

## Acknowledgments

To:

- Dr. Uzi Muallem (A.R.O.), for assisting in the manual, expertly cow scoring.
- Mr. Aharon Antler (A.R.O.), for helping in running the experiments in the barn.
- Mr. Shamay Yakobi (A.R.O.) the research-farm manager, for his valuable help in chasing the cows, always in a good-will mood.
- Dr. Ofer levi (BGU) for the image processing advice at early stages of the project.
- Dr. Yisrael Parmet (BGU) for the statistical advice and support.

And finally my supervisors:

- Dr. Ilan Halachmi (A.R.O.)
- Prof. Yael Edan (BGU)

For their support, patience and professional guidance.

## Abstract

Body condition scoring (BCS) is an important tool for dairy farm management that allows to estimate the mobilization of energy reserves of dairy cows. Despite intensive research on automating BCS using computer vision it is still a manual operation. 3D computer vision is a new and powerful tool that enables to capture images and movies with color and depth information. Analysis of such data achieves a higher level of topography understanding which is the main source of information for experts to estimate the body condition score.

The main goal of this thesis was to develop a three dimensional computer vision algorithm for a fully automated BCS evaluation system. An automatic data acquisition system consisting of a Microsoft Kinect camera and a Passive infrared motion detector which served as the camera trigger was constructed at the ARO, the Volcani center research farm in Bet-Dagan. Data was automatically acquired along four work days resulting in 422 correctly captured movies of 100 cows which served as the database for the research. Manual body condition scores were evaluated by a trained expert and were used as target values in the research ('gold standard test', Pepe, 2003).

The developed image processing algorithm included the following steps: **image restoration** – the removal of noise from all frames in movies; **object recognition and separation** – the identification of the cow in the image and her separation from the background; **movie and image selection** – selection of only movies and frames in movies that include the relevant data from the cow's surface; **image rotation** – the alignment of the cow parallel to the X axis; **image cropping and normalization** – the removal of irrelevant data from the image, setting the image size to 150x200 and normalizing image values.

Movies containing cows' backend were automatically selected resulting in 389 out of 422 potential movies and 0 out of 100 movies with irrelevant data selected during the image processing algorithm.

Twelve features were extracted from the movies: six average relative heights of ten pixel zones, the average of all pixel values, the standard deviation of all pixel values, the number of maximum points in the image (points with values larger than all 4 neighbors), the number of minimum points in the image (points with value smaller than all 4 neighbors), the maximum pixel value before normalization and average distance from the contour to the image left center. Two additional features were used from farm records: age and last recorded weight. The features were calculated for each image and for each movie (as mean, median, maximum or minimum of all images). The data was divided into a training set that included 80 cows and a test set that included 20 cows. Each data set including cows from the different BCS groups in with equal distributions. The image processing algorithm was able to produce 100% correct output images for 92.18% of potential movies.

Four polynomial regression prediction models were developed and optimized for performance using six main measures: mean absolute error (MAE), median absolute error (MDAE),  $R^2$  (R2), the number of correct classifications into six classes (A6), the number of classifications with up to 1 class misclassification (A6+1) and the number of correct classifications into four classes (A4). The models differed in the features they employed. Sensitivity analyses were conducted to fully understand the strengths and weakness of the models.

The polynomial regression model that included all features except for the X6 feature yielded the best results resulting with a MAE of 0.2809, MDAE of 0.2294, R2 of 0.7575, A6 of 0.4737, A6+1 of 0.9895 and A4 of 0.8105. The results were found to be as good as or better than state of the art research with 2D imaging proving the concept of automatic body condition scoring using 3D computer vision. Additional advantages of the system constructed was that it was fully automated (a main research achievement), had no background dependency, and used a low cost off-the-shelf camera.

# Table of Contents

1. Introduction .....	8
1.1. Description of the problem .....	8
1.2. Objectives .....	9
2. Literature Review.....	10
2.1. Body Condition Scoring (BCS) .....	10
2.2. Computer vision and image processing.....	13
2.3. Statistical analysis .....	22
3. Methods.....	24
3.1. Overview.....	24
3.2. Data acquisition .....	24
3.3. Image processing .....	28
3.4. Feature selection and extraction.....	32
3.5. Prediction models .....	33
3.6. Performance evaluation and validation .....	34
4. Algorithms .....	36
4.1. Image processing .....	36
4.2. Feature extraction .....	42
5. Results and Discussion.....	44
5.1. Automated image selection.....	44
5.2. Correlation analysis .....	44
5.3. Model selection .....	47

5.4.	Models results .....	48
5.5.	Movie data vs. image data .....	49
5.6.	Results correlation.....	50
5.7.	Confusion Matrices .....	51
5.8.	Models repeatability .....	60
5.9.	Error evaluation.....	61
5.10.	Sensitivity analyses.....	63
5.11.	Summary of models.....	67
5.12.	Comparison to previous research .....	69
6.	Conclusions and future research.....	71
6.1.	Conclusions.....	71
6.2.	Future research .....	72
7.	References.....	76
8.	Appendices .....	82
	Appendix A. All models and results.....	82
	Appendix B. Image processing pseudo code.....	88
	Appendix C. Correlations between models output and manual BCS.....	94
	Appendix D. Models repeatability tables and figures .....	99
	Appendix E. Research data .....	107
	Appendix F. Matlab codes used in research.....	107

## List of Figures

Figure 1 - Body condition scores and matching cow's appearances (A.J. Edmonson, 1989).....	11
Figure 2 - The tail head area .....	12
Figure 3 - Algorithm steps for computer vision BCS (Azzaro et al., 2009).....	22
Figure 4 - Research main steps .....	24
Figure 5 - Schematic layout of the system (in meters).....	26
Figure 6 – Training (a) and testing (b) data distributions .....	27
Figure 7 - Image before (a) and after (b) restoration .....	29
Figure 8 - Example 1 of initial image (a) and extracted object (b) .....	30
Figure 9 – Example 2 of initial image (a) and extracted object (b).....	30
Figure 10 – (a) original image (b) cropped normalized image .....	32
Figure 11 - Work flow of image processing algorithms.....	36
Figure 12 - Examples of holes in the image .....	37
Figure 13 - Original image (a) and rotated object (b).....	41
Figure 14 - Correlation between relative heights and BCS.....	46
Figure 15 - Model 3 correlation between the model output (Y axis) and the manual BCS (X axis). .....	50

## List of Tables

Table 1 - Decision chart for body condition score (Ferguson et al., 1994) .....	11
Table 2 - Examples of livestock agriculture computer vision applications.....	18
Table 3 - State of the art research on automatic BCS and BCS from images .....	21
Table 4 - Number of cows in each BCS group .....	26
Table 5 - Filters used for the object extraction .....	39
Table 6 - Model variable list .....	45
Table 7 - Correlation of model variables .....	46
Table 8 - Model details .....	47
Table 9 - Models results testing (training) data .....	48
Table 10 - Models results testing (training) data with group selection .....	48
Table 11 - Movie data model results.....	49
Table 12 - Model 1 confusion matrix with 6 classes for testing data.....	51
Table 13 - Model 2 confusion matrix with 6 classes for testing data.....	52
Table 14 - Model 3 confusion matrix with 6 classes for testing data.....	53
Table 15 - Model 4 confusion matrix with 6 classes for testing data.....	54
Table 16 - Model 1 confusion matrix with 6 classes for testing data with group selection.....	55
Table 17 - Model 2 confusion matrix with 6 classes for testing data with group selection.....	56
Table 18 - Model 3 confusion matrix with 6 classes for testing data with group selection.....	57
Table 19 - Model 4 confusion matrix with 6 classes for testing data with group selection.....	58
Table 20 - Standard deviation of cow's movies BCS in models 1 to 4 with and without group selection .....	60
Table 21 - Maximum difference of cow's movies BCS in models 1 to 4 with and without group selection .....	61
Table 22 - Evaluation of error rate on the testing set .....	62
Table 23 - Influence of training set size on model 1 R2 and MAE for testing data .....	64
Table 24 - K fold cross validation of model 1 .....	64
Table 25 - Type of movie data analysis .....	65
Table 26 - Image processing result with added noise .....	66
Table 27 - Prediction results of model 1 with added noise.....	66

# 1. Introduction

## 1.1. Description of the problem

Body Condition Scoring (BCS) is an important management tool in dairy farming. BCS evaluates the cow's energy reserves in a 5 point scale, where 1 is an emaciated cow and 5 is an obese cow (Ferguson et al., 1994). A cow's body condition score (BCS) and the change in a cow's BCS were found to be a good forecasting tool for problems with the cow's health, production and reproduction (Wildman et al., 1982). BCS is currently performed manually by an experienced farmer using tactile or visual methods (Hady et al., 1994). A manually performed BCS requires experienced personnel, it is time consuming and the result is a subjective measure that cannot be automatically collaborated in a computer report (Halachmi et al., 2008).

Several attempts have been made to automate the BCS process, but current practice is still manual (Azzaro et al., 2011). Several computerized vision systems have been developed for automatic BCS (Azzaro et al., 2011, Bercovich, 2012, Bewley et al., 2008, Halachmi et al., 2008). Most of the systems developed still require some level of manual classification and others are not fully operational due to further improvements necessary in several stages including image selection, cleaning the background and classification (Bercovich, 2012).

Three dimensional (3D) computer vision is one of the most studied techniques in the last years due to large amount of data received, which allows us to better understand the curvature of the picture, and the ability to easily deal with background issues (Cyganek & Siebert, 2011).

The **main objective** of this study is to develop an automatic BCS by applying 3D computer vision. The research aimed to provide an objective, automatic system that will produce an accurate BCS with no dependencies on the background.

## **1.2. Objectives**

The main objective of this study was to develop a 3D computer vision for a fully automated system of body condition scoring evaluation. The secondary objectives of the research were to:

1. Acquire 3D data automatically using computerized system.
2. Create an automatic image selection process that filters all data for appropriate inputs for the image processing algorithm.
3. Develop a 3D image processing algorithm to separate the cows object from the data and extract relevant features.
4. Develop a BCS prediction model.
5. Analyze and evaluate the results to estimate the abilities of the system and its preparedness for wide usage.

## **2. Literature Review**

### **2.1. Body Condition Scoring (BCS)**

#### ***2.1.1. BCS fundamentals***

Body Condition Scoring (BCS) is a scale of estimating the fat and muscle energy stored in a live animal (Edmonson et al., 1989). Body condition score is a useful management tool for evaluating cows' body condition (Wildman et al., 1982). For example, a cow's feeding, after a period of weight loss, should be more than their normal requirements to restore their normal body condition (Ferguson et al., 1994). Managing dairy cattle body reserves is a critical factor in herd management success. A cow's body weight alone is not enough to determine her body reserves because body weight depends also on other biological features such as the cow's height and skeleton frame size (Roche et al., 2004).

#### ***2.1.2. BCS methods and techniques***

There are two main approaches to evaluate BCS, visual and tactile, or a combination of the two (Hady et al., 1994). In most cases, cows are scored on a five point scale (1=emaciated to 5=obese). Changes in a cow's BCS can testify on the cow's production, reproduction and most importantly health (Hady et al., 1994). In order to maintain a healthy herd, cows must be scored on a regular basis (Ferguson et al., 1994). Figure 1 shows the change in a cow's appearance, in different body parts, for the five different body condition scores. Table 1 shows a chart that has been produced to facilitate body condition scoring. The use of a chart involves assessing body regions and integrating the information into a BCS (Ferguson et al., 1994).

Body Condition Score	Vertebrae at the middle of the back	Rear view (cross-section) of the hook bones	Side view of the line between the hook and pinbones	Cavity between tailhead and pinbone	
				Rear view	Angled view
1 Severe underconditioning					
2 Frame obvious					
3 Frame and covering well balanced					
4 Frame not as visible as covering					
5 Severe overconditioning					

Figure 1 - Body condition scores and matching cow's appearances (Edmonson, 1989).

Table 1 - Decision chart for body condition score (Ferguson et al., 1994)

	BCS												
Body region	2.00	2.25	2.50	2.75	3.00	3.25	3.50	3.75	4.00	4.25	4.50	4.75	5.00
Thurl	V					U					flat		rounded
Ileal tuberosity	angular				rounded							just visible	not visible
Ischial tuberosity	angular		fat pad palpable	rounded							not visible		
Transverse processes of lumbar vertebrae	>.5 visible	.25 to .50 visible				.10 to .25 visible		only tips visible		tips not visible			
Coccygeal ligament	visible						just visible	not visible					
Sacral ligament	visible							just visible	not visible				

The scoring methods includes looking and handling the cow's hip bone, pin bone, the top of the backbone and ends of the short ribs, which together combine the tail head area. With no muscle tissue covering, the tail head area covering you see or feel is the only a combination of skin and fat deposits. As shown in Figure 2, the tail head area is an ideal location to assess body condition due to its skin and fat cover (Rodenburg, 2000).

