7108	.5463	.8509	87	2	87

Taskid	Side	Group	Alpha	LL95%CI	UL95%CI	Units	Observrs	Pairs
3	L	Ex1-Ex2-Ex3	.4870	.3819	.5892	87	3	261
3	L	Ex1-Ex2	.4219	.2486	.5846	87	2	87
3	L	Ex1-Ex3	.4754	.3026	.6395	87	2	87
3	L	Ex2-Ex3	.5744	.3957	.7428	87	2	87
3	L	Ex1-GT	.6407	.4964	.7747	87	2	87
3	L	Ex2-GT	.7596	.6131	.8870	87	2	87
3	L	Ex3-GT	.8308	.7087	.9348	87	2	87
3	L	S1-S2-S3-S4	.6255	.5255	.7196	87	4	522
3	L	S1-GT	.7451	.6132	.8646	87	2	87
3	L	S2-GT	.7528	.6123	.8755	87	2	87
3	L	S3-GT	.7571	.6252	.8772	87	2	87
3	L	S4-GT	.6707	.5096	.8118	87	2	87
3	R	Ex1-Ex2-Ex3	.4080	.2917	.5259	87	3	261
3	R	Ex1-Ex2	.3138	.1135	.5049	87	2	87
3	R	Ex1-Ex3	.4411	.2537	.6174	87	2	87
3	R	Ex2-Ex3	.4505	.2391	.6452	87	2	87
3	R	Ex1-GT	.6490	.5062	.7752	87	2	87
3	R	Ex2-GT	.6619	.4873	.8249	87	2	87
3	R	Ex3-GT	.8298	.7139	.9310	87	2	87
3	R	S1-S2-S3-S4	.6120	.5191	.7013	87	4	522
3	R	S1-GT	.7154	.5857	.8382	87	2	87
3	R	S2-GT	.7524	.6265	.8638	87	2	87
3	R	S3-GT	.7736	.6415	.8827	87	2	87
3	R	S4-GT	.7560	.6222	.8747	87	2	87
4	L	Ex1-Ex2-Ex3	.7534	.6827	.8166	87	3	261
4	L	Ex1-Ex2	.7873	.6976	.8681	87	2	87
4	L	Ex1-Ex3	.7646	.6471	.8697	87	2	87
4	L	Ex2-Ex3	.7084	.5857	.8210	87	2	87
4	L	Ex1-GT	.9184	.8516	.9742	87	2	87
4	L	Ex2-GT	.8631	.7850	.9306	87	2	87
4	L	Ex3-GT	.8605	.7716	.9377	87	2	87
4	L	S1-S2-S3-S4	.8644	.8098	.9140	87	4	522
4	L	S1-GT	.9402	.8804	.9860	87	2	87
4	L	S2-GT	.9266	.8608	.9822	87	2	87
4	L	S3-GT	.8925	.8103	.9571	87	2	87
4	L	S4-GT	.8731	.7779	.9542	87	2	87

Taskid	Side	Group	Alpha	LL95%CI	UL95%CI	Units	Observers	Pairs
4	R	Ex1-Ex2-Ex3	.7300	.6610	.7926	87	3	261
4	R	Ex1-Ex2	.7493	.6630	.8310	87	2	87
4	R	Ex1-Ex3	.7688	.6600	.8648	87	2	87
4	R	Ex2-Ex3	.6740	.5387	.7910	87	2	87
4	R	Ex1-GT	.9052	.8321	.9645	87	2	87
4	R	Ex2-GT	.8278	.7486	.9023	87	2	87
4	R	Ex3-GT	.8703	.7875	.9424	87	2	87
4	R	S1-S2-S3-S4	.8625	.8024	.9161	87	4	522
4	R	S1-GT	.9428	.8857	.9857	87	2	87
4	R	S2-GT	.9428	.8857	.9857	87	2	87
4	R	S3-GT	.9047	.8315	.9670	87	2	87
4	R	S4-GT	.8501	.7591	.9292	87	2	87
5	L	Ex1-Ex2-Ex3	.5324	.4187	.6385	87	3	261
5	L	Ex1-Ex2	.4695	.2895	.6482	87	2	87
5	L	Ex1-Ex3	.5060	.3380	.6712	87	2	87
5	L	Ex2-Ex3	.6271	.4387	.7953	87	2	87
5	L	Ex1-GT	.6542	.5004	.7957	87	2	87
5	L	Ex2-GT	.8068	.6709	.9235	87	2	87
5	L	Ex3-GT	.8414	.7108	.9478	87	2	87
5	L	S1-S2-S3-S4	.6544	.5446	.7538	87	4	522
5	L	S1-GT	.8388	.7228	.9320	87	2	87
5	L	S2-GT	.8354	.7171	.9305	87	2	87
5	L	S3-GT	.7389	.5729	.8807	87	2	87
5	L	S4-GT	.6284	.4414	.7981	87	2	87
5	R	Ex1-Ex2-Ex3	.6006	.4889	.7004	87	3	261
5	R	Ex1-Ex2	.6068	.4360	.7561	87	2	87
5	R	Ex1-Ex3	.5564	.3769	.7236	87	2	87
5	R	Ex2-Ex3	.6389	.4562	.7978	87	2	87
5	R	Ex1-GT	.7592	.6250	.8767	87	2	87
5	R	Ex2-GT	.8372	.7149	.9394	87	2	87
5	R	Ex3-GT	.8052	.6607	.9249	87	2	87
5	R	S1-S2-S3-S4	.6326	.5242	.7329	87	4	522
5	R	S1-GT	.7870	.6632	.8935	87	2	87
5	R	S2-GT	.7746	.6446	.8888	87	2	87
5	R	S3-GT	.7928	.6499	.9080	87	2	87
5	R	S4-GT	.6846	.5027	.8453	87	2	87

Taskid	Side	Group	Alpha	LL95%CI	UL95%CI	Units	Observers	Pairs
6	L	Ex1-Ex2-Ex3	.6889	.6107	.7613	87	3	261
6	L	Ex1-Ex2	.6646	.5454	.7780	87	2	87
6	L	Ex1-Ex3	.6959	.5891	.7987	87	2	87
6	L	Ex2-Ex3	.7122	.5654	.8405	87	2	87
6	L	Ex1-GT	.8047	.7141	.8872	87	2	87
6	L	Ex2-GT	.8503	.7492	.9373	87	2	87
6	L	Ex3-GT	.8957	.8158	.9605	87	2	87
6	L	S1-S2-S3-S4	.7967	.7345	.8555	87	4	522
6	L	S1-GT	.8834	.8035	.9489	87	2	87
6	L	S2-GT	.8687	.7859	.9409	87	2	87
6	L	S3-GT	.8549	.7696	.9291	87	2	87
6	L	S4-GT	.7911	.6722	.8955	87	2	87
6	R	Ex1-Ex2-Ex3	.6977	.6207	.7696	87	3	261
6	R	Ex1-Ex2	.6538	.5275	.7627	87	2	87
6	R	Ex1-Ex3	.6860	.5724	.7905	87	2	87
6	R	Ex2-Ex3	.7620	.6365	.8713	87	2	87
6	R	Ex1-GT	.7890	.7019	.8682	87	2	87
6	R	Ex2-GT	.8729	.7818	.9518	87	2	87
6	R	Ex3-GT	.9197	.8598	.9721	87	2	87
6	R	S1-S2-S3-S4	.8883	.8406	.9315	87	4	522
6	R	S1-GT	.9054	.8409	.9608	87	2	87
6	R	S2-GT	.8867	.8179	.9461	87	2	87
6	R	S3-GT	.8926	.8262	.9502	87	2	87
6	R	S4-GT	.8429	.7646	.9115	87	2	87