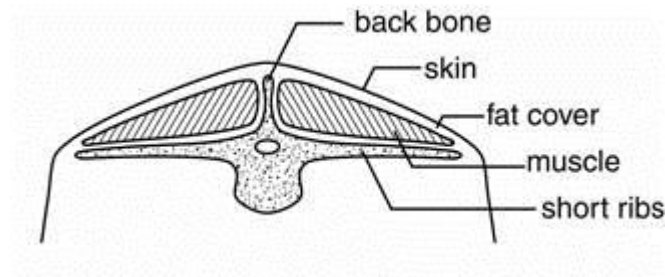


Figure 2 - The tail head area

### **2.1.3. Dairy farming automation**

Information systems and automation have increased in their dominance in agricultural management (Lucey, 2004). The lowering of production costs, improvement in the quality of fresh produce and raised environmental control are some of the advantages of introducing automation to agriculture (Edan et al., 2009). Tools for automation of the milking process were developed, such as the milking stall, test sensing systems, robotic arms and milking equipment (Rossing et al., 1997). Although the required amount of physical labor has been reduced by automated systems in dairy farming, they have also increased the amount of information needed for computerized decisions (Pietersma et al., 1998). In automatic dairy farming, most data is collected through sensors (e.g., weigh scale, near-infrared (NIR), pedometer, sound sensor), but body condition scoring is still performed manually (Bercovich et al., 2013). Several attempts have been made to automate the process of BCS measurement, but so far there has been no change in the BCS practice (Halachmi et al., 2008, Bewley et al., 2008, Ferguson et al., 2006, Bercovich, 2012).

#### **2.1.4. *Body condition scoring automation***

Manual BCS, as performed today, is both time consuming and requires great skill and experience from the working staff (Halachmi et al., 2008). The current method is manual and subjective: the scores depend on the person who performs the measurements, and sometimes a given person might give different scores to the same cow, depending on the previous cows seen (Halachmi et al., 2008). Furthermore, the technique is subjective, expensive (since requires frequent measurements) and has no computerized reports (Azzaro et al., 2011). Therefore, despite the need, only 5% of US dairy farms have adapted the BCS system (Azzaro et al., 2011). An automatic BCS system would be more time effective, more objective, less expensive and enable better animal healthcare (Bewley et al., 2008). Previous research (Bercovich, 2012, Azzaro et al., 2011, Bewley et al., 2008, Halachmi et al., 2008) dealt with automating the BCS process, but so far the system is still manual. Further details on automated BCS attempts using computer vision are detailed in section 2.2.

## **2.2. Computer vision and image processing**

### **2.2.1. *Background***

Imaging technology is commonly employed since it is nondestructive, can be fairly accurate, and yields consistent results (Chen et al., 2002). In computer vision it is easy to get good results with a good connection to reality but it is hard to achieve great results that are completely realistic (Szeliski, 2010). Computer vision and image processing are studies of algorithms and methods for automating the visual perception (Keller & Gader, 1995). This involves tasks such as noise removal, smoothing, changing picture brightness, and sharpening of contrast (image processing - low-level vision); segmentation of images to isolate regions or objects and recognition of the segmented regions (image processing - intermediate-level

vision); and finally interpretation of the scene and achieving non-picture output (computer vision - high-level vision) (Keller & Gader, 1995). Typical problems in computer vision and image processing applications include enhancement, restoration, smoothing, filtering, data compression, detection and identification, shape analysis, region analysis, object motion estimation and tracking, feature extraction and factorization and synthesis (Vajda, 1994).

### **2.2.2. Common methods**

In the field of computer vision and image processing there are many algorithms, methods and topics. Three major steps are edge detection, segmentation and object recognition (Szeliski, 2010). Edge detection is the process of analyzing the data in pixels in order to determine if the pixel is an edge or not. The analysis is usually performed by using the first or second derivative on the pixel values and looking for maximum in the first derivatives or zero in the second derivative (Szeliski, 2010). Segmentation is used to separate object in an image and decide which pixel belong to which object. Segmentation is usually performed by either detecting the edges and using them as border for the segments or by combining neighboring pixels with similar values (Schowengerdt, 2006). Object recognition is the decision if an object in an image is the same object as in prior data and it is performed by comparing the features of these objects and receiving a result with a very small error; the hard part in performing object recognition is deciding what are the right features for my object (Szeliski, 2010).

### **2.2.3. 3D computer vision**

3D computer vision uses color data (Red, Green and Blue data denoted as RGB) and depth data (denoted as D) in order to obtain a better understanding of the environment (Cyganek & Siebert, 2011). RGBD cameras capture depth information per-pixel for a RGB image.

An important 3D computer vision sensor is Kinect initially devised for the gaming world and it is low-cost, fast and reliable (Smisek et al., 2013). The impact of Kinect has extended beyond the gaming world and is now used by researchers and practitioners in computer science, electronic engineering, and robotics (Zhang, 2012). The Kinect uses time-of-flight sensing to obtain the depth data (Henry et al., 2014). Kinect can be used to reconstructing surfaces and more accurately approximate real-world geometry (Izadi et al., 2011).

#### ***2.2.4. Kinect and RGBD cameras applications***

The Kinect camera was originally designed for gaming applications, but was adopted by the academic community and was found to be useful in a wide variety of fields (Zhang, 2012). Kinect applications in the robotic field included navigation, tracking and object detection (El-laithy et al., 2012). The Kinect camera can be used for hand gesture recognition and as an input device for various hardware sets (Ren et al., 2011). In the healthcare field Kinect was used for physical rehabilitation as part of motion based virtual reality systems (Huang, 2011, Fern'ndez-Baena et al., 2012). Several industrial applications of the Kinect were developed over the years in the fields of safety, performance improvement, monitoring and layout planning (Rafibakhsh et al., 2013).

#### ***2.2.5. 3D computer vision common methods***

The goals of 3D computer vision algorithms are similar to the goals of 2D computer vision algorithms. Methods applied in 3D vision include all 2D methods and additional (Smisek et al., 2013). In goals such as object detection and object recognition methods like background subtraction can be used the same as in 2D algorithms, but when integrated with depth analysis the methods become much more reliable and accurate (Jungong Han et al., 2013). The use of detecting and classifying an object based on similar color and 2D location features becomes

much more powerful when adding the depth information to the list of features; this illustrates the important contribution of 3D computer vision (Tanabe et al., 2012). The combination of color and depth information not only opened the field to new methods and techniques, but also made the use of methods from the field of 2D computer vision more accurate and with higher consideration to changes in the three dimensional space (Chen et al., 2000).

#### **2.2.6. *Agriculture computer vision***

The development of computer vision technology had completely penetrated the agriculture field due to decreasing technical costs, the large amount of technical means and the biological diversity (Ji et al., 2009). Computer vision systems have helped improve the welfare and efficiency of livestock (Van der Stuyft et al., 1991).

#### **2.2.7. *Agriculture livestock computer vision***

Livestock farms have turned from small farms with small numbers of several species into large farms with large numbers of a single specie with a single farmer responsible for the control (Frost et al., 1997). The health of the livestock and quality of the products from the livestock farms are almost entirely dependent on the experience and subjective assessments of the farmers. Advanced sensors such as cameras performing as monitoring systems can help achieve experienced and objective measures (Frost et al., 1997). Computer vision is an alternative to human observation and control in livestock farms (Shao & Xin, 2008). Developments in computer vision are appearing in many applications in livestock farming such as determining the weight and size of livestock animals (Tscharke & Banhazi, 2013), monitoring welfare of fishes (Zion, 2012) and detection of skin tumors on chicken carcasses (Kim et al., 2004).

### **2.2.8. *Agriculture computer vision applications***

Typical target applications in agriculture include grading, quality estimation, monitoring of processes, decision support such as when to harvest, and disease detection (Costa et al., 2011, Tillett, 1991, Tillett et al., 1997). Table 2 contains examples from all review articles from 2011 onwards of computer vision applications in livestock (including fish and poultry) agriculture. The main applications for livestock farming are quality control, animal welfare and grading (Tillett et al., 1997).

Table 2 - Examples of livestock agriculture computer vision applications

Application	Field in agriculture	Computer vision methods used	citation
Fish and fish eggs counting	Aquaculture and fish farms	Pixel count, histogram analysis and shape analysis	(Alver et al., 2007)
Size measurement and mass estimation of fish	Aquaculture and fish farms	Stereo vision, segmentation, edge detection and geometry analysis of images	(Torisawa et al., 2011)
Gender identification and quality assessment of fish	Aquaculture and fish farms	Morphometric and color analysis	(Zion et al., 2008)
Species and stock identification	Aquaculture and fish farms	Segmentation, morphometric, sample shape analysis, color analysis, frequency analysis and background illumination	(Aguzzi et al., 2009)
Monitoring welfare of fishes	Aquaculture and fish farms	Edge detection, movement detection and analysis, segmentation and noise redaction	(Zion, 2012)
Meat and meat products quality control	Livestock products	Hyperspectral Imaging, thermography, color analysis, texture analysis, frequency analysis and X-ray spectrum analysis	(Dale et al., 2013, Singh et al., 2013, Vadivambal & Jayas, 2011)
Milk and milk products quality control	Livestock products	Color analysis, surface and volume analysis and texture analysis	(Jeliński et al., 2007)
Aquatic food and food products quality control	Aquaculture products	Size and volume analysis, color analysis and shape analysis	(Xiong et al., 2010)
Detection of meat and bone meal in compound feedstuffs	Livestock feeding	Hyperspectral Imaging	(Fernández Pierna et al., 2004)
Discriminating between fish and terrestrial protein	Meat products analysis	Hyperspectral Imaging	(Riccioli et al., 2011)
Detection of skin tumors on chicken carcasses	Meat products analysis	Hyperspectral Imaging and segmentation	(Kim et al., 2004)

Discriminate different types of cheeses	Livestock products	Hyperspectral Imaging	(Burger & Geladi, 2006)
automated detection of estrus	Livestock farming (cows)	Behavior and motion analysis	(Rutten et al., 2013)
Temperature mapping of beef	Livestock products	Thermography	(Vadivambal & Jayas, 2011)
Determine weight and size of livestock	Livestock farming (cattle, buffalo, chickens, pigs, sheep, fish, horses and rabbits)	Segmentation, object recognition,	(Tscharke & Banhazi, 2013)

### **2.2.9. Kinect and RGBD cameras applications in agriculture**

The agriculture field holds many challenges and the use of a 3D camera such as Kinect can be a big help and a step forward for the field (Yoshida & Kawasue, 2014). The Kinect camera has been used for many applications despite the fact that it is a relatively new technology. Examples include detection of stems of potted tomato (Fu et al., 2014), characterization of vegetation structure (Azzari et al., 2013), 3D digitisation of plant leaves (Kempthorne et al., 2014), extracting corn geometric structural parameters (Chen et al., 2012) and other plant related uses. Kinect has also been used in characterization of soil (Marinello et al., 2013) and terrain analysis (Bellone et al., 2013). In livestock agriculture the Kinect was used for measurement of back posture in dairy cows (Viazzi et al., 2014) and for more accurate milking (Akhoulfi, 2013)

### **2.2.10. BCS using Computer vision**

Computer vision has been used to predict BSC in previous research (Table 3). Bewley et al. based his algorithm on identifying 23 anatomical points on the cow's body contour representing her shape, the points were found manually from automatically acquired photos. The points were transformed to a Cartesian system, 15 angles were extracted, and two types of BCS models were calculated. The models showed good results (99.87% of predicted BCS were in a distance of up to 0.5 Points in the BCS) however, they lacked full automation (Bewley et al., 2008). Halachmi et al. tested the correlation between the BCS and the mean absolute error (MAE) for fitting a polynomial curve with the cow contour (Halachmi et al., 2008). Azzaro et al. used 23 anatomical points from Bewley et al.'s model (Azzaro et al., 2011), their size and angle were adjusted and they were scaled to fit in a unit square as shown in Figure 3. Azzaro et al. tested models from both Bewley et al. and Halachmi et al. The results favored Bewley et

al.'s model (Azzaro et al., 2011). Bercovich et al. (Bercovich et al., 2013) used aspects from all previous studies and developed an image processing algorithm that uses color transformation, noise removal and segmentation to extract the cows' contour and calculate angles and frequencies in the contour for BCS prediction. The algorithm achieved a mean absolute error of 0.34 (Bercovich, 2012). Table 3 shows four state-of-the-art research on automatic BCS and BCS from images.