## C.6 Weakness grading features

feature label	#	feature label	#
T0_I1_n3_A___subtractabs	2240	T2_I1_n0_L_3_p11p56p55_triangle	1984
T0_I1_n4_A___subtractabs	2240	T2_I1_n0_A_3_p37p38p44p43_dist	1984
T0_I1_n1_L_2_p45_motion2d	1984	T2_I1_n0_R_3_p17p18_dist	1980
T0_I1_n0_LM2_p43p47_dist	1984	T2_I1_n0_LM2_p23p44_dist	1980
T0_I1_n1_R_2_p41_motion2d	1984	T2_I1_n0_R_3_p03p48p59_triangle	1972
T0_I1_n1_R_2_p40_motion2d	1984	T2_I1_n0_L_3_p56p64p63_triangle	1968
T0_I1_n0_R_3_p31p48p49_triangle	1984	T2_I1_n0_A_3_p01p36p15p45_dist	1884
T0_I1_n0_R_3_p49p50p60_triangle	1984	T2_I1_n0_A_3_p60p64p62p64_dist	1848
T0_I1_n0_L_2_p54_motion2d	1892	T2_I1_n0_R_3_p21p27_dist	1792
T0_I1_n0_L_3_p35p34_dist	1728	T2_I1_n0_A_3_p03p48p59p13p54p55_triangle	1788
T0_I1_n1___M3_p48p54_Sjogreen_MWA	1612	T2_I1_n0_L_2_p22p43p42_triangle	1716
T0_I1_n1_R_2_p36_motion2d	1608	T2_I1_n0_L_3_p52p63_dist	1668
T0_I1_n0_LM2_p44p46_dist	1604	T2_I1_n0_L_3_p22p27_dist	1656
T0_I1_n0_L_3_p55p56_dist	1556	T2_I1_n0_L___subtractabs	1652
T0_I1_n0_A_3_p00p17p16p26_dist	1496	T2_I1_n0_R_2_p19p38_dist	1640
T0_I1_n1___N3_p48p54_Sjogreen_MWA	1492	T2_I1_n0_R_3_p37p38p41_triangle	1584
T0_I1_n0_A_3_p48p65p59p54p63p55_triangle	1484	T2_I1_n0_L_3_p53p52p62_triangle	1532
T0_I1_n0_R_3_p32p33_dist	1348	T2_I1_n0_R_2_p19p37p38_triangle	1508
T0_I1_n1___M2_p48p54_Sjogreen_MWA	1316	T2_I1_n0_R_3_p18p37_dist	1496
T0_I1_n0_RM2_p38p40_dist	1316	T2_I1_n0_A_3_p02p41p14p46_dist	1476
T1_I1_n2_A___subtractabs	2180	T3_I1_n0_L___subtractabs	2976
T1_I1_n1_L_2_p22_motion2d	1984	T3_I1_n0_R___subtractabs	2976
T1_I1_n0_R_3_p17p18p37_triangle	1984	T3_I1_n0_L_3_p25p24p44_triangle	1984
T1_I1_n0_L_3_p53p62_dist	1984	T3_I1_n0_R_3_p17p18p37_triangle	1964
T1_I1_n0_R_3_p18p19p37_triangle	1964	T3_I1_n1_A_2_p58p56_motion2d	1928
T1_I1_n0_L_3_p54p53p62_triangle	1956	T3_I1_n0_L_3_p24p23p43_triangle	1844
T1_I1_n0_LM2_p22p47_dist	1824	T3_I1_n0_LN2_p11p67_dist	1836
T1_I1_n0_L_3_p15p45_dist	1768	T3_I1_n0_R_3_p51p58_dist	1724
T1_I1_n1_R_2_p20_motion2d	1724	T3_I1_n0_L_3_p55p67_dist	1704
T1_I1_n0_RM2_p38p39p40_triangle	1596	T3_I1_n0_LM2_p26p45_dist	1528
T1_I1_n0_A_3_p61p64p65p61p64p63_triangle	1564	T3_I1_n0_R_3_p48p59_dist	1528
T1_I1_n0_L_3_p25p24p44_triangle	1556	T3_I1_n0_RM2_p17p36_dist	1516
T1_I1_n0_L_3_p56p63_dist	1508	T3_I1_n0_A_3_p50p59p52p55_dist	1504
T1_I1_n0_R_3_p36p41_dist	1508	T3_I1_n0_RM2_p03p48p59_triangle	1480
T1_I1_n0_L_2_p25p24p44_triangle	1500	T3_I1_n4_A___subtractabs	1396
T1_I1_n0_R_3_p20p21p38_triangle	1480	T3_I1_n1___M2_p48p54_Sjogreen_MWA	1376
T1_I1_n0_A_3_p38p39p43p42_dist	1464	T3_I1_n0_RM2_p18p41_dist	1352
T1_I1_n0_L_3_p43p47p46_triangle	1432	T3_I1_n0_A_3_p50p61p52p61_dist	1320
T1_I1_n0_A_3_p40p41p47p46_dist	1292	T3_I1_n0_L_3_p35p34_dist	1308
T1_I1_n0_R_3_p49p60_dist	1276	T3_I1_n0_RM2_p18p36_dist	1236



feature label	#	feature label	#
T4_I1_n0_R_2_p48_motion2d	1984	T6_I1_n4_R___subtractabs	2132
T4_I1_n0_R_2_p66_motion2d	1984	T6_I1_n0_L_3_p55p56_dist	1984
T4_I1_n0_R_2_p59_motion2d	1984	T6_I1_n0_L_3_p54p63_dist	1984
T4_I1_n0_RM2_p58p57_dist	1984	T6_I1_n0_L_3_p54p53_dist	1984
T4_I1_n0_L_2_p33p52p51_triangle	1984	T6_I1_n0_R_3_p59p58_dist	1984
T4_I1_n0_LN2_p10p67_dist	1972	T6_I1_n0_A_3_p50p51p61p52p51p61_triangle	1984
T4_I1_n0_R_2_p33p50p51_triangle	1852	T6_I1_n0_R_2_p48p66_dist	1976
T4_I1_n0_RM2_p60p66_dist	1832	T6_I1_n0_R_3_p49p66_dist	1968
T4_I1_n0_R_3_p08p57p58_triangle	1824	T6_I1_n0_L_3_p54p56_dist	1780
T4_I1_n0_A_3_p36p41p45p46_dist	1776	T6_I1_n0_A_3_p17p36p26p45_dist	1584
T4_I1_n0_A_2_p60p61p65p62p61p63_triangle	1776	T6_I1_n0_A_3_p17p18p26p25_dist	1576
T4_I1_n0_R_2_p51p58_dist	1736	T6_I1_n0_RN2_p21p39_dist	1576
T4_I1_n0_R_3_p18p19p37_triangle	1656	T6_I1_n1_D_2_p48p54_Sjogreen_MWA	1552
T4_I1_n0_L_3_p51p56_dist	1572	T6_I1_n0_L_3_p10p54_dist	1544
T4_I1_n0_D_2_p48p54_Sjogreen_MWA	1516	T6_I1_n0_L_3_p56p57_dist	1480
T4_I1_n0_LN2_p11p56p55_triangle	1488	T6_I1_n0_LM2_p54p62p63_triangle	1464
T4_I1_n0_A_2_p48p66p54p67_dist	1476	T6_I1_n0_R_3_p48p49p60_triangle	1364
T4_I1_n0_L_3_p35p34_dist	1448	T6_I1_n0_R_3_p59p64_dist	1288
T4_I1_n0_R_3_p31p49p50_triangle	1332	T6_I1_n0_RM2_p59p66_dist	1252
T4_I1_n0_A_2_p48p60p65p54p62p63_triangle	1304	T6_I1_n0_A_2_p61p64p65p61p64p63_triangle	1232
T5_I1_n0_L___subtractabs	2976		
T5_I1_n1_R___subtractabs	2064		
T5_I1_n0_R_2_p51p61_dist	1980		
T5_I1_n0_L_2_p67_motion2d	1980		
T5_I1_n0_LN2_p54p67_dist	1968		
T5_I1_n0_RM2_p38p41_dist	1968		
T5_I1_n0_R_3_p48p66_dist	1936		
T5_I1_n0_L_2_p56p67_dist	1908		
T5_I1_n0_R_3_p51p60_dist	1852		
T5_I1_n0_R_3_p60p66_dist	1816		
T5_I1_n0_R_3_p00p36p48_triangle	1764		
T5_I1_n0_R_3_p60p59_dist	1748		
T5_I1_n0_LN2_p53p67_dist	1688		
T5_I1_n1_A_2_p21p22_motion2d	1664		
T5_I1_n0_RM2_p32p50_dist	1664		
T5_I1_n0_L_2_p51p62_dist	1404		
T5_I1_n0_A_3_p19p20p24p23_dist	1272		
T5_I1_n0_RM2_p48p65_dist	1208		
T5_I1_n0_RN2_p03p48p59_triangle	1188		
T5_I1_n0_A_2_p49p65p53p63_dist	1092		

**Table C.7:** The 20 most frequently selected facial features for estimating facial weakness per task for the right side of the face.

feature label	#	feature label	#
T0_I1_n3_A___subtractabs	2220	T2_I1_n0_A_3_p37p38p41p44p43p46_triangle	1984
T0_I1_n1_L_2_p45_motion2d	1984	T2_I1_n0_R_3_p21p27_dist	1984
T0_I1_n0_LM2_p43p47_dist	1984	T2_I1_n0_A_3_p37p38p44p43_dist	1984
T0_I1_n0_R_3_p31p48p49_triangle	1984	T2_I1_n0_R_3_p19p38_dist	1984
T0_I1_n1___M3_p48p54_Sjogreen_MWA	1984	T2_I1_n0_L_3_p26p25_dist	1984
T0_I1_n1_R_2_p37_motion2d	1980	T2_I1_n1___2_p29_motion2dcl	1972
T0_I1_n1___M2_p48p54_Sjogreen_MWA	1928	T2_I1_n0_LM2_p23p44_dist	1968
T0_I1_n1_R_2_p36_motion2d	1920	T2_I1_n0_A_3_p03p48p59p13p54p55_triangle	1940
T0_I1_n1_L_2_p46_motion2d	1896	T2_I1_n1___2_p28_motion2dcl	1940
T0_I1_n0_A_3_p59p58p55p56_dist	1896	T2_I1_n1___N3_p48p54_Sjogreen_MWA	1868
T0_I1_n0_R_3_p49p50p60_triangle	1888	T2_I1_n0_R_3_p03p48p59_triangle	1832
T0_I1_n0_L_3_p35p34_dist	1884	T2_I1_n0_A_3_p50p51p52p51_dist	1804
T0_I1_n1_R_2_p40_motion2d	1828	T2_I1_n0_RN2_p00p36_dist	1756
T0_I1_n0_A_3_p00p17p16p26_dist	1784	T2_I1_n0_R_3_p29p30_dist	1720
T0_I1_n0_L_2_p54_motion2d	1616	T2_I1_n3_A___subtractabs	1604
T0_I1_n1___N3_p48p54_Sjogreen_MWA	1572	T2_I1_n3_R___subtractabs	1596
T0_I1_n0_L_3_p55p56_dist	1528	T2_I1_n0_L_3_p56p64p63_triangle	1564
T0_I1_n1_D_2_p48p54_Sjogreen_MWA	1516	T2_I1_n0_R_3_p17p18_dist	1532
T0_I1_n0_R_3_p32p33_dist	1504	T2_I1_n0_R_3_p31p36p41_triangle	1504
T0_I1_n4_A___subtractabs	1448	T2_I1_n1_L_2_p23_motion2d	1480
T1_I1_n0_A_3_p38p39p43p42_dist	1984	T3_I1_n0_L___subtractabs	2976
T1_I1_n0_R_3_p18p19p37_triangle	1984	T3_I1_n0_R___subtractabs	2520
T1_I1_n1_L_2_p22_motion2d	1984	T3_I1_n4_A___subtractabs	2512
T1_I1_n0_L_3_p54p53p62_triangle	1984	T3_I1_n0_R_3_p18p19p37_triangle	1984
T1_I1_n0_L_3_p43p47p46_triangle	1968	T3_I1_n0_L_3_p25p24p44_triangle	1984
T1_I1_n0_A_3_p61p64p65p61p64p63_triangle	1968	T3_I1_n0_A_2_p48p60p65p54p62p63_triangle	1964
T1_I1_n0_RM2_p31p48p49_triangle	1968	T3_I1_n0_LN2_p54p67_dist	1856
T1_I1_n0_L_3_p53p62_dist	1956	T3_I1_n0_A_3_p02p36p14p45_dist	1816
T1_I1_n0_L_3_p25p24p44_triangle	1948	T3_I1_n0_LM2_p26p45_dist	1780
T1_I1_n0_L_2_p25p24p44_triangle	1828	T3_I1_n0_L_3_p55p63_dist	1756
T1_I1_n0_L_3_p23p22p43_triangle	1776	T3_I1_n0_R_3_p51p58_dist	1664
T1_I1_n0_LM2_p45p44p46_triangle	1760	T3_I1_n1_A_2_p58p56_motion2d	1528
T1_I1_n2_A___subtractabs	1732	T3_I1_n0_L_3_p26p25p44_triangle	1412
T1_I1_n0_R_3_p32p33_dist	1632	T3_I1_n0_L_2_p34_motion2d	1384
T1_I1_n0_L_3_p26p45p44_triangle	1524	T3_I1_n0_A_3_p20p21p38p23p22p43_triangle	1368
T1_I1_n0_R_3_p19p37p38_triangle	1460	T3_I1_n0_LM2_p44p47_dist	1344
T1_I1_n0_L_2_p13p54p55_triangle	1444	T3_I1_n1_A_2_p50p52_motion2d	1328
T1_I1_n0_L_2_p35p54p53_triangle	1428	T3_I1_n0_A_3_p50p59p52p55_dist	1308
T1_I1_n0_LM2_p26p45p44_triangle	1360	T3_I1_n0_A_3_p60p64p62p64_dist	1272
T1_I1_n0_L_3_p15p45_dist	1228	T3_I1_n0_A_3_p02p41p14p46_dist	1164

feature label	#	feature label	#
T4_I1_n0_L_2_p33p52p51_triangle	1984	T6_I1_n4_R___subtractabs	2932
T4_I1_n0_D_2_p48p54_Sjogreen_MWA	1976	T6_I1_n0_A_3_p39p40p42p47_dist	1984
T4_I1_n0_RM2_p58p57_dist	1968	T6_I1_n0_R_2_p48p66_dist	1984
T4_I1_n0_LM2_p54p64_dist	1880	T6_I1_n0_R_3_p59p58_dist	1984
T4_I1_n0_RM2_p51p57_dist	1860	T6_I1_n1___M2_p48p54_Sjogreen_MWA	1980
T4_I1_n0_R_3_p08p57p58_triangle	1836	T6_I1_n0_L_3_p54p53_dist	1972
T4_I1_n0_R_2_p59_motion2d	1816	T6_I1_n0_L_3_p55p56_dist	1960
T4_I1_n0_R_2_p66_motion2d	1760	T6_I1_n0_A_3_p50p51p61p52p51p61_triangle	1932
T4_I1_n0_L_3_p45p46_dist	1748	T6_I1_n0_R_3_p48p49p60_triangle	1908
T4_I1_n0_R_2_p33p50p51_triangle	1748	T6_I1_n0_R_3_p48p49_dist	1852
T4_I1_n0_LN2_p10p67_dist	1612	T6_I1_n0_RM2_p60p61p65_triangle	1816
T4_I1_n0_R_2_p49_motion2d	1600	T6_I1_n0_L_3_p10p54_dist	1808
T4_I1_n0_LM2_p62p67_dist	1548	T6_I1_n1_D_3_p48p54_Sjogreen_MWA	1752
T4_I1_n0_R_2_p51p58_dist	1536	T6_I1_n0_L_3_p54p53p62_triangle	1748
T4_I1_n0_R_2_p48_motion2d	1420	T6_I1_n0_L_3_p53p52_dist	1748
T4_I1_n0_LM2_p56p63_dist	1404	T6_I1_n1_D_2_p48p54_Sjogreen_MWA	1728
T4_I1_n3_A___subtractabs	1340	T6_I1_n0_R_3_p49p66_dist	1720
T4_I1_n1_R___subtractabs	1228	T6_I1_n0_R_3_p48p58_dist	1608
T4_I1_n1_A___subtractabs	1148	T6_I1_n0_L_3_p54p62_dist	1580
T4_I1_n0_R_3_p50p58_dist	1028	T6_I1_n0_R_3_p07p58_dist	1444
T5_I1_n0_L_2_p56p67_dist	1984		
T5_I1_n0_R_3_p48p66_dist	1984		
T5_I1_n0_LM2_p22p27_dist	1984		
T5_I1_n0_RM2_p50p60_dist	1892		
T5_I1_n0_R_3_p03p48p59_triangle	1744		
T5_I1_n0_L_3_p16p26p45_triangle	1740		
T5_I1_n0_L_2_p51p62_dist	1608		
T5_I1_n1_A_2_p21p22_motion2d	1496		
T5_I1_n0_RM2_p03p48_dist	1488		
T5_I1_n0_R_2_p31p36p41_triangle	1488		
T5_I1_n0_RN2_p03p48p59_triangle	1464		
T5_I1_n0_L_3_p62p55_dist	1416		
T5_I1_n0_L___subtractabs	1408		
T5_I1_n0_R_3_p60p59_dist	1388		
T5_I1_n0_RM2_p48p60_dist	1388		
T5_I1_n0_R_3_p51p64_dist	1308		
T5_I1_n0_A_3_p03p48p59p13p54p55_triangle	1236		
T5_I1_n0_L_2_p67_motion2d	1216		
T5_I1_n0_L_3_p55p56_dist	1124		
T5_I1_n0___2_p27_motion2dcl	1048		

**Table C.8:** The 20 most frequently selected facial features for estimating facial weakness per task for the left side of the face.