Table 3 – State-of-the-art research on automatic BCS and BCS from images

Title	Study highlights	Citation
Potential for estimation of body condition scores in dairy cattle from digital images	Research included the automated acquisition of images and the manual extraction of 23 anatomical points. From the anatomical points 15 angles were calculated and two linear mixed models were constructed. The two models were evaluated for BCS extraction.	(Bewley et al., 2008)
Cow body shape and automation of scoring BCS	Thermal images were taken from above and were analyzed. The research hypothesis is that for any contour, it is possible fit a polynomial curve and calculates the mean absolute error between them. From the error it will be possible to calculate the BCS with the following model: $BCS = 5x - 9x^2 (1/MAE)$ . The process of finding the best fitted curve for a thermal image and the calculating of the BCS will be done automatically.	(Halachmi et al., 2008, Halachmi et al., 2013)
Objective estimation of body condition score by modeling	The research created a system for evaluation of BCS and was based on unified coordinate system. Bewley et al., (2008) 23 anatomical points were used in order align and rotate and to fit into a unit square. The variability from the average shape was used in order to determine the BCS. The images used in the research were from a low cost digital camera.	(Azzaro et al., 2011)
Automatic cow's body condition scoring	Using images captured by a DSLR camera over a red floor at the exit from a milking parlor the research found the contour of the cows and calculated their Fourier descriptors. The Fourier descriptors were used in a linear regression model in order to evaluate the BCS.	(Bercovich, 2012, Bercovich et al., 2013)

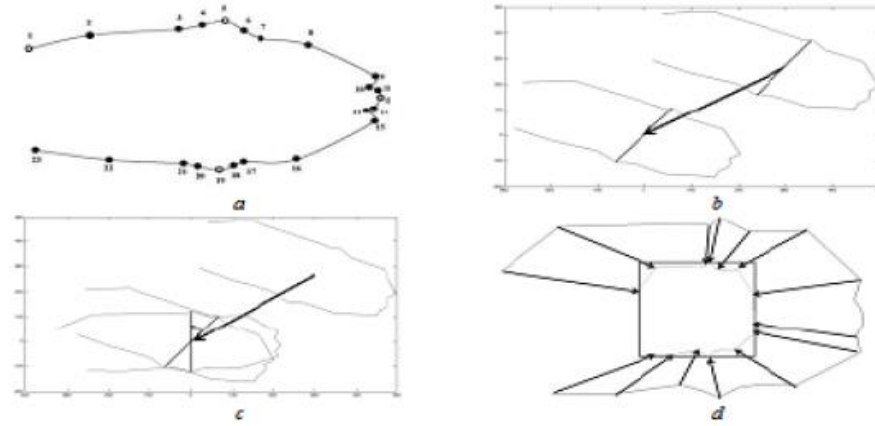


Figure 3 - Algorithm steps for computer vision BCS (Azzaro et al., 2009)

This thesis study aims to build upon these research and their results using a low cost 3D digital camera instead of 2D camera and thermal camera. Additionally, major effort was focused on automating the whole process – automated image selection and object extraction (independent of different backgrounds).

## 2.3. Statistical analysis

### 2.3.1. Regression

The statistical process of estimating the relationship between variables; the relationship can be linear, polynomial, binary or a deviation to classes (Cohen et al., 2013). The relationship is described using a formula that shows what mathematical operations are needed in order to get from the independent variables to the dependent variable (Cohen et al., 2013). As with any statistical analysis the estimating of the formula is performed from past data (Cohen et al., 2013). There are many ways of performing regression and detecting the relationship formula, among the most common ones are least squares method and conjugate gradient method; both methods aim to find the formula that will minimize the error from the real data (Long & Freese, 2006).

### ***2.3.2. Regression in agriculture***

Regression can be found in almost every field in agriculture as an evaluation tool or as a prediction tool (Chatterjee & Hadi, 2013) . Several examples for regression in agriculture are: modeling crop yield (Imran et al., 2013), assessing the effect of soil tillage on crop growth (Van den Putte et al., 2010) and estimation of digestible and metabolizable energy of wheat for pigs (Bolarinwa & Adeola, 2012). Regression has also been used to estimate body condition score for dairy cows (Bercovich et al., 2013).

## 3. Methods

### 3.1. Overview

The research steps (Figure 4) included data acquisition, image processing, feature selection and extraction, prediction model, performance evaluation and validation and are detailed in the followings sections.



Figure 4 - Research main steps

### 3.2. Data acquisition

#### 3.2.1. Database

The data in this research are RGBD images (detailed below) and additional cow data. The images were taken at the ARO research farm at Bet Dagan between October 2013 and November 2013 (30.10.2013 – 03.11.2013) by automatically sampling cows each night for four consecutive days using an experimental system especially constructed (Figures 5-6). A Microsoft Kinect camera (version 1) was placed at the exit from the milking parlor at 2.5 meter height and images were captured automatically using a passive infrared motion detector which served as the camera trigger. The trigger was placed 0.3 meters before the camera after the cow exited the milking parlor (Figures 5-6).

The camera and the trigger were connected to a PC and operated using OpenNI software. Each time the camera was triggered, a 4 seconds movie clip was captured containing between 100 and 120 frames.

All movies were saved, dated and time coded (Appendix E). The movies were automatically split into singulated 480X640 images using Matlab: each image contained 4 matrices - one corresponding to Red, Green and Blue and one for depth (the distance between the sensor and the floor is 2.5 meters and the distance to the cow depends on the cows height). Additional cow data included the cows' age, as was stored on the farm's records, the cows' weight, measured at the same time as the image collection, and the cows' body condition score, evaluated manually by a trained expert at the first day of data collection. The cows' manually extracted BCS was used as the ground truth for this research.

Raw data included 2647 movies captured for 100 cows. The cows in this research were chosen so that for each BCS group there will be a sufficient amount of cows. Table 4 shows the number of cows in each BCS group. 14474 frames were extracted from 389 movies that remained after initial screening. The screening included deleting images of cows not included in the research and empty movies or movies without the backend of the cow. This resulted in 14474 images for the database analysis (Appendix E). The 389 movies were of 100 cows, and it should be noted that same cows appeared in more than one movie.

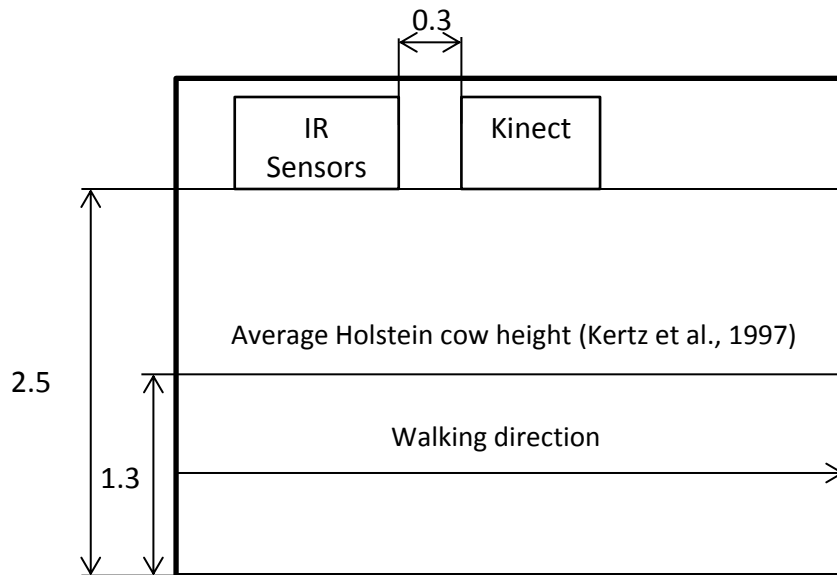


Figure 5 - Schematic layout of the system (in meters)



Figure 6 - Photograph of the experimental setup

Table 4 - Number of cows in each BCS group

BCS	Less than or equal to two	Between two and two and a half	Between two and a half and three	Between three and three and a half	Between three and a half and four	Larger than four
Number of cows	12	33	21	9	11	14
Percent from total	12%	33%	21%	9%	11%	14%

### 3.2.2. Training and testing data

Data was trained using two types of data, image data and movie data, calculated from all images in a movie. Each movie was composed out of 100 images, the image used as data were selected in the image selection part of the image processing algorithm were used in the image data set.

When training on image data, results were predicted for each frame. The final result for the movie was calculated using the following measures: mean, median and selection of largest group of frames.

When training on movie data the features for each movie were calculated and then predicted separately for each movie.

Data was randomly split three times, into training and testing sets with each set consisting of different cows. Between the three random splits, the training set and the testing set that were chosen had cows from all groups of BCS and similar distributions (Figure 7).

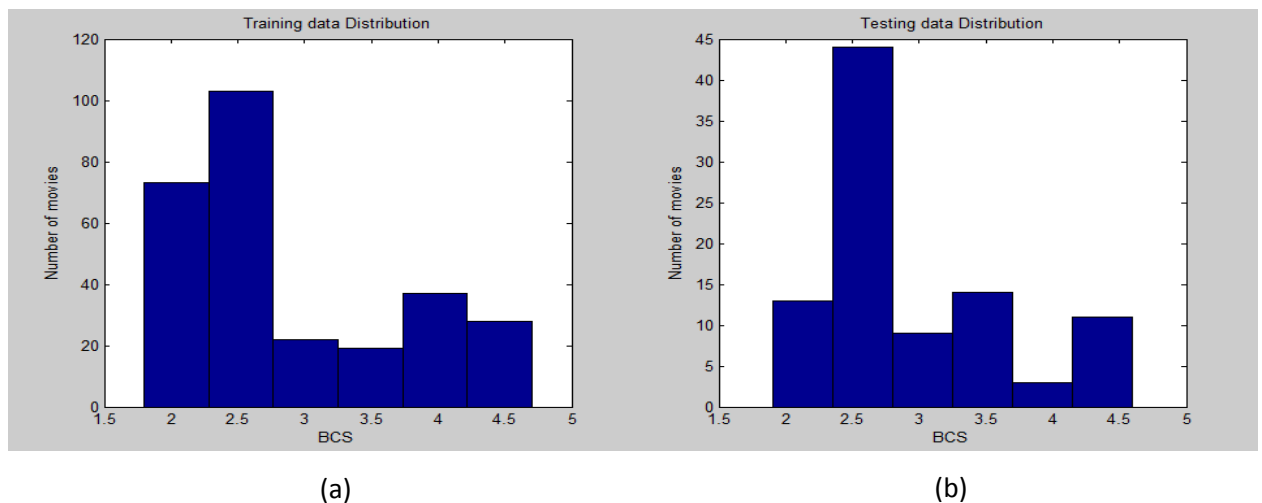


Figure 7 – Training (a) and testing (b) data distributions

### **3.3. Image processing**

#### **3.3.1. Overview**

The image processing procedures aim to produce a data set of normalized images that include the back end of the cow with equal size and rotation. Images without the data needed (images without the backend of the cow) or with obstructions (e.g., humans passing, other cows hiding part of the cow) were filtered out automatically during the image processing algorithm.

Image processing included the following steps: restoration, object recognition and separation, image rotation and image cropping and normalization. The principles of the algorithms are presented below followed by detailed elaboration in section 4.1.

#### **3.3.2. Image restoration**

The depth image includes noise in the form of empty groups of pixels caused by several sources: movement in the environment, infrared light from other sources (moon, farm lamps and electronic devices) and blank spots due to lack of view. The empty groups of pixels and blank spots are filled using interpolation of the data from the data in neighboring pixels and other frames in the same movie (Figure 8 includes two images, the left one with blank, dark blue spots noise and the right one after the restoration). Several types of filters and convolutions (detailed in object recognition and separation, section 4.1) are applied to the images and to the contour of the selected object in order to reduce added noise found in every natural image; further information on this step can be found in the algorithm chapter (Appendix B pseudo code).