# Appendix D

## Software tools

During this project, a lot of time and effort was spent on developing software tools for the collection of participant data and management of the participant-database. These tools were written in either Python, Java, Matlab or C-sharp. The majority of the developed tools are explained in this section and are divided in the following categories: data collection tools, data management tools, annotation tools and feature extraction tools.

### D.1 Data collection

#### D.1.1 SCHEDULER TOOL

This simple and small Java command-line program was used for generating multiple random permutations for the 7 tasks described in section 2.1.2 and to automatically store them to files. The random permutations function as task orderings, which were used by the EXPERIMENTER TOOL during recording sessions to couple a participant identification number to a certain preset ordering. This tool was used to generate a series of 300 participants task orders that were saved to separate files with each a unique number ranging from 1 to 300. During a recording session the EXPERIMENTER TOOL would use the task order file that had the same filename as the identification number of the patient. E.g. for a participant with identification number 50, task order file with filename “50.txt” was used to determine the order and for a control with identification number 30, task order file with filename “30.txt” was used. The SCHEDULER TOOL had the following interface:

command	description
-h -help	displays help
-n (number)	generates a series of (number) task order files
-overwrite	overwrites task files if they are already present
-descramble	re-orders task order files so all relevant tasks come first

The directory for storing the task order files was determined by placing a text-file named “config.txt” in the same folder as the executable jar-file which contained the following line: “orderdir path/to/taskorders”, where “path/to/taskorders” was replaced with the actual directory path. The “-descramble” flag was used to reorder existing task orderings in such a manner that all the relevant tasks were positioned before the irrelevant tasks. This was used in practice

to remove tasks involving arm movements, which were dropped during the collection of the participant-database because the focus of this research shifted to solely facial characteristics. The following line gives an example of using the tool to generate 300 task order files, without overwriting existing task order files:

```
java -jar scheduler.jar -n 300
```

### D.1.2 EXPERIMENTER TOOL

The EXPERIMENTER TOOL was used to guide the data acquisition procedure from section 2.1.4 and was intended for the instructor. The software tool was required to cue participants using auditory tones so the recorded tones would correspond to the filmed tasks. The task orders would be loaded from the task order files generated by the SCHEDULER TOOL. Furthermore, the tool was required to keep track of the timing of the cues that the instructor would give during the session and to store the cue timings and participant information once finished. A graphical user interface was designed that could fulfill these requirements, which consists of two important windows: the “main window” for entering participant information and the “session window” which would serve to give cues during a recording session.

The “main window” graphical interface is shown in Figure D.1 and it consists of a button for starting a new session and four entry fields for entering participant information, that are from top to bottom: a control group check box, a participant number drop down menu, an age text field and a check box for the gender. On starting the EXPERIMENTER TOOL the “main window” would be shown and could be used to start a new recording session with a participant after entering participant information. A new session could be attempted to start on pressing the button. However, if the entered participant already had a completed session a pop-up window would appear to inform the user about it. Notice that there were two counts for the participants, one for the patients, which would be used when the control group check box was unchecked and one for the controls which was used when the control group check box was checked. Thus, if a new session was started for a patient with number 30 and there would be a finished session for a control with number 30 this would be perfectly fine, but if a new session was started for the control with number 30, that would yield the pop-up warning. This security mechanism was introduced to prevent accidentally recording over previously recorded participant logs.

The session window is shown in Figure D.2 and consists of a list with the task order for the participant with the currently active task selected, a text field displaying the current task, a text field displaying the coming task, a text field displaying the time the current task has been active in seconds and three buttons that are from top to bottom: next task button for moving on to the next task, a test sound button for testing the auditory cues and an end session button for ending and saving the session. After successfully starting a new session from the “main window”, the “session window” would appear and replace the former window entirely. From this window the instructor could cue coming tasks, do a trail run with a participant and complete a full session. The main interaction of the instructor was with the three buttons, which will be explained next.

Upcoming tasks within the task order could be cued by pressing the next task button, which would instantly trigger the following events in order:

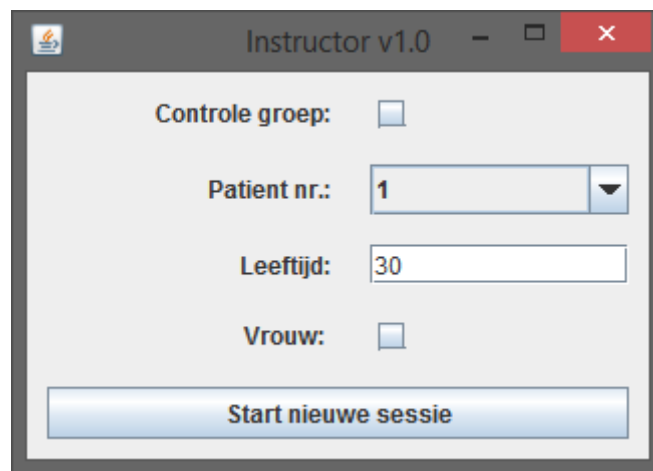
1. It would play a tone corresponding to the next task
2. It would reset the task timer to zero
3. It would increase the task index by one

4. It would update the graphical user interface, by showing the newly selected task within the list and by refreshing the text fields

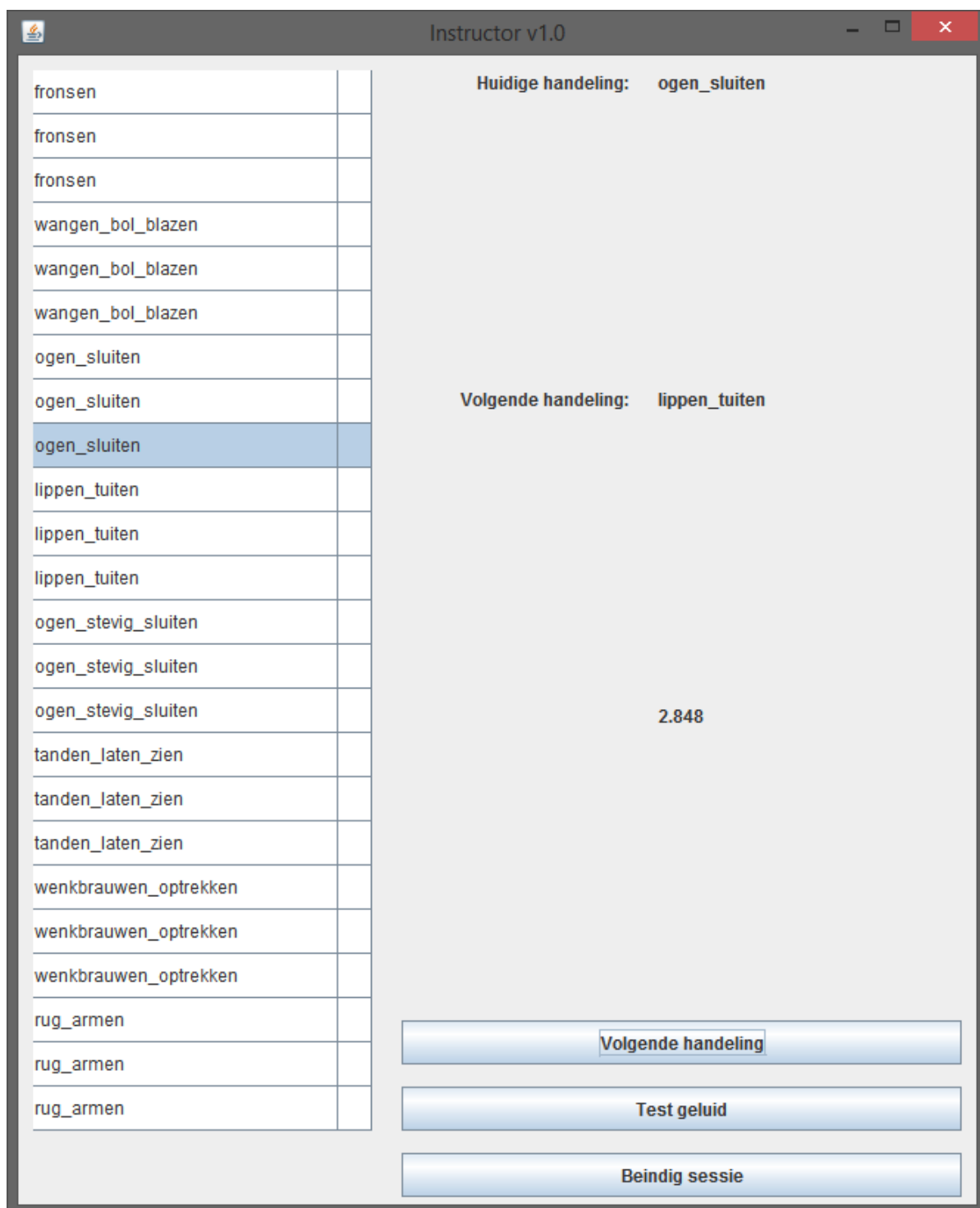
The test sound button would simply play the tone corresponding to the currently selected task, but would not advance to the next task. The button had multiple uses: it could be used by the instructor to test if the sound was audible or loud enough, it could be used at the beginning of a session by the instructor to practice the recording routine with the participant and it could be used during a session to make the participant repeat an action if he or she failed to perform one correctly.

Finally, the end session button could be pressed to end the session, which would write the participant and session information to a participant log file and consecutively return to the “main window”. If the instructor would press the button after all tasks from the task list were cued it would simply save the information and return to the “main window”, otherwise the instructor would be informed that some tasks were not cued and the instructor would be asked to reconfirm aborting the session using a confirmation pop-up with “Yes” and “Cancel” buttons.

During a session every click on the test sound and next task buttons were stored in a list together with the passed time since the beginning of the session and the index of the current task. This session timing information was used to later find the exact timing of the tasks within the videos. When a session was ended the entered participant information from the “main window” together with the session timing information from the “session window” were saved to a single participant log file. The location for placing the log files was determined in similar fashion to the SCHEDULER TOOL by placing a text file named “config.txt” next to the executable jar file containing the following line: “patientsdir path/to/participantlogs”, were “path/to/participantlogs” was replaced with the actual directory path to the participant logs.



**Figure D.1:** The “main window” graphical user interface for the EXPERIMENTER TOOL.



**Figure D.2:** The “session window” graphical user interface for the EXPERIMENTER TOOL.

## D.2 Data management

For managing and visualizing data from the participant-database a special Python library was developed, that contained a set of modules, tools and scripts, to manage, visualize and manip-

ulate video files (.mp4), and Kinect files (.xef / .xrf) and to label the data. For manipulating the video files the open source FFmpeg cross-platform command line tool (version N-67118-g08e6832) [67] was used and for managing Kinect files a set of new tools were developed using the Microsoft Kinect 2 SDK (version 2.0\_1409) [68]. Visualizations and image processing were accomplished using matplotlib from the SciKitLearn package and OpenCV 2.0.

After new video data was recorded using the PowerShot cameras and the Kinect 2 sensor the data would be placed on a single external hard disk to first tag the files before further processing and making backups. The participant logs would be placed under a “participants” folder, the Kinect 2 files would be placed under a “kinect” folder and the PowerShot videos would be placed respectively under “front black”, “left gray” and “right blue”. The PARTICIPANT INDEXER would be used to index the new participant logs under the “participant” folder so the participant order would include the new participants. Consecutively, the DATA TAGGER would be used to match all novel recorded materials under the other folders the right participant tags. After these steps were correctly completed the video materials could be distributed for backups over the other disks.

The recorded Kinect 2 streams were relatively large, i.e. 3 minutes of recording used up to approximately 30 GB. Therefore, the data were compressed using 7zip compression to one third of the original file size and distributed among three external hard drives, where two had storage capacities of 3TB and one had a storage capacity of 4TB. To be able to conveniently manage the data across disks the data would be stored using a *data tree*, which was a folder structure that served as a reference for the tools within this section to operate on and to locate files. In this way all programs only needed a path to the root of the data tree to locate any participant-database related file. The general layout of the data tree and the stored files can be found in Table D.1.



path	file description	file type
kinect/	Kinect 2 video files (front)	.xrf/.xef/.7z
kinect/wav/	Kinect 2 extracted audio	.wav
front black/	Powershot video files (front)	.mp4
left gray/	Powershot video files (left)	.mp4
right blue/	Powershot video files (right)	.mp4
(videofolder)/cues/	audio frequency over time	.npy
(videofolder)/cues/	video temporary cues	.json
(videofolder)/cues/	video final cues	.json
participants/	participant logs	.txt
participants/ordering/	participant order	.txt
participants/ordering/order/	task orderings	.txt
participants/cues/	participant order cues	.json
participants/labels/	expert labels	.csv
analyses/featuresets/	FEATURE EXTRACTOR cached feature sets	.csv
analyses/featurematrices/	FEATURE EXTRACTOR cached feature matrix	.txt

**Table D.1:** All paths within a data tree and the stored file types within those folders. When the text “(video folder)” is within a path it can be replaced by any one of the following video folders: “kinect”, “front black”, “left gray” or “right blue”

### D.2.1 PARTICIPANT INDEXER

The PARTICIPANT INDEXER is a Python script developed to extract all relevant information from the participant logs produced by the EXPERIMENTER TOOL. This information was the participant information (identification tag, gender, age and if the participant was a control) and the session timing information (a list of button presses with corresponding tasks and time it was pressed from the start of the session in milliseconds). All participant information was combined and stored in a single ordered list, the participant order, in order of increasing file modification date, i.e. starting with the information from the first participant recorded and ending with last participant recorded. The session timing information was stored to a separate file for later use by the other data management tools.

A typical usage of the tool is shown below, where a user would re-index the participant order for a data tree stored under “f:/”. The only required argument was the location of the data tree.

```
python participantindexer.py f:/
```

### D.2.2 DATA TAGGER

The DATA TAGGER was a Python command line tool used for providing unlabeled video files with a tag. The tool was mainly used after having raw unlabeled video materials within their respective video folders and having indexed the participant information of the files using the PARTICIPANT INDEXER. The DATA TAGGER tool would then be able to propagate the right tags

from the participant order to the unlabeled video files. Which tags were assigned was determined by the participant order. The tags from the latest entries in the participant order would be assigned in decreasing order to the latest video files. Optionally the actual file modification dates could be used to match the participant order to the unlabeled video files, by assigning each participant video to the nearest modification time stored in the participant order.

The program requires at least one argument specifying the data tree to work on and two optional flags: “-forreal” and “-usetime”. When the program was run without the “-forreal” flag would simply list all the files within the video folders (“kinect/”, “front black/”, “left gray/”, “right blue/”) that had not yet been tagged and shows the tags assigned by the program. When the “-forreal” flag was given the program would list all the files, but would also actually rename the files with the assigned tags. To use the actual modification dates for tagging the unlabeled video files the “-usetime” flag could be set.

An example of using the tool is given below, here the tool is used to tag all the video files under data tree “f:/” based on the modification date, without actually renaming the files:

```
python datatagger.py f:/ -usetime
```

### D.2.3 DATA TOOL

The DATA TOOL is a Python command line interface, which operates on a selection of video files based on the available participants in the participant order. The selection contains by default all participants for all video folders, but several flags can be set to restrict the selection, i.e. the “-K”, “-L”, “-R” and “-F” flags restrict the set to the Kinect, the gray left, the right blue and the front black video folders respectively, the “-subset” flag with an additional argument can further restrict specific participants from the participant order, the “-no-back” flag restricts the selection to all videos of the front of the participants and the “-controls-only” flag removes all patient videos from the selection.