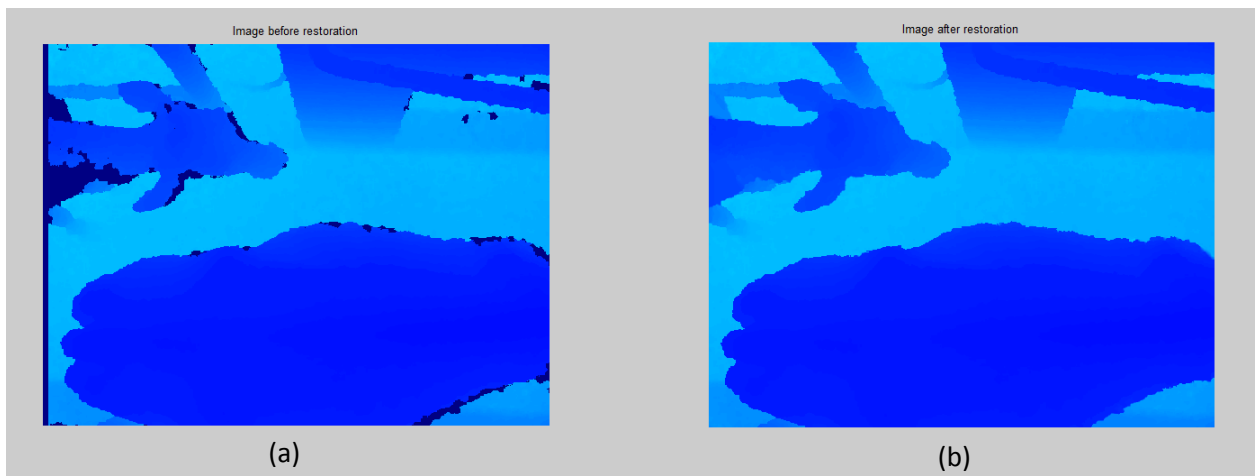


Figure 8 - Image before (a) and after (b) restoration

### **3.3.3. Object recognition and separation**

The input images to the object recognition step are depth images with zero to two cows in them including the farm background (the floor, the bars containing the movement of the cows and anything else that might appear e.g., farmers, cows from other parts of the farm inserting their head into the image from behind the bars). This step aims to identify the cow going out from the milking parlor with her rear end in the image and a sufficient part of her body in the image and to separate it from all other objects in the image. A series of image processing techniques are applied on the image: thresholding the background and transforming to a black and white image, subtracting the background to eliminate objects from the background, edge detection to separate two sub sequential cows followed by a series of tests on the objects size, height, width, length and position in order to make sure that the right object is selected (detailed in algorithm chapter). The algorithm flowchart is described in chapter X (Figure 12). The output of this step is an image containing only depth values of the cow and zeroes in all other pixels (Figure 9 and Figure 10 shows two samples of initial images and the objects extracted from them, Appendix B pseudo code).

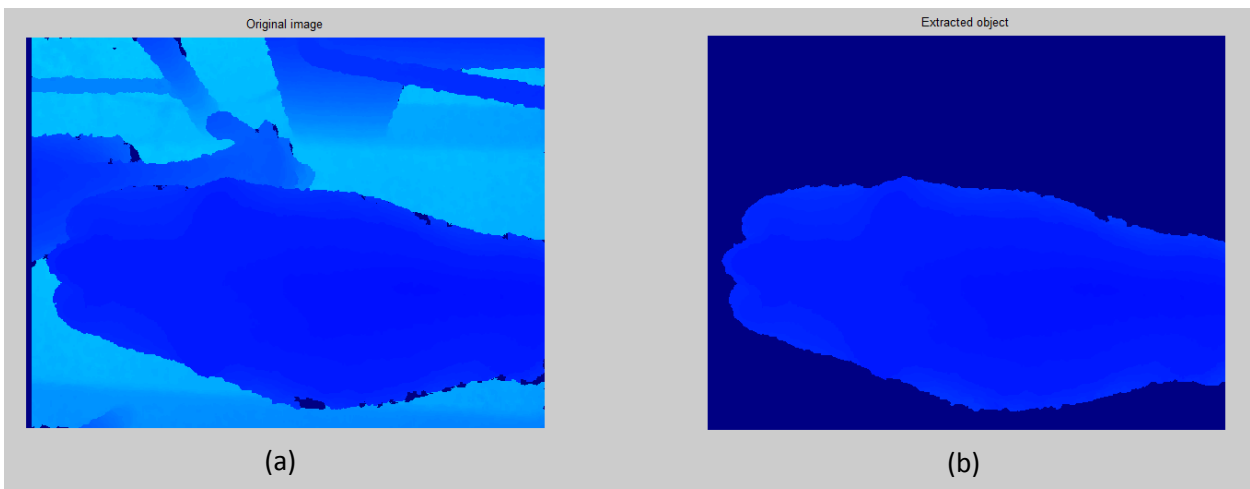


Figure 9 - Example 1 of initial image (a) and extracted object (b)

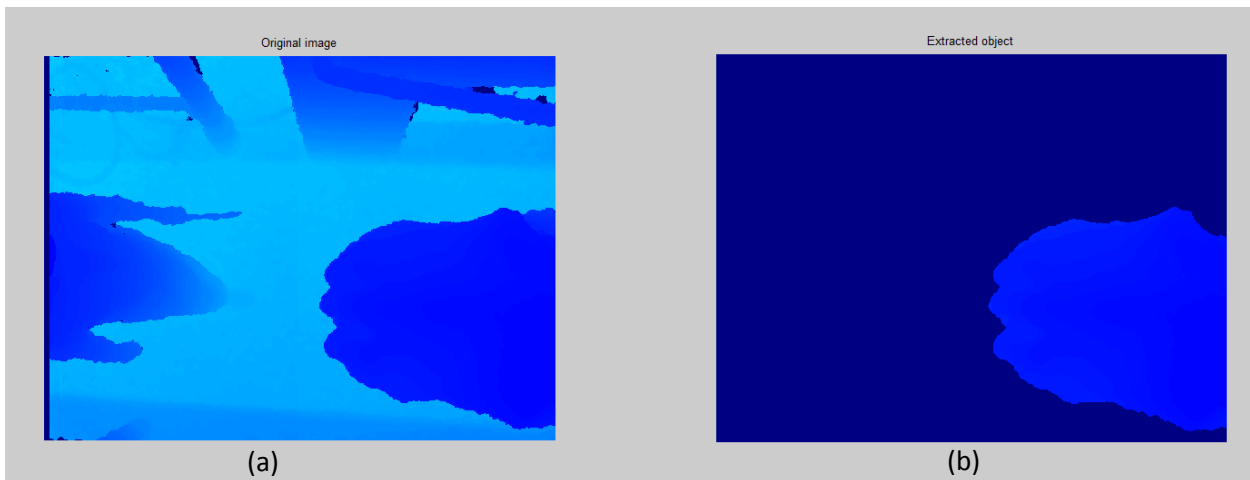


Figure 10 – Example 2 of initial image (a) and extracted object (b)

### 3.3.4. Image rotation

In order to make sure that all images have identical angles between the cow's backbone and the X axis of the image, two methods were developed and automatically compared. The first method takes the highest points in the cow's spine, calculates the angle of the line created from the two highest points and rotates the image at that angle. The second method uses linear regression on all the highest points in the cow to determine the position of her spine. After applying regression, the image is rotated according to the regression coefficient which

represents the slope. To automatically determine which method works better for each image, the symmetry of the images were compared; the symmetry was calculated as the sum of differences between the lower and upper parts of the image. To ensure the image is correctly rotated for every method a range of 6 degrees around the result angle with an accuracy of 0.1 is checked for better symmetry, the image is rotated at an angle of 0.1 at each step and the symmetry (sum of differences between the lower and upper parts) is calculated by for each step (Appendix B pseudo code).

### ***3.3.5. Image cropping and normalization***

Cropping and normalization were conducted to ensure that all images are identical in size with similar value ranges. The image is cropped by deleting all pixel rows and/or pixel columns that do not contain any part of the cow (parts identified in the Object recognition and separation step above), and the cow's tail (identified as a part of the cow with less than 50 pixels width and everything left to it) is cut out from the image. The image size is adjusted to be 150X200. If the image is larger than the goal size (150X200), the image will be cropped again making sure that the back end of the cow is at the left of the image and the cows spine is at the middle; if the image is smaller than the goal size (150X200) then the image will be framed with zeroes ensuring that the cow's end is at the left of the image and the cow's spine is at the middle. At the end of the cropping stage, all pixel values are normalized to values between 0 and 1, see for example, Figure 11 which includes two images, the original image and the cropped normalized image (Appendix B pseudo code).

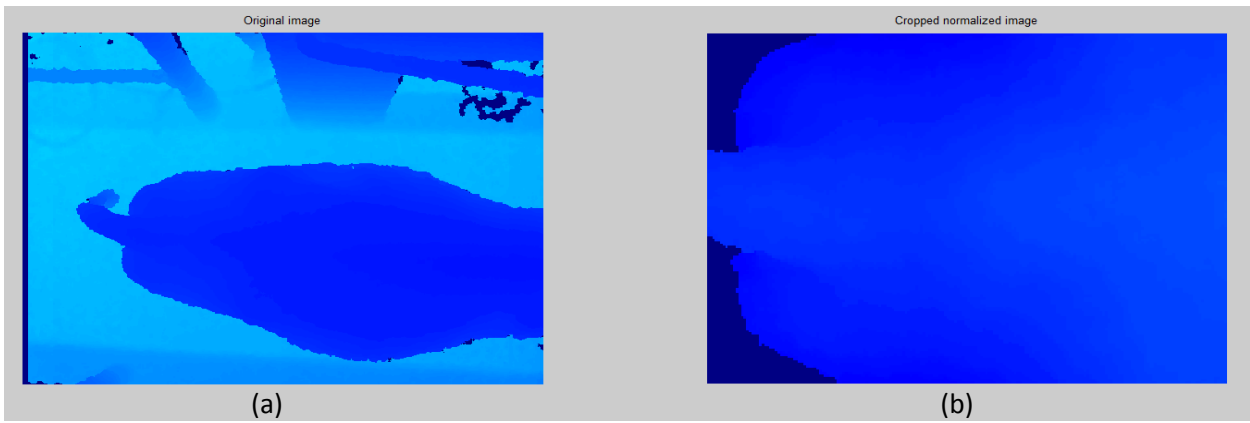


Figure 11 – original image (a) cropped normalized image (b)

### 3.4. Feature selection and extraction

Three types of features are used in this research: features based on pixel data, features calculated from image data and features from prior data.

Features based on pixel data are the means of small groups of pixel values, selected using correlation of the pixel (relative height) to BCS; the groups were created by combining close pixels (neighbors) to one feature by calculating their value's mean. The pixel with the largest correlation was detected and the nine neighboring pixels (also neighbors of selected neighbors) with the biggest correlation, of pixel value to BCS, were added to the first group; then again the pixel with largest correlation was detected in the image, without considering the group already found and pixels with in distance of up to ten pixels away from the first group, and its neighbors were found. The process was repeated six times (the number of maximum and minimum points in the correlation image).

Features calculated from the image data are the set of features calculated from the output of the image processing algorithm:

- The number of local minimums in an image which correlates to the number of bulges in the cow's back.

- The number of local maximums in an image which correlates to the number of bulges in the cow's back.
- The mean of all pixel values (the mean value of the relative heights) which correlates to the degree of change from the highest pixel.
- The standard deviation of the all pixel values which correlates to the degree of change from the mean value.
- The average distance between the contour and the middle of the right column this correlates to the roundness of the contour.

Features from prior data include the cow's age and the last recorded weight of the cow.

Feature selection is based on their contribution to the prediction model (based on correlation and the model's measures with and without the feature) and is selected separately for each model. Multiplication of features and the ratio between height and weight are included as additional features for part of the prediction models. All features are normalized to be between 0 and 1 in order to receive value in the same scale for all cows and features.

Details of all features and their correlations are noted in section 5.2.

### **3.5. Prediction models**

Prediction and classification of the BCS was calculated using several methods based on a-priori analysis of several models detailed in Appendix A. The best models were based on polynomial regression.

Prediction models were trained separately on image and movie data and tested only on movie data. For image data, a method based on K-means, separated the results into groups and the largest group was selected for the prediction of the movie result (this is denoted as '*Group Selection*'), the results for this method were compared to the results without the use of '*group*

*selection'*. The expected result for '*group selection*' is that either the noise will go down and the result will be more accurate or that the '*group selection*' will screen the accurate results and the error will increase.

The different models were compared to results of previous studies. The model that achieved the best results for the testing set was selected for the system implementation.

### **3.6. Performance evaluation and validation**

Evaluation of the regression models is based on six main measures: the mean and median absolute errors, the percent of correct classifications (for 6 and 4 classes), the percent of classification that are less than or exactly one class away from the target class (for 6 classes) and the  $R^2$  calculated between the results and the target values. The classification models were evaluated based on classification into six and four classes (6 classes: lower than 2 BCS, between 2 and 2.5, 2.5 to 3, 3 to 3.5, 3.5 to 4 and higher than 4 BCS; 4 classes: lower than 2 BCS, between 2 and 3, 3 to 4 and higher than 4 BCS).