The major operations on the selection of video files are: extraction of the audio frequencies over time, extraction of temporary cues from the audio frequencies over time, cutting the videos using the final cues, compressing/decompressing video files and listing the selection for all video files. The listing operation would additionally indicate availability of files within the selected video folders for compressed video files, uncompressed video files, final cues and video cut files. The DATA TOOL has the following interface:

```
python datatool.py [data tree] [flags]
```

where “data tree” is the path to the root of a data tree and where “flags” can be any of the flags from Table D.2.

flag	description
-h -help	displays help
-K -L -R -F	iterate flags for video folders (kinect, left gray, right blue, front black)
-subset [set]	takes a subset [set] of all participants from the participant order
-no-back	removes all participant recordings from selection that contain the back
-controls-only	removes all patient recordings from selection
-freq	extract frequency files from the selected recordings
-no-boost	treats Kinect recordings the same as PowerShot recordings for frequency extraction
-cues	extract cues from frequency files
-testcues	test cues for number of cues found and certainty
-cut	cuts the recordings by the stored final cues
-remove-source	remove recording sources after cutting
-target [data tree ]	store cut files to a different data tree
-uncut	remove all cut files for the selection
-compress	compresses the selected recordings using 7zip
-decompress	decompresses the selected recordings
-log-extract [folder]	import FaceReader logs from [folder]
-list	lists current selection and shows available files

**Table D.2:** DATA TOOL flags and their effect.

#### D.2.4 CUE TOOL

The CUE TOOL is a Python command line tool that implements the video synchronization methods described in section 2.1.2 and provides an interface for applying those on audio signals and extracted cue files. There are two main approaches for finding the correct offsets within the videos: cue matching and time-frequency matching. Cue matching is performed through the “-corrdt” flag and requires that the auditory cues are already extracted. Time-frequency matching is performed through the “-corrdtfreq” flag and requires access to the auditory stream of the recording. Both methods match the participant cues to the extracted auditory information from the recordings. Hence, both methods require the participant cues to be available. To investigate if the video synchronization was correctly performed the cues can be visualized using the “-vis” flag through the matplotlib Python library. When the obtained corrections are satisfactory they can be saved to the data tree as final cues using the “-savecorr” flag. When the participant cues contain more than 21 cues or some of the set durations are invalid, it is possible through the tool to edit them through a text editor using the “-edit” flag.

The CUE TOOL has the following interface:

```
python cuetool.py [data tree] [participant tag] [flags]
```

where “data tree” is the path to the root of a data tree, “participant tag” refers to a participant and where “flags” can be any of the flags listed in Table D.3.

flag	description
-h -help	displays help
-K -L -R -F	flags for recordings to select (kinect, left gray, right blue, front black)
-vis	visualizes the cues for all selected recordings
-corrdt	find cue correction offsets using cue matching
-alpha [number]	sets the “corrdt” alpha parameter to “number”, which should be a continuous number in the $[0, 1]$ interval.
-corrdtfreq	find cue correction offsets using time-frequency matching
-corrlabels	use the participant cue information to overwrite the type, duration and certainty for the final cue files. Only works when no matching method is used.
-savecorr	save the found cue correction offsets to the data tree
-nopool	serializes the jobs instead of running multiple processes
-edit	opens the cue-files for edit in a text editor
-editfinal	opens the final-cue files for edit in a text editor

**Table D.3:** CUE TOOL flags and their effect.

### D.2.5 Kinect 2 tools

The Kinect 2 tools were developed as command line tools in C-Sharp for managing Kinect recordings (.xrf .xef) and were specifically written because certain functionality was missing from the Kinect 2 SDK. The Kinect 2 tools consist of three programs: the AudioExtractor, the Cutter and the ImageGrabber. The AudioExtractor extracts the audio stream from a Kinect recording and writes it to a .wav file. The Cutter cuts Kinect recordings related to a single task into smaller files. Finally, the ImageGrabber can extract the color and depth information from a given Kinect stream for a specific frame given time offset.

The tools can be called from the command line or from the project library. The latter method is preferred, since it will keep executing the tool until the job is performed successfully. In this way, calling the tools from the project library can be used to serialize Kinect recording operations while handling unexpected failures. The interface for the tools in the project library can be found under “lib/kinect.py”.

It must be noted that the mentioned software tools are currently not reliable, i.e. they sometimes crash during execution, yet function properly at other times. Another big limitation of the current version of the Cutter tool is that the resulting cut files do not correctly retain the cutting onsets, i.e. the recordings start a few milliseconds earlier or later. This problem might be solved by annotating the cue positions directly in the Kinect 2 recordings using meta-data and extracting task specific information directly from the original full-length recording. However, the file size will remain very large that way.

#### AudioExtractor

The AudioExtractor extracts the audio stream from a Kinect file and writes it to a wave file. It uses the Kinect tools environment to read the audio stream directly from the Kinect 2 recording. Subsequently, the audio stream is exported to a mono-channel 16000 samples per second

wave file using IEEE float encoding.

The syntax for executing the AudioExtractor from the command line is:

```
XEFAudioExtractor.exe (source) (target) [-r]
```

where “source” should be replaced with the filename of the input Kinect recording (.xef or .xrf) and the “target” should be replaced with a target wave file filename. When the optional “-r” flag is set, it overwrites the target file if it exists.

## **Cutter**

The Cutter tool is used for cutting the raw Kinect 2 recordings into smaller parts per task and iteration. The tool uses the Kinect tools environment to create a destination file and subsequently copies all stream information from a source Kinect recording to it. The timing offset and the duration are given in milliseconds.

The syntax for executing the Cutter tool from the command line is:

```
XEFCutter.exe (source) (target) [-t (start time)] [-d (duration)] [-r] [-c]
```

where “source” should be replaced with the filename of the input Kinect recording (.xef or .xrf) and the “target” should be replaced with the filename of the output Kinect recording. The “-t” flag followed by an integer indicates the starting offset time in milliseconds in the recording. The “-d” flag followed by an integer indicates the duration of the cut recording in milliseconds. When the optional “-r” flag is set, it overwrites the destination file if it exists. When the “-c” flag is set, the program ignores minimal problems encountered during cutting the raw recording.

## **ImageGrabber**

The ImageGrabber extracts a color image and the corresponding depth information (as a point cloud) for a target frame from a given cut Kinect recording. This result is achieved by playing back the recorded streams around the relative time for the entered frame and using the build in coordinate mapper to get the corresponding depths for each pixel within the color image. For storage efficiency it is also possible to define a subregion to export for the pointcloud. This was used in combination with the user defined crop box.

The syntax for executing the ImageGrabber tool from the command line is:

```
XEFIImgGrab.exe (source) (time offset) [-ci (color image)] [-pc (point cloud)]  
[-region (x, y, w, h)]
```

where “source” should be replaced with the filename of the input Kinect recording (.xef or .xrf), the “time offset” should be set to the position to extract the color image and the pointcloud from in milliseconds. The “color image” and “point cloud” should be replaced with the filenames of the output color image file (.png) and the output pointcloud file (.txt). When the optional “-region” flag is set followed by 4 integers defining respectively the x, y, width and height of the subregion, it outputs only the defined region for the pointcloud instead of the whole picture.

## D.3 Annotation tools

There were multiple programs developed for positioning the timing offsets within the recordings and for placing the facial landmarks for the participant images at rest and at maximal expression. For manual placement of the timing offsets the VIDEO POSITION MARKER software was used and for manual placement of the facial landmarks the FACE MARKER software was used. Both programs were graphical user interfaces implemented in Python using OpenCV 2.0.

### D.3.1 VIDEO POSITION MARKER

The purpose of the VIDEO POSITION MARKER tool was to manually position and label the timing offsets within the cut recordings. The key idea behind this tool is to quickly and iteratively position the timing offsets and store those to the data tree for each participant, task and iteration combination. The tool allowed multiple videos to be queued in sequence to be processed serially. The tool was used to first label the timing offsets within the frontal videos and subsequently to extract the images related to the offsets from the Kinect 2 recordings using the ImageGrabber tool from section D.2.5. The VIDEO POSITION MARKER could be called from the command line in the following manner:

```
python interface/vidpositionmarker.py (data tree) (participant tag) [(task)]  
[(iteration)]
```

where “data tree” is the path to the root of a data tree and “participant tag” is the participant tag for a participant. Optionally, the “task” can be set to a task identification number (0-6) to only select a particular task and the “iteration” can be set to a number within (1-3) to select only a particular iteration. If task and iteration were omitted, all task and iteration videos would be queued for that task.

Figure D.3 shows the VIDEO POSITION MARKER interface. At the center the frontal Power-Shot recording is shown at the selected timing offset in the video. In the top left of the interface the current video frame offset is shown with the total duration of the video in number of frames. At the center top, the current evaluated video for a certain participant, task and iteration is shown. The first number indicates the current evaluated video, the second number indicates the total number of queued videos before the program automatically halts, the three codes next to that number are the participant tag, the task identification number and the iteration.

Using the “4” and “6” keys (the arrow keys on the numpad) it is possible to skip through the video and search for the frame at maximal expression and the frame at rest. Using the “r” key, the rest state can be tagged in the video and using the “e” key, the state at maximal expression can be tagged in the video. At the bottom of the GUI the duration of the video can be seen with markers for the current position within the video and the two tagged states. Using the “k” key the corresponding images from the Kinect recordings are extracted and displayed to the left and the right from the image at the center. Finally, pressing the “s” key stores the selected rest and active states to the data tree. For a complete listing of all key commands see Table D.4.



**Figure D.3:** The graphical user interface for the video position marker.

key	function
7	Move the current video offset 1 frame back
9	Move the current video offset 1 frame further
4	Move the current video offset 5 frames back
6	Move the current video offset 5 frames further
r	Set the rest state to the current video offset
e	Set the extreme state to the current video offset
o	Open and edit the cut Kinect 2 recording directly
t	Move the active video offset between the rest and active states
s	Save the current selected rest and active states
k, j	Extracts the Kinect image data
n	Move to the next video in the queue
8	Move the center video image up (visuals only)
2	Move the center video image down (visuals only)
q	Quit the program immediately

**Table D.4:** Keyboard commands and their function.

### D.3.2 DRMF fitter

The DRMF fitter is a Matlab script that applies a DRMF [52] based landmark pre-fitting on the extracted facial images from the VIDEO POSITION MARKER. It is highly recommended to run this script before working with the FACE MARKER tool as it gives a good pre-fitting of the landmarks to manually correct. The script batch processes input face images at rest and at maximal expression and skips files that have already been processed.

The script can be found under “tools/DRMF/Extractor.m”. Before running the script, the “export\_path” and “image\_path” variables should be set to the correct output and image input directories respectively.

### D.3.3 FACE MARKER

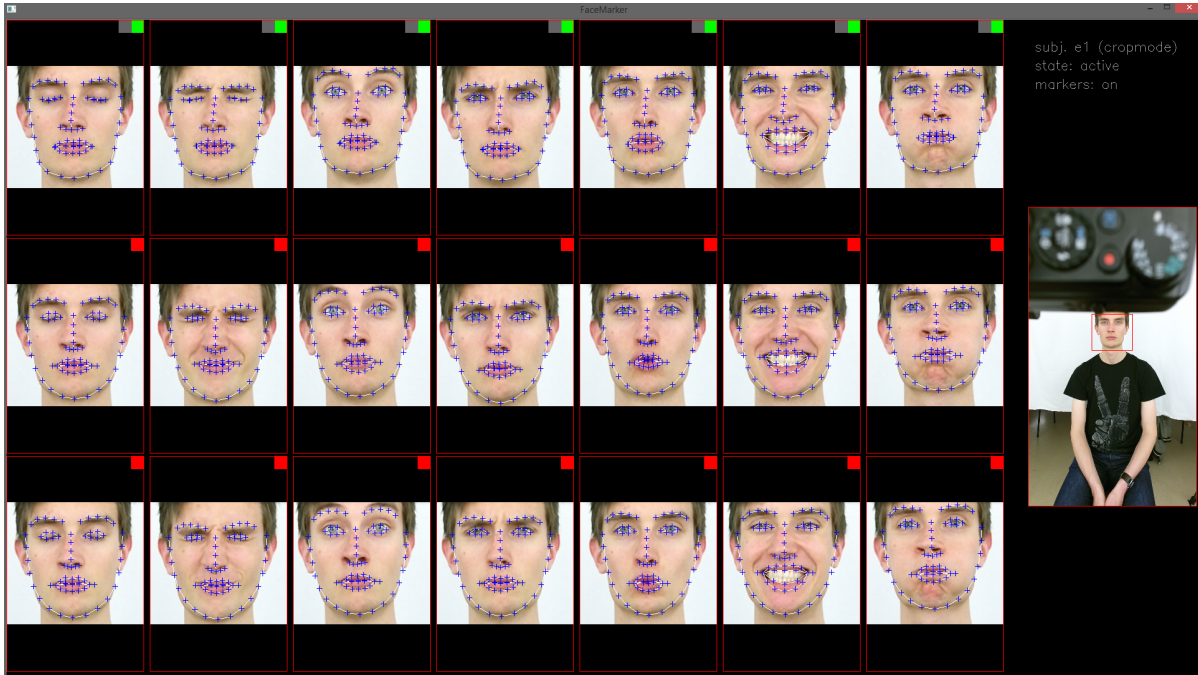
The FACE MARKER software was used to manually position the facial landmarks for the images at rest and at maximal expression for each participant. By default, the tool was set to work with extracted Kinect 2 state images from a data tree, but could be used for extracted images from the cut frontal videos from the same data tree as well. The tool had two distinct operation modes: the “overview mode” which gives an overview of all tasks and iterations for a participant and the “face mode” which zooms in on a particular task and iteration for a participant and enables landmark editing. To run the FACE MARKER tool, the following command needs to be administered:

```
python interface/facemarker.py (data tree) [(participant tag)]
```

where “data tree” should be a path to the root of a data tree. When a “participant tag” is entered, the program tries to load that participant first. If that fails or if no “participant tag” is entered, the first participant from the participant order is loaded.

Figure D.4 shows the interface of the “overview mode”, where an overview of all the tasks and iterations can be seen in one screen, for either the images at rest (rest state) or the images at maximal expression (active state) for a particular participant. The rows indicate the different iterations and the columns indicate the different tasks ordered by identification number. The user could switch between the active and rest state by using the “t” key. In the “overview”-mode screen to the far right a subregion called a crop box within the extracted image could be defined by drawing a rectangle using the mouse, to select and zoom in on the relevant area from the images. The crop box for this participant for the data tree is saved on pressing the “s”-key in the “overview mode”. Furthermore, in this mode, a user can click on a particular task and iteration image to select it. Subsequently, by pressing the space-bar key the tool will transfer to the “face mode” and change screens.





**Figure D.4:** The graphical user interface for “overview mode” of the Face marker tool.

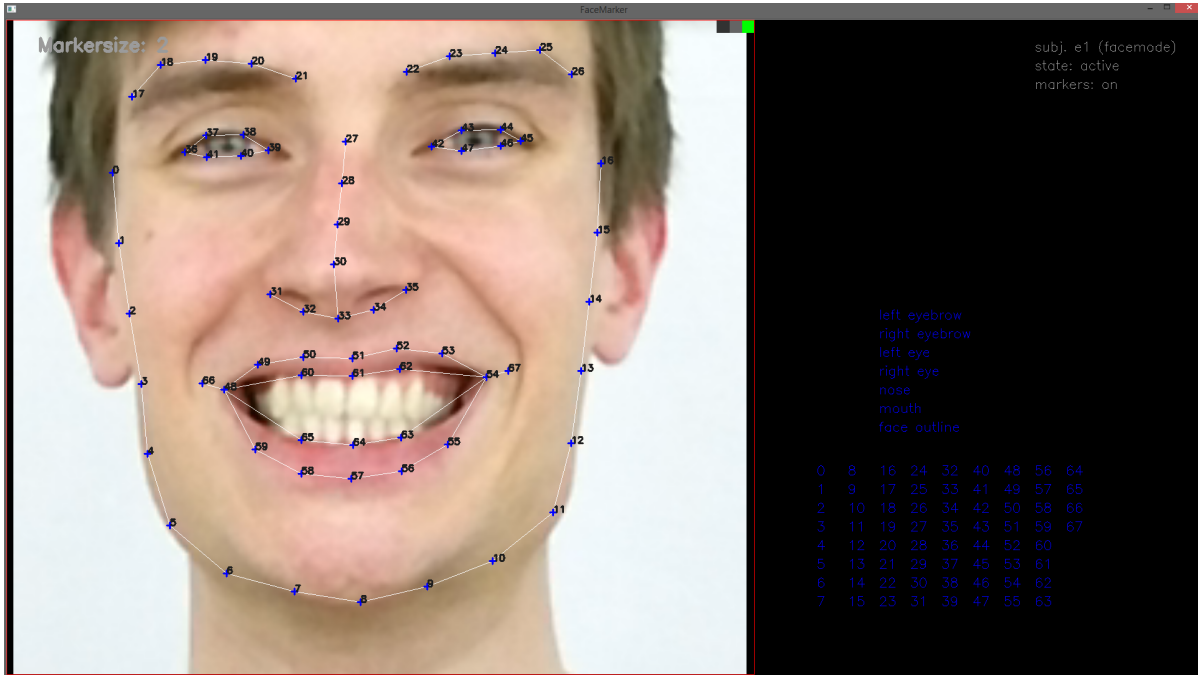
Figure D.5 shows the interface of the “face mode”. This mode has an enlarged image view on the left for a selected task and iteration, which enables viewing and editing of the the landmark positions. The three small squares at the top of the landmark image indicate in order: whether the landmarks have been edited since the last save, if the landmarks have been edited before (i.e. if the landmarks are different from the initial face fitting) and if the image has been flagged as finished (green) or unfinished (red). The “face mode” has four different submodes for editing the landmarks: the selection mode (“q” key), the movement mode (“w” key), the scale mode (“e”) and the rotation mode (“r”).