Same classification models were also trained using these classes.

Additional analyses of the results included correlation of the results and the target values, confusion matrices, repeatability analysis which checked several movies from the same cow for similar results (the number is different for each cow and is determined by the number of movies in the database) and error analysis that checked the error sizes and the percentage of all errors.

#### **3.6.1. Sensitivity analyses**

The following sensitivity analyses were performed

- **K-fold cross validation** of the training sets (with K values of 10, 20 and 30).

- **Size of the training set** – 10 random selections of part of the training set is performed for each of the following training set sizes (90%, 80%, 70%, 60%, 50%, 40%, 30%, 20%, 10%, 5% and 3%).
- **Type/distribution of training data** - to ensure that no movies with false data will be considered as cows, 100 of movies with distractions (no cow's backend in them) were inserted into the database (dairy farmer passing through, more than one cow in the movie, cows touching, movie with no cows and movie with cow's head looking from behind the fence).
- **Noise** - randomly added Gaussian noise was added in to two parts of the system, raw data movies and the collected data before entering the prediction models. After inserting the noise into 10 movies, the number of movies which gives a correct output (images focused on the backend of the cow with 150X200 size and the backbone parallel to the X-axis of the image) is counted and compared (200:1, 100:1, 50:1 and 20:1 ratios of signal to noise).

## 4. Algorithms

### 4.1. Image processing

#### 4.1.1. Work process

Figure 12 shows the flow of image processing algorithms developed in the research.

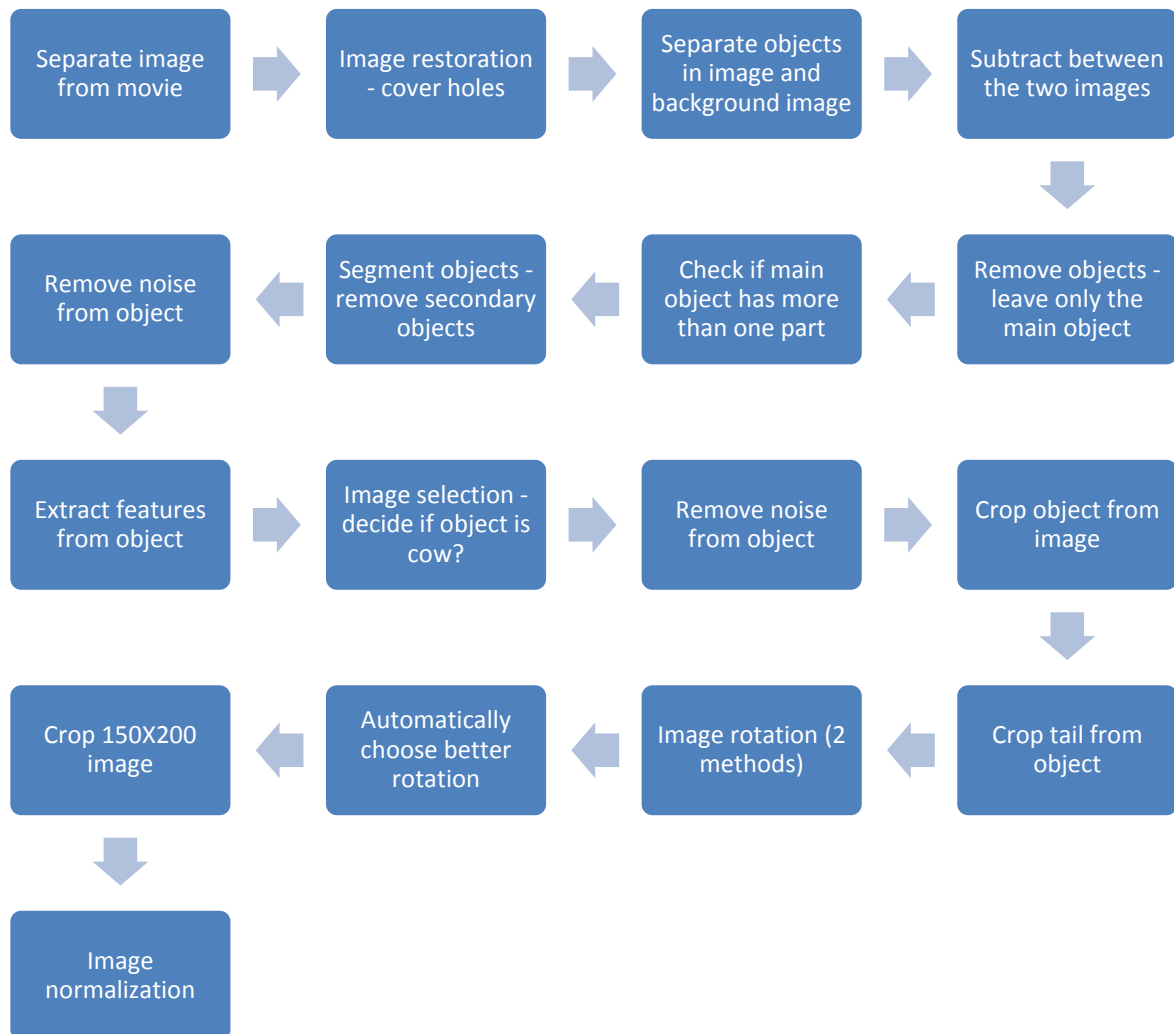


Figure 12 - Work flow of image processing algorithms

#### **4.1.2. Image restoration**

A common method of linear interpolation based on neighboring pixels and a neighbor image were used in order to repair small holes in image. The process has two steps, first for all pixels, if the left neighbor pixel in the earlier frame had a value, the pixel will receive that value. The second step gives still missing pixels the value of the closest neighbor pixel (with a preference to the right pixel – more likely to be the cow due to the direction of movement), with a value, less the difference between its value and its neighbor's value (in the same direction). Figure 13 shows an example of the holes in the image; holes are surrounded with red circles. Part of the object recognition and separation is to apply filters to the image and the object contour, which helps reduce noise in the image (Appendix B pseudo code).



Figure 13 - Examples of holes in the image

#### **4.1.3. Object recognition and separation**

Two thresholds were used to separate the main object from the background:

1. A preset threshold designed to eliminate pixels too far from the camera to be the main object. The preset value was determined empirically to be 1800, the value that will be farther away from the camera than all the cows' backs and as high as possible from the floor.

2. Automatically calculated for each image separately using Otsu's method (Otsu, 1975)

in order to select the main object in the image.

Every pixel left in the image received the value of 1 resulting in a black and white image. After using thresholds, on both background image and main image, a subtraction of the two was made to consider only the objects that did not exist in the background image. Separation of objects was made using an 8 neighbor filter and the size of each object was measured by counting the number of pixels in it. The largest object was selected and all other objects were deleted from the image. Two convolutions of the objects values and masks of  $[1;-1]$  and  $[1 -1]$  were used in order to detect the border of two cows touching each other and then united into one object. For each pixel we tested if the derivative value, performed by the convolution, was in the range of the thresholds selected (larger than 90 and smaller than 500) then the pixel was considered a border between two cows and its value was set to zero. Again the image was divided into objects using an 8 neighbors rule and the largest object was selected. Table 5(a) shows a filter used to detect the contour of the object; after the filter was used on the image each pixel that was under 1.8 and above 1. In Table 5(b) we can see an additional filter used to smooth the contour.

A second phase of object recognition and separation is performed after image selection; the filter in Table 5(c) was used to cover single pixel holes in the contour by detecting pixel values of less than 0.5 and setting their values to the value of a dynamic 3X3 averaging filter, considering the number of pixels determined as part of the object (Appendix B pseudo code).

Table 5 - Filters used for the object extraction

0.1	0.1	0.1
0.1	1	0.1
0.1	0.1	0.1

(a)

0	1	0
1	10	1
0	1	0

(b)

0.1	0.1	0.1
0.1	0.2	0.1
0.1	0.1	0.1

(c)

#### 4.1.4. Image selection

The following features, all derived empirically, were applied to ensure the image is of a single cow (Appendix B pseudo code):

1. The width of the object, calculated as the largest subtraction between maximum row index and minimum row index for each column – should be smaller than 320.
2. The size of the object, calculated as the number of pixels – should be greater than 60000.
- 3-4. Binary values representing the presence of the object in the first and last column of the image; the features are used to make sure that the cow's backend is in the image – the first column's binary value should be 0 and the last column's should be 1 .
5. The maximum height of the object was used to make sure that the object is a cow and not a fence or a dairy farmer - the minimum distance from the camera should be larger than 1100.
6. The width object's end (at the first column of the image) was compared to the width of object's middle (at the middle column of the object image – the number of row divided into two) to ensure that the object shape is similar to a cow's shape.

#### **4.1.5. Image cropping**

All parts of the image with no object in them are cropped. Image columns and rows were summed each separately and those with a sum of 0 are removed from the image. In order to cut the cow's tail from the image the indexes of the object's edges in every column are subtracted to calculate the width of the object in every column. The image is cut from the highest indexed column with a width of less than 50. After image rotation the image is cropped again; at first the left 200 columns are selected and all others removed; if the image is less than 200 columns the extra empty columns are attached to the right of the image. Images with less than 150 rows extra empty rows are attached to both sides of the image and for images with more than 150 rows the mean center of the columns was detected and the image cropped to have 150 rows around the center or as close as possible to the center (Appendix B pseudo code).

#### **4.1.6. Image rotation**

Two methods of detecting the angles of cows in the image were performed, for each image automatically, the image was rotated according to both angles and the rotated image with the highest symmetry was selected automatically. The first method, determines the angle of rotation, using arctangent on the division of calculated difference of row index for the maximum height (the spine) in the front and rear of the cow and the image width. The second method uses the angle between the x axis and the linear regression line, of highest points in every column representing the spine of the cow. After each method a range of 3 degrees to every direction is tested to correct minor mistakes, the test is performed by rotating the image in an angle of 0.1 until reaching a range of 3 degrees and every time calculating the symmetry.

The symmetry of the image was calculated by subtracting the top part of the image with the bottom part and summing the difference. The symmetry was calculated on the 150 by 200 pixels image. Figure 14 shows an example of an image after this step in the algorithm (Appendix B pseudo code).

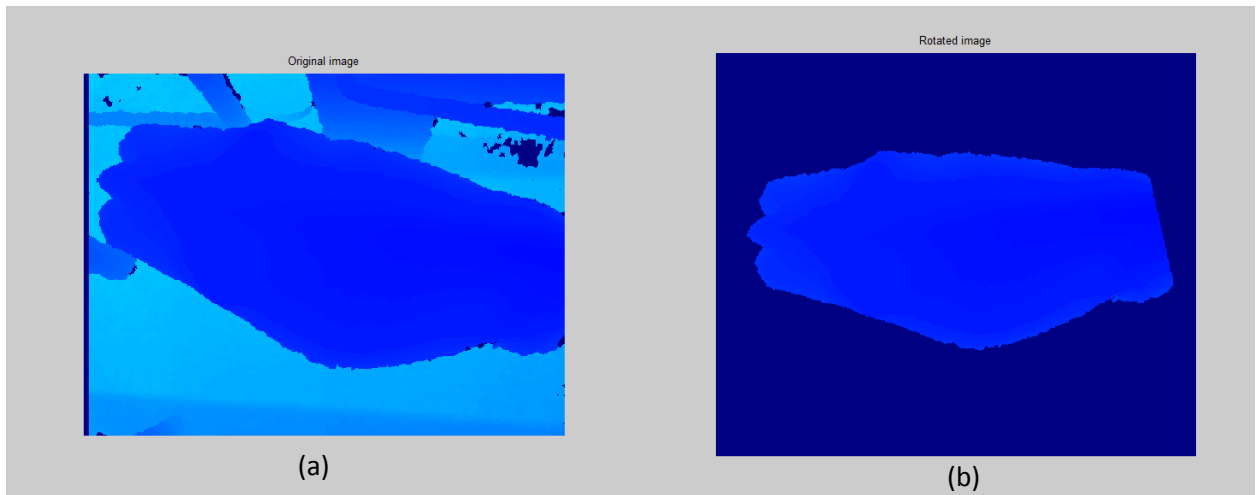


Figure 14 - Original image (a) and rotated object (b)

#### **4.1.7. Image normalization**

The minimum value of the image, set by the threshold at the beginning of the algorithm, was subtracted from all image values; the values were then divided by the maximum value of the image. All pixel values in the image are between 0 and 1 where the maximum value is 1 and the minimum value is 0 (Appendix B pseudo code).

## 4.2. Feature extraction

### 4.2.1. *Correlated points*

The 150X200 images extracted from the movie with a focus on the backend of the cow were reshaped to be a vector with 30000 values. The correlation of each point in the vector and the BCS was tested and 6 zones in the image were detected to have higher correlation. From each zone 10 pixels with the highest correlation were chosen and their mean was calculated. The 6 means of the pixel values found using the correlated zones are used as features in the prediction algorithm and their positions were saved in order to automatically extract their values from new incoming images (the test data for instance).

### 4.2.2. *Image features*

Several features were used in the prediction algorithm. The average relative height, computed as the image averaged value, shows a dominance of higher or lower values that can suggest a convex or concave 3D curvature. The relative height standard deviation, computed as the standard deviation of all image values, was also be linked to the 3D curvature of the cow. The maximum height of the cows' backend, the same value that was used to normalize the output image of the image processing algorithm, a direct connection to the original image value scale which is suggestive to the size of the cow and her BCS. The number of local minima and the number of local maxima of the relative heights image are both important features, who help the prediction algorithm consider the level of torsion in the cow's back, a large part of the manual BCS assessment. The average of the Euclidean distances to the cows outline from the middle of the far right column was used to learn about the 2D curvature. All features were

automatically extracted from each selected frame using the image processing algorithms described above.

#### ***4.2.3. Non-image data***

Two additional important features were used: the first was the last recorded weight of the cow, as was measured by a walking weight scale in a time period of 1 to 5 days before the Kinect data was collected. The second feature is the age of the cow in months as was stored in the farm computer.

## **5. Results and Discussion**

### **5.1. Automated image selection**

The database included 422 movies all with at least one cow's backend (the rear part of the cow's body) and most with the aforementioned disruptions. The 422 movies included 100 different cows. The image processing algorithms were tested on all movies (training and testing sets) from the database. Additionally, the 100 movies from the same dairy farm, with no cows in them were also analyzed. Analyses indicated that 100% of the image outputs (the end of the image processing phase) were correct (0% false positive), including just the cow's backend in the right centering and rotation. Out of the 422 movies, 389 movies were selected as movies that contained the backend of a cow (92.18%). All 33 movies with no images in them selected were checked visually and all of them indeed had interferences (such as more than one cow) indicating the success of the image processing algorithms in incorrect frame selection.

### **5.2. Correlation analysis**

All model variables are listed in Table 6 and the linear correlation between them and the BCS is shown in Table 7. Shown in Figure 15 is a colored map of the correlation between anatomical points in the cows and the BCS, the left side of the image is the backend of the cow and the right side of the image is slightly above the cow's waist. It can be seen that the images automatically removed from the movies were set to include at least 80% of the target zone (not all images include the far right side of the image). Circled and named in Figure 15 are variables used in the model. All correlations were calculated on training data. Variables 1 to 6 were chosen as a mean of the 10 pixel values for each variable, the six zones (10 pixels areas)

were the 6 clearest maximum and minimum points at the correlation map between pixel value and BCS (Figure 15).

Table 6 – List of variables of the prediction models

Variable	Meaning
Y1	Cow BCS
X1	The relative height of the right edge of the cow's backend
X2	The relative height of the cow's spine
X3	The relative height of front left side of the cow's backend
X4	The relative height of front right side of the cow's backend
X5	The relative height of back left side of the cow's backend (next to the tail)
X6	The relative height of back right side of the cow's backend (next to the tail)
X7	The average relative height of the cow's backend
X8	The standard deviation of relative height of the cow's backend
X9	The height of the cow
X10	The number of minima height points on the cow's backend
X11	The number of maxima height points on the cow's backend
X12	The cow's weight (form last performed weighing)
X13	The cow's age
X14	The average distance from the cows contour to center of cows right edge

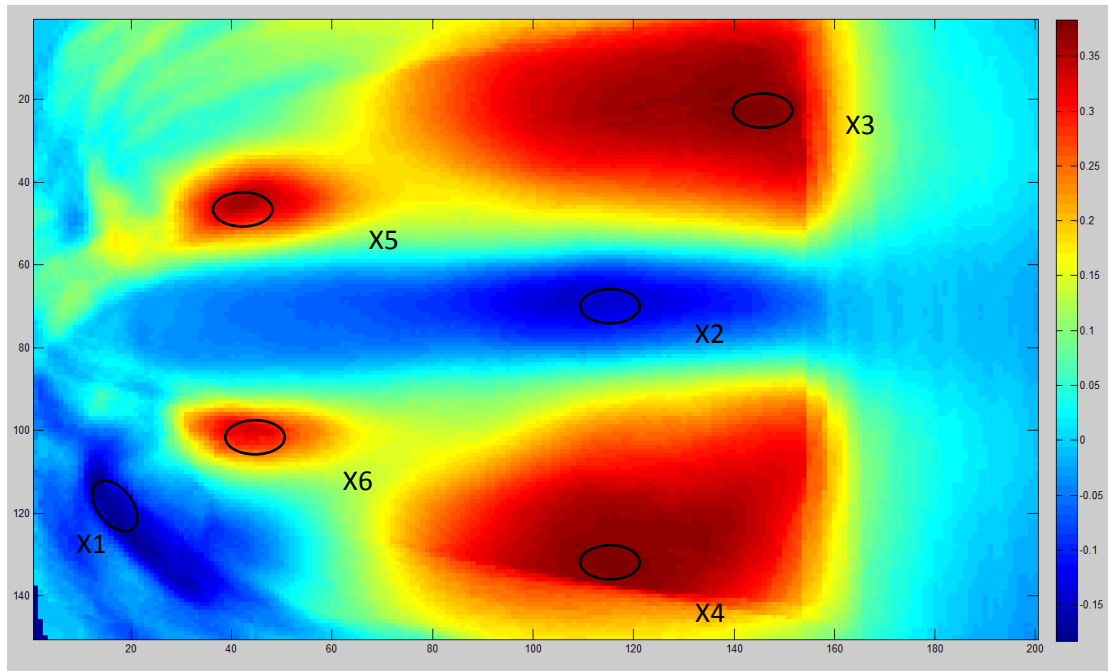


Figure 15 - Correlation between relative heights and BCS

Table 7 -Correlation of model variables

	Y1	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	X11	X12	X13	X14
Y1	1	0.13	0.11	0.38	0.46	0.46	0.46	0.33	0.18	0.02	0.15	0.09	0.32	0.09	0.06
X1		1	0.17	0.32	0.16	0.01	0	0.07	0.06	0.14	0.16	0.12	0.07	0.04	0.39
X2			1	0.22	0.27	0.33	0.33	0.59	0.28	0.15	0.15	0.06	0.22	0.22	0.08
X3				1	0.33	0.49	0.43	0.56	0.35	0.24	0.06	0.14	0.22	0.08	0.12
X4					1	0.5	0.49	0.7	0.47	0.25	0.11	0.16	0.18	0.12	-0.2
X5						1	0.74	0.71	0.57	0.06	0.02	0.15	0.08	0.26	0.16
X6							1	0.67	0.44	0.06	0.06	0.14	0.11	0.21	0.06
X7								1	0.78	0.22	0.09	0.17	0.1	-0.1	0.22
X8									1	0.49	0.15	0.19	0.16	0.05	0.36
X9										1	0.12	0.09	0.12	0.14	0.36
X10											1	0.19	0.07	0.03	0.19
X11												1	0.16	0.06	0.03
X12													1	0.44	0.09
X13														1	0.17
X14															1

As can be seen in Table 7 none of the variables have high correlation to the BCS and the cross correlation between variables is not higher than 0.8. However, every variable adds new information and supports the prediction model.

### 5.3. Model selection

Polynomial regression models were selected after initial comparison of different models that included linear regression, polynomial regression, regression trees, classification trees and artificial neural networks detailed in Appendix A.

Four models of polynomial regression with different features were selected for in depth evaluation. The models are detailed in Table 8.

Table 8 - Model details

Model number	Model details
Model 1	polynomial regression with all variables up to a second cubic equation on image data
Model 2	polynomial regression with all variables except X5 up to a second cubic equation on image data
Model 3	polynomial regression with all variables except X6 up to a second cubic equation on image data
Model 4	stepwise polynomial regression with all variables up to a second cubic equation on image data

Performance measures include the mean absolute error of movie prediction (MAE), the median absolute error of movie prediction (MDAE), the  $R^2$  for movie prediction (R2), the accuracy percentage for 6 classes for movie data (A6), the percentage of up to 1 class miss for 6 classes for movie data (A6 +1), the accuracy percentage for 4 classes for movie data (A4). Table 9 shows the results of the same models with a K-means selection of images (*'group selection'*) to include in the prediction of movie result.

## 5.4. Models results

Tables 9 and 10 show the training and testing data main results for ten models. The best results achieved were MAE of 0.256 for model 2 with '*group selection*', MDAE of 0.1608 for model 1 with '*group selection*', R2 of 0.7575 for model 3, A6 of 0.6526 for model 2, A6+1 of 0.9895 for model 3 and A4 of 0.8105 for model 3. It can be seen that model 3 is the most dominant. Model 2 also shows good results with very high performance in one of the main measures – MAE. For more analysis on models results see section 5.11.

Table 9 - Models results testing (training) data

	MAE	MDAE	R2	A6	A6 +1	A4
Model 1	0.2602 (0.3302)	0.1921 (0.2457)	0.753 (0.7866)	0.6316 (0.5306)	0.9474 (0.915)	0.7684 (0.6633)
Model 2	0.259 (0.3684)	0.1819 (0.2614)	0.7404 (0.7386)	0.6526 (0.4558)	0.9474 (0.8707)	0.7789 (0.5714)
Model 3	0.2809 (0.3432)	0.2294 (0.2599)	0.7575 (0.7713)	0.4737 (0.5136)	0.9895 (0.9014)	0.8105 (0.6497)
Model 4	0.2647 (0.326)	0.1817 (0.2384)	0.7464 (0.7903)	0.5895 (0.534)	0.9474 (0.9184)	0.7789 (0.6599)

Table 10 - Models results testing (training) data with '*group selection*'

	MAE	MDAE	R2	A6	A6 +1	A4
Model 1	0.2621 (0.3302)	0.1608 (0.2304)	0.7427 (0.7717)	0.5895 (0.5544)	0.9263 (0.9082)	0.7684 (0.6769)
Model 2	0.256 (0.3746)	0.1752 (0.2629)	0.7211 (0.723)	0.6316 (0.483)	0.9579 (0.8673)	0.7789 (0.5918)
Model 3	0.2895 (0.3532)	0.2102 (0.2611)	0.754 (0.7555)	0.4737 (0.5408)	0.9474 (0.9048)	0.8 (0.6599)
Model 4	0.2656 (0.334)	0.1673 (0.2499)	0.7339 (0.7755)	0.5789 (0.5476)	0.9368 (0.9116)	0.7789 (0.6633)

## 5.5. Movie data vs. image data

Two ways were tested in order to evaluate a BCS for each movie, the first focused on calculating the BCS for each image and averaging the results of all images. In the second way the features of relevant images were combined with mathematical operations (mean, maximum, minimum and median) followed by evaluating the BCS based on these features. The results for image data were better in all six major measures. The results of polynomial regression on all features on the movie testing movie data are shown in Table 11. The best result for movie data was with the mean value of the features resulting in a MAE that was almost twice (1.71) as large from the MAE of the same model with image data. Even the MDAE that was closest to the result of the same model on image data was still 1.11 of the image data result.

Table 11 - Movie data model results

Mathematical operation	MAE	MDAE	R2	A6	A6 +1	A4
Mean	0.4464	0.2141	-0.2039	0.4211	0.8211	0.6947
Maximum	0.6444	0.547	-0.5495	0.2842	0.7579	0.3895
Minimum	1.0037	0.3421	-30.2052	0.3158	0.7579	0.5368
Median	0.4894	0.4026	0.1056	0.4	0.8316	0.5684

## 5.6. Results correlation

The correlation between the BCS and the model's output of the testing sets was analyzed (Appendix C shows all correlation figures). The best correlation (model 3) is shown in Figure 16. Model 3 (polynomial regression with all values except X6 up to a second cubic equation on image data) has an  $R^2$  of 0.7575 in the test set.

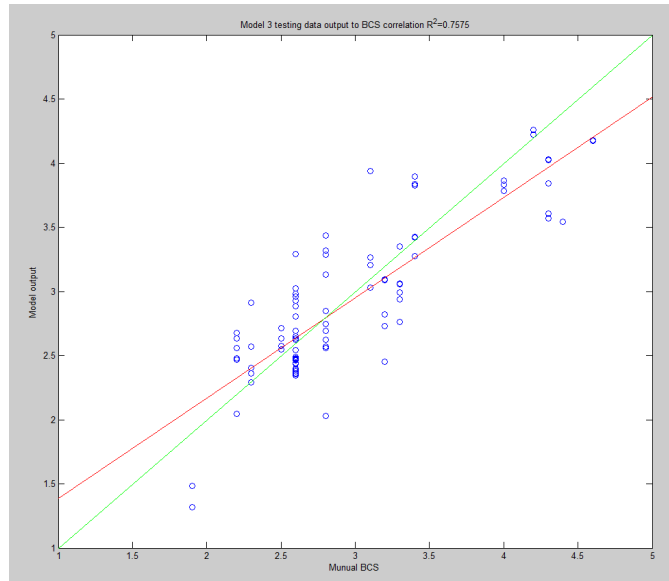


Figure 16 - Model 3 correlation between the model output (Y axis) and the manual BCS (X axis). The red line is the linear fit line and the green line is 1:1

## 5.7. Confusion Matrices

The confusion matrices for models 1 to 4 with (Tables 12 -15) and without (Tables 16 - 19) 'group selection' are presented.

Classification into six classes for model 1 (polynomial regression with all values up to a second cubic equation on image data) results in a good distinction between classes (Table 12) and even classes with relatively small representation in the data set yielded good results (class 1-2 100%, class 3.5-4 100% and class 4-5 56.55%).

Table 12 - Model 1 confusion matrix with 6 classes for testing data

			Model output class						
			1-2	2-2.5	2.5-3	3-3.5	3.5-4	4-5	Total
Manual class	BCS	1-2	3 100% 75%	0 0% 0%	0 0% 0%	0 0% 0%	0 0% 0%	0 0% 0%	3 100% 3.16%
		2-2.5	0 0% 0%	9 60% 90%	6 40% 15%	0 0% 0%	0 0% 0%	0 0% 0%	15 100% 15.79%
		2.5-3	1 2.5% 25%	1 2.5% 10%	28 70% 70%	8 20% 38.1%	2 5% 14.29%	0 0% 0%	40 100% 42.11%
		3-3.5	0 0% 0%	0 0% 0%	6 26.09% 15%	11 47.83% 52.38%	6 26.09% 52.85%	0 0% 0%	23 100% 24.21%
		3.5-4	0 0% 0%	0 0% 0%	0 0% 0%	0 0% 0%	3 100% 21.42%	0 0% 0%	3 100% 3.16%
		4-5	0 0% 0%	0 0% 0%	0 0% 0%	2 18.18% 9.52%	3 27.27% 21.42%	6 54.55% 100%	11 100% 11.58%
	Total		4 4.21% 100%	10 10.53% 100%	40 42.11% 100%	21 22.11% 100%	14 14.74% 100%	6 6.32% 100%	95 100% 100%

Table 13 shows model 2's (polynomial regression with all values except X5 up to a second cubic equation on image data) confusion matrix. Results indicate excellent classification ability for the most common class (2.5-3) with 30 out of 40 (75%) correct classifications and 40 out of 40 (100%) classifications with in a distance of up to 1 class away.

Table 13 - Model 2 confusion matrix with 6 classes for testing data

		Model output class						
		1-2	2-2.5	2.5-3	3-3.5	3.5-4	4-5	Total
Manual BCS class	1-2	3 100% 75%	0 0% 0%	0 0% 0%	0 0% 0%	0 0% 0%	0 0% 0%	3 100% 3.16%
	2-2.5	1 6.67% 25%	10 66.67% 90.91%	4 26.67% 10.26%	0 0% 0%	0 0% 0%	0 0% 0%	15 100% 15.79%
	2.5-3	0 0% 0%	1 2.5% 9.09%	30 75% 76.92%	9 22.5% 36%	0 0% 0%	0 0% 0%	40 100% 42.11%
	3-3.5	0 0% 0%	0 0% 0%	5 21.74% 12.82%	11 47.82% 44%	7 30.43% 63.63%	0 0% 0%	23 100% 24.21%
	3.5-4	0 0% 0%	0 0% 0%	0 0% 0%	0 0% 0%	3 100% 27.27%	0 0% 0%	3 100% 3.16%
	4-5	0 0% 0%	0 0% 0%	0 0% 0%	5 45.45% 20%	1 9.09% 9.09%	5 45.45% 100%	11 100% 11.58%
	Total	4 4.21% 100%	11 11.58% 100%	39 41.05% 100%	25 26.31% 100%	11 11.58% 100%	5 5.26% 100%	95 100% 100%

Table 14 shows model 3's (polynomial regression with all values except X6 up to a second cubic equation on image data) confusion matrix. The model resulted with reasonable results for correct classifications with only 47.37%, however, misclassifications are only 1 classification within a distance of more than 1 class away (a distance of 2 classes); this explains the good results of the model for correct classification into 4 classes (81.05).

Table 14 - Model 3 confusion matrix with 6 classes for testing data

		Model output class						
		1-2	2-2.5	2.5-3	3-3.5	3.5-4	4-5	Total
Manual BCS class	1-2	3 100% 100%	0 0% 0%	0 0% 0%	0 0% 0%	0 0% 0%	0 0% 0%	3 100% 3.16%
	2-2.5	0 0% 0%	6 40% 24%	9 60% 29.03%	0 0% 0%	0 0% 0%	0 0% 0%	15 100% 15.79%
	2.5-3	0 0% 0%	18 45% 72%	16 40% 51.61%	6 15% 35.9%	0 0% 0%	0 0% 0%	40 100% 42.11%
	3-3.5	0 0% 0%	1 4.35% 4%	6 26.09% 19.35%	11 47.83% 64.7%	5 21.74% 38.46%	0 0% 0%	23 100% 24.21%
	3.5-4	0 0% 0%	0 0% 0%	0 0% 0%	0 0% 0%	3 100% 23.06%	0 0% 0%	3 100% 3.16%
	4-5	0 0% 0%	0 0% 0%	0 0% 0%	0 0% 0%	5 45.45% 38.46%	6 54.55% 100%	11 100% 11.58%
	Total	3 3.16% 100%	25 26.32% 100%	31 32.63% 100%	17 17.89% 100%	13 13.68% 100%	6 6.32% 100%	95 100% 100%

Table 15 shows model 4's (stepwise polynomial regression with all values up to a second cubic equation on image data) confusion matrix. The model shows very good symmetry for the middle (and larger) classes with an average class of 3 for class number 3.1 (2.5-3) and an average class of 3.96 for class number 4 (3-3.5).

Table 15 - Model 4 confusion matrix with 6 classes for testing data

			Model output class						
			1-2	2-2.5	2.5-3	3-3.5	3.5-4	4-5	Total
Manual BCS class	1-2	3 100% 75%	0 0% 0%	0 0% 0%	0 0% 0%	0 0% 0%	0 0% 0%	0 0% 0%	3 100% 3.16%
	2-2.5	1 6.67% 25%	9 60% 50%	5 33.3% 16.13%	0 0% 0%	0 0% 0%	0 0% 0%	0 0% 0%	15 100% 15.79%
	2.5-3	0 0% 0%	8 20% 44.44%	22 55% 70.96%	8 20% 34.78%	2 5% 15.38%	0 0% 0%	0 0% 0%	40 100% 42.11%
	3-3.5	0 0% 0%	1 4.35% 5.56%	4 17.39% 12.9%	13 56.52% 56.52%	5 21.73% 38.46%	0 0% 0%	0 0% 0%	23 100% 24.21%
	3.5-4	0 0% 0%	0 0% 0%	0 0% 0%	0 0% 0%	3 100% 23.08%	0 0% 0%	0 0% 0%	3 100% 3.16%
	4-5	0 0% 0%	0 0% 0%	0 0% 0%	2 18.18% 8.69%	3 27.27% 23.08%	6 54.54% 100%	0 0% 0%	11 100% 11.58%
	Total	4 4.21% 100%	18 18.95% 100%	31 32.63% 100%	23 24.21% 100%	13 13.68% 100%	6 6.32% 100%	0 0% 0%	95 100% 100%

Table 16 shows the confusion matrix of model 1 (polynomial regression with all values up to a second cubic equation on image data), with '*group selection*'. Results indicate that the '*group selection*' did not help the confusion matrix in this model with less correct classification (58.95% compared to 63.16% without '*group selection*') and less classification within a distance of up to 1 class away (58.95% compared to 94.74% without '*group selection*'), this is mostly due to the decrease in performance for BCS 2.5-3 that has 15% correct classifications and a double number of classifications with a distance of more than 1 class away for BCS 4-5.

Table 16 - Model 1 confusion matrix with 6 classes for testing data with '*group selection*'

			Model output class						
			1-2	2-2.5	2.5-3	3-3.5	3.5-4	4-5	Total
Manual BCS class	1-2	3	0	0	0	0	0	3	
		100%	0%	0%	0%	0%	0%	100%	
		75%	0%	0%	0%	0%	0%	3.16%	
	2-2.5	0	10	4	1	0	0	15	
		0%	66.67%	26.67%	6.67%	0%	0%	100%	
		0%	58.82%	12.9%	4%	0%	0%	15.79%	
	2.5-3	1	7	22	8	2	0	40	
2.5%		17.5%	55%	20%	5%	0%	100%		
25%		41.17%	70.96%	32%	16.67%	0%	42.11%		
3-3.5	0	0	5	12	6	0	23		
	0%	0%	21.73%	52.17%	26.09%	0%	100%		
	0%	0%	16.12%	48%	50%	0%	24.21%		
3.5-4	0	0	0	0	3	0	3		
	0%	0%	0%	0%	100%	0%	100%		
	0%	0%	0%	0%	25%	0%	3.16%		
4-5	0	0	0	4	1	6	11		
	0%	0%	0%	36.36%	9.09%	54.54%	100%		
	0%	0%	0%	16%	8.33%	100%	11.58%		
Total	4	17	31	25	12	6	95		
	4.21%	17.89%	32.63%	26.32%	12.63%	6.32%	100%		
	100%	100%	100%	100%	100%	100%	100%		

Table 17 shows the confusion matrix of model 2 (polynomial regression with all values except X5 up to a second cubic equation on image data), with '*group selection*'. For model 2's confusion matrix the influence of the '*group selection*' was not as expected with a lower amount of correct classifications (63.16% compared to 65.26% without '*group selection*') and a higher amount of classifications within a distance of 1 class (95.79% compared to 94.74% without '*group selection*'). The result was even more surprising after looking at the MAE measure that was lowered by 0.003 from 0.259 to 0.256 as expected (the errors were reduced); the best recorded result for MAE in a testing set.

Table 17 - Model 2 confusion matrix with 6 classes for testing data with '*group selection*'

		Model output class						
		1-2	2-2.5	2.5-3	3-3.5	3.5-4	4-5	Total
Manual BCS class	1-2	3 100% 75%	0 0% 0%	0 0% 0%	0 0% 0%	0 0% 0%	0 0% 0%	3 100% 3.16%
	2-2.5	1 6.67% 25%	10 66.67% 76.92%	4 26.67% 10.81%	0 0% 0%	0 0% 0%	0 0% 0%	15 100% 15.79%
	2.5-3	0 0% 0%	3 7.5% 23.08%	28 70% 77.68%	8 20% 36.36%	1 2.5% 7.14%	0 0% 0%	40 100% 42.11%
	3-3.5	0 0% 0%	0 0% 0%	5 21.73% 13.51%	11 47.83% 50%	7 30.43% 50%	0 0% 0%	23 100% 24.21%
	3.5-4	0 0% 0%	0 0% 0%	0 0% 0%	0 0% 0%	3 100% 21.43%	0 0% 0%	3 100% 3.16%
	4-5	0 0% 0%	0 0% 0%	0 0% 0%	3 27.27% 13.64%	3 27.27% 21.43%	5 45.45% 100%	11 100% 11.58%
	Total	4 4.21% 100%	13 13.68% 100%	37 38.95% 100%	22 23.16% 100%	14 14.74% 100%	5 5.26% 100%	95 100% 100%

Table 18 shows the confusion matrix of model 3 (polynomial regression with all values except X6 up to a second cubic equation on image data), with ‘group selection’. Model 3 responded to the use of ‘group selection’ with an increase of the MAE and a decrease of the MDAE which is consistent with the expected result of ‘group selection’, this influence cannot be seen in the confusion matrix.

Table 18 - Model 3 confusion matrix with 6 classes for testing data with ‘group selection’

		Model output class						
		1-2	2-2.5	2.5-3	3-3.5	3.5-4	4-5	Total
Manual BCS class	1-2	3 100% 100%	0 0% 0%	0 0% 0%	0 0% 0%	0 0% 0%	0 0% 0%	3 100% 3.16%
	2-2.5	0 0% 0%	4 26.67% 17.39%	11 73.33% 34.38%	0 0% 0%	0 0% 0%	0 0% 0%	15 100% 15.79%
	2.5-3	0 0% 0%	18 45% 78.26%	15 37.5% 46.88%	6 15% 35.2%	1 2.5% 7.69%	0 0% 0%	40 100% 42.11%
	3-3.5	0 0% 0%	1 4.35% 4.35%	6 26.09% 18.75%	11 47.83% 64.71%	4 17.39% 30.77%	1 4.35% 0%	23 100% 24.21%
	3.5-4	0 0% 0%	0 0% 0%	0 0% 0%	0 0% 0%	3 100% 23.08%	0 0% 0%	3 100% 3.16%
	4-5	0 0% 0%	0 0% 0%	0 0% 0%	0 0% 0%	5 45.45% 38.46%	6 54.54% 85.71%	11 100% 11.58%
	Total	3 3.16% 100%	23 24.21% 100%	32 33.68% 100%	17 17.89% 100%	13 13.68% 100%	7 7.37% 100%	95 100% 100%

Table 19 shows the confusion matrix of model 4 (stepwise polynomial regression with all values up to a second cubic equation on image data), with '*group selection*'. In model 4 with '*group selection*' all result were equally good or less good except from the MDAE which improved by 0.015 a result that shows that some predictions were slightly better. However, some results yielded larger deviations, this can also be seen in the confusion matrix with a larger number of classifications in a distance of more than 1 class.

Table 19 - Model 4 confusion matrix with 6 classes for testing data with '*group selection*'

		Model output class						
		1-2	2-2.5	2.5-3	3-3.5	3.5-4	4-5	Total
Manual BCS class	1-2	3 100% 75%	0 0% 0%	0 0% 0%	0 0% 0%	0 0% 0%	0 0% 0%	3 100% 3.16%
	2-2.5	0 0% 0%	10 66.67% 58.82%	4 26.67% 12.9%	1 6.67% 3.85%	0 0% 0%	0 0% 0%	15 100% 15.79%
	2.5-3	1 2.5% 25%	7 7.5% 41.18%	22 55% 70.97%	9 22.5% 34.62%	1 2.5% 9.09%	0 0% 0%	40 100% 42.11%
	3-3.5	0 0% 0%	0 0% 0%	5 21.73% 16.1%	12 52.17% 46.15%	6 26.09% 54.54%	0 0% 0%	23 100% 24.21%
	3.5-4	0 0% 0%	0 0% 0%	0 0% 0%	0 0% 0%	3 100% 27.27%	0 0% 0%	3 100% 3.16%
	4-5	0 0% 0%	0 0% 0%	0 0% 0%	4 36.36% 15.38%	1 9.09% 9.09%	6 54.54% 100%	11 100% 11.58%
	Total	4 4.21% 100%	17 17.89% 100%	31 32.63% 100%	26 27.37% 100%	11 11.58% 100%	6 6.32% 100%	95 100% 100%

## ***Summary***

The confusion matrixes (Tables 12- 19 ) show that all models have good results, some are better in exact matches (models 1 and 2) and some are better having most results within 1 output class away from the target class (model 3 and model 2 with '*group selection*'). All models obtain good results even for low and high BCS (100% for all models in 1-2 BCS and above 45.45% for all models in 4-5 BCS). The best model in accurate classification is model number 2 with 65.26% and the best model for classifications within 1 class distance from the target class is model 3 with 98.95%. All models results are with at least 47% correct classifications, with above 92% classifications within 1 class away from the target class and 100% classifications within 2 classes away from the target class.

## 5.8. Models repeatability

Repeatability analyses (Tables and Figures of detailed analysis are in Appendix D), that include the results of all movies from the same cow, indicate that all models are able to predict values for the same cows with a maximum difference up to 0.7403 with a maximum standard deviation of 0.271. The best model is model 1 with a maximum standard deviation for one cow of 0.195 for a 7 samples repeatability check and a maximum difference for one cow of 0.5224. The '*group selection*' for model 1 decreased the maximum difference for one cow to 0.5087 but increased the maximum standard deviation for one cow to 0.199. Table 20 shows the standard deviation of BCS for movies of the same cow for all models; Table 21 shows the maximum difference of BCS for movies of the same cow for all models.

Table 20 - Standard deviation of cow's movies BCS in models 1 to 4 with and without '*group selection*'

Std of	3061	3009	2961	3174	3155
Model 1	0.195	0.135	0.031	0.164	0.042
Model 2	0.12	0.165	0.17	0.216	0.076
Model 3	0.259	0.073	0.117	0.178	0.043
Model 4	0.212	0.147	0.001	0.213	0.059
Model 1 with ' <i>group selection</i> '	0.183	0.199	0.054	0.171	0.025
Model 2 with ' <i>group selection</i> '	0.119	0.236	0.15	0.244	0.068
Model 3 with ' <i>group selection</i> '	0.271	0.027	0.058	0.202	0.024
Model 4 with ' <i>group selection</i> '	0.195	0.219	0.106	0.211	0.05

Table 21 - Maximum difference of cow's movies BCS in models 1 to 4 with and without '*group selection*'

Maximum difference of	3061	3009	2961	3174	3155
Model 1	0.4575	0.3275	0.044	0.5224	0.0829
Model 2	0.3584	0.3975	0.2401	0.5985	0.1335
Model 3	0.6179	0.1638	0.1655	0.5842	0.0848
Model 4	0.4954	0.3583	0.0019	0.6952	0.1055
Model 1 with ' <i>group selection</i> '	0.4297	0.4811	0.0767	0.5087	0.0487
Model 2 with ' <i>group selection</i> '	0.3178	0.4859	0.212	0.6647	0.1335
Model 3 with ' <i>group selection</i> '	0.7403	0.0611	0.0826	0.5947	0.047
Model 4 with ' <i>group selection</i> '	0.4697	0.5237	0.1497	0.6671	0.0974

## 5.9. Error evaluation

Table 22 shows the evaluation of error rate on the testing set for models 1 to 4 with and without '*group selection*'; when comparing the result of models with '*group selection*' and without '*group selection*', it can be seen that the models with '*group selection*' have less errors in the medium range, with errors between 0.25 – 0.75 (34% vs. 35%, 30% vs. 30%, 39% vs.40% and 22% vs. 25% for models 1 to 4 respectively) and more errors in the small range of 0 - 0.25 and errors in the large of range of 0.75 – 5 (66% vs.65%, 70% vs.70%, 61% vs.60% and 68% vs. 65%). It can also be seen that in all models most errors are in the low range, with errors under 0.25, and very few errors are in the high range, with errors above 0.75. The model with the largest error in the lower scale (0 -0.25) is model 2 with 65% of errors between 0 and 0.25. The model with the lowest amount of large errors is model 3 with no errors above 1 and only 4% errors between 0.75 and 1. Even model 2 with '*group selection*' that has the largest amount of

errors above 0.75 (3% above 1 and 4% between 0.75 and 1) has very high percentage of errors in the low ranges - 63% errors with less than 0.25 and 21% errors between 0.25 and 0.5.

Table 22 - Evaluation of error rate on the testing set

	0-0.25	0.25-0.5	0.5-0.75	0.75-1	1-1.5	1.5-5
Model1	56 (59%)	24 (26%)	9 (9%)	6 (6%)	0 (0%)	0 (0%)
Model2	62 (65%)	16 (17%)	12 (13%)	2 (2%)	3 (3%)	0 (0%)
Model3	53 (56%)	29 (31%)	9 (9%)	4 (4%)	0 (0%)	0 (0%)
Model4	57 (60%)	25 (27%)	8 (8%)	3 (3%)	2 (2%)	0 (0%)
Model 1 with ' <i>group selection</i> '	57 (60%)	23 (25%)	9 (9%)	5 (5%)	1 (1%)	0 (0%)
Model2 with ' <i>group selection</i> '	59 (63%)	20 (21%)	9 (9%)	4 (4%)	3 (3%)	0 (0%)
Model3 with ' <i>group selection</i> '	53 (56%)	25 (27%)	12 (12%)	5 (5%)	0 (0%)	0 (0%)
Model4 with ' <i>group selection</i> '	58 (61%)	23 (25%)	7 (7%)	5 (5%)	2 (2%)	0 (0%)

## 5.10. Sensitivity analyses

### *5.10.1. Training set size and content*

The influence of training set size and training set content (K fold cross validation) on model 1 can be seen in Tables 23 and 24. The size of the training set had little influence on the results. When the training set size is 5912 (50% of the original training set size), the  $R^2$  and MAE were 0.7399 and 0.2671, with a respectively reduce/increase of only 0.013 and 0.007 from the results of models trained with the full training set; even at only 10% of the training data size, the  $R^2$  was still above 0.7 and the MAE was still below 0.3; when only 724 images are used in the training set (5% of the original training set size), performance decreased with an  $R^2$  of 0.6 and an MAE of 0.37.

The data for the K fold cross validation was randomly divided into K groups and each time the training was performed with the extraction of one group. The cross validation average results of  $R^2$  equals to 0.6658 and MAE of 0.3357 (in the worst choice for K); these are, as expected, the worst case estimations of the models ability. The analysis shows that even the poorest results will be with an average MAE less than 0.3357 and an average  $R^2$  larger than 0.6658.

The cross validation results indicate that the content of the training set is important. The gaps in the training data can be seen in two places; the first is the large difference between the average measure and the maximum/minimum measure, and the second is the bad results for the average measures. This is as expected and is intensified since it is impossible to ensure that every training-set has all BCS groups.

The maximum/minimum results are better than the initial results, which show's that the training data in the research was not specific.

## Summary-

Based on the sensitivity analyses the training set size should be above 30 movies and should include data from all BCS groups. The number 30 was calculated as 10% out of 294 movies in the training set. The conclusion that all BCS groups should be in training set is due to the large gap in average scores and best scores in the K fold cross validation with a large K (small number of samples in each group).

Table 23 - Influence of training set size on model 1 R2 and MAE for testing data

Size	Average R2	Average MAE	Average R2 with 'group selection'	Average MAE with 'group selection'
100%	0.753	0.2602	0.7427	0.2621
~90%	0.7591	0.2597	0.7550	0.2591
~80%	0.7504	0.2632	0.7342	0.2706
~70%	0.7455	0.2697	0.7239	0.2759
~60%	0.7513	0.2599	0.7256	0.2708
~50%	0.7399	0.2671	0.7307	0.2682
~40%	0.7384	0.2738	0.7197	0.2788
~30%	0.7468	0.2747	0.7315	0.2106
~20%	0.7448	0.2611	0.7075	0.2766
~10%	0.7332	0.2891	0.7225	0.2901
~5%	0.6009	0.3713	0.5957	0.3718
~3%	0.4973	0.3935	0.3981	0.4236

Table 24 - K fold cross validation of model 1

K	Average R2	Maximum R2	Average MAE	Minimum MAE	Average R2 with 'group selection'	Maximum R2 with 'group selection'	Average MAE with 'group selection'	Minimum MAE with 'group selection'
10	0.6737	0.8437	0.3440	0.2863	0.6592	0.8231	0.3494	0.2944
20	0.6767	0.9387	0.3412	0.1767	0.6633	0.9182	0.3452	0.1746
30	0.6658	0.9527	0.3357	0.1380	0.6480	0.9558	0.3398	0.1330

### 5.10.2. Movies interference analysis

Results of the image processing algorithm, deciding if there is a cow's backend in the image, is presented in Table 25. As aforementioned, 100% of accepted movies were correct and 100% of the movies with no cow's backend were rejected. 92.18% (389) of the movies with the important content of the cows backend were correctly selected and the rest of the movies with the important information that were not selected (7.82% - 33 movies) were all with some obstructions (for example 2 cows standing together) or with very little frames of the relevant data (for example 2 frames from 100 with the relevant data).

Table 25 - Types of automatic movie data analysis

		Algorithm decision		
		Has cow backend	No cow's backend	Total
Real movie content	Has cow backend	389 92.18% 100%	33 7.82% 24.81%	422 100% 80.84%
	No cow's backend	0 0% 0%	100 100% 75.19%	100 100% 19.16%
	Total	389 74.52% 100%	133 25.48% 100%	522 100% 100%

### 5.10.3. Noise analysis

Table 26 shows the results of the noise analysis on the image processing algorithms. The mean