Within the selection mode, landmarks can be added or removed to the current landmark selection by mouse. The mouse can be used to select a rectangle area in the image, or can click on an item from the groups on the bottom right of the interface screen. On a click (i.e. when making a new selection) the current selection is set to the new selection by default. However, when the “+” key is pressed, the selection is added to the current selected landmarks and when the “-” key is pressed, the selection is removed from the current selected landmarks.

Within the movement mode, it is possible to move the selected landmarks by drag and drop within the image. Additionally, the landmarks can be moved by pressing the “4”, “6”, “8” and “2” directional keys on the numpad.

Within the scale mode the landmarks can be scaled using the “4”, “6”, “8” and “2” directional keys on the numpad. This scaling is proportional to the size of the bounding box of the selected landmarks. The scaling is performed relative to the center of this bounding box.

Within the rotation mode the landmarks can be rotated around the position of the mouse cursor when it is on the image. Rotation can then be performed by pressing the “4”, “6”, “8” and “2” directional keys on the numpad.



**Figure D.5:** The graphical user interface for “face mode” of the Face marker tool.

From the FACE MARKER tool it is also possible to switch between participants without first quitting the program. This part of the tool is called the “participant selector” window and can be opened and closed by pressing the “ ” key. Figure D.6 shows the “participant selector” window, which lists all participants by their participant tag. Next to the participant tags three colored rectangles provide information about the present files for the corresponding participant. The first rectangle indicates if the images are present, the second rectangle indicates if the landmark pre-fitting files are available and the last rectangle indicates if the manual landmark annotations are available. If a rectangle is bright green this means that all  $7 \times 3 \times 2 = 42$  files are present, if a rectangle is dark green it indicates that all files for the first iteration are present, if a rectangle is blue then a few files are present and if a file is red, then no files are present. The colored rectangles give information about which participants have been labeled and which participants should still be labeled and are meant to keep track of the current state of work. The user can select a participant by clicking near the participant tag of the desired participant. Subsequently, by pressing the “!” key, the participant images are loaded into the FACE MARKER tool.



**Figure D.6:** The “participant selector” window of the Face marker tool.

In addition to the mentioned functionality, many more utility functions were implemented. Table D.5 provides an overview of all the functionality and the associated keys. Note that some functionality is only available when the user is within a certain mode. The main modes are the “Overview” and “Face” modes and the edit modes are the “select”, “move”, “scale” and “rotate” modes.

key	main mode	edit mode	function
space	Both	All	Toggles between “overview” and “face” modes
h	Both	All	Toggles the landmarks visualizations on or off
l	Both	All	Toggles the landmarks on or off
[, ]	Both	All	Increase and decreases landmark size
:, ;	Both	All	Toggles landmark wire frame
?	Both	All	Sharpens a selected image or all images
t	Both	All	Toggles between rest and active states
!	Both	All	Reloads images for participant
, `	Both	All	Shows or hides the “participant selector” window
d	Both	All	Toggles the done flag for a selected image
backspace	Both	All	Resets to last saved landmarks for a selected image
	Both	All	Delete landmarks for selected image, reset to prefit
v	Both	All	Paste copied selection to selected image
#	Both	All	Edit the selected image with the VIDEO POSITION MARKER
escape	Both	All	Quit the program immediately
s	Overview	All	Saves the crop box
c	Overview	All	Copy all landmarks from selected image
8,4,6,2	Overview	All	Position the crop box
s	Face	All	Saves landmarks for selected image
c	Face	All	Copy all selected landmarks from selected image
a	Face	All	Select all landmarks
i, , `	Face	All	Toggle landmark numbers on or off
+	Face	All	Toggle additive selection mode
-	Face	All	Toggle subtractive selection mode
x, q	Face	All	Set edit mode to “select”
w	Face	All	Set edit mode to “move”
e	Face	All	Set edit mode to “scale”
r	Face	All	Set edit mode to “rotate”
8,4,6,2	Face	select, move	Position the landmarks
8,4,6,2	Face	scale	Scale the landmarks
8,4,6,2	Face	rotate	Rotate the landmarks

**Table D.5:** Keyboard commands and their function for the FACE MARKER tool.

## D.4 Feature extraction tools

The FEATURE EXTRACTOR tool, extracts relevant information from the participant-database and exports those for further evaluation and analysis. The system evaluation pipeline from section ?? is used for further evaluation and can be found under “scripts run\_eval\_pipeline.py”. The remainder of this section will describe the FEATURE EXTRACTOR.

#### D.4.1 FEATURE EXTRACTOR

The FEATURE EXTRACTOR is a command line tool which was developed in Python for this project. Its purpose was to facilitate rapid development and deployment of features. The tool can be used to calculate all the features and subsequently store all the features in a file as a feature matrix for later processing. Through the program it is possible to quickly prototype and develop different feature sets or recalculate the feature matrix when more participant data is collected.

The FEATURE EXTRACTOR works directly with the concept of a feature set, which is simply a collection of features that are grouped based on some shared characteristics. Through the tool, feature sets can be manipulated. First of all, using the “-calculate” and “-test” flags, the features for the feature sets can be calculated, which caches the resulting values to disk. Secondly, using the “-join” flag, cached feature sets values can be merged (or joined) as a feature matrix and written to a file for later use in the system evaluation pipeline. Finally, using the “-remove” flag, cached feature sets values can be removed. Once feature set results are cached, the program enforces that they are not calculated again. To recalculate the features for a feature set first the cached results need to be removed explicitly using the “-remove” flag.

The available feature sets are defined in “lib/features/\_\_init\_\_.py” as the module level variable called “featuresets”. Feature sets are implemented by sub-classing the abstract “FeatureSet” class. A single “FeatureSet” instance deals with a single participant for a particular data tree. All feature set implementations can be found under the folder “lib/features/”. One particularly useful class for developing a feature set is the “FeaturePropagater” class, which once properly configured, will automatically calculate asymmetry features and feature labels.

Besides the core functionality, the tool also supports some utility functions. By using the “-subset” flag followed by with a Python-style nd-array list selection (e.g.: “[3]”, “[0,1,2,4,6]”, “[7]”), it is possible to make a sub selection from all the participants from the participant order to perform the selected operations on. Following the same mechanic, using the “-features” flag followed by a similarly styled list selection, yields a sub selection from all the implemented feature sets to perform the selected operations on. Through the “-list” flag, it is possible to list all the selected feature sets and participants, indicating which feature sets have already been cached for each feature set and participant. When setting both the “-list” and the “-required” flags, the availability of the required files for calculating the feature sets are shown as well. Finally, by setting the “-config” flag followed by a filename, the default configuration file is ignored and the file specified by the filename is taken instead. The configuration file is passed to each feature set when calculating the features and can be used to influence how features are calculated. The settings within the configuration file are also written to the header of the feature matrix on the “join” operation for administrative purposes.

Below are some exemplar operations of the tool with their corresponding commands. For calculating the first available feature set under data tree “f:/” the following command can be entered:

```
python featureextracter.py f:/ -features [0] -calculate
```

The previous command will automatically skips cached feature set results. To recalculate the cached feature set files, the “-remove” flag should be added:

```
python featureextracter.py f:/ -features [0] -remove -calculate
```

The following command merges and exports all cached feature set results to a feature matrix:

```
python featureextracter.py f:/ -join
```

The next command lists all cached feature set results for the first ten participants:

```
python featureextracter.py f:/ -subset [:10] -list
```

It is also possible to chain multiple commands. For example, the following command recalculates all feature sets and subsequently exports them as a feature matrix:

```
python featureextracter.py f:/ -remove -calculate -join
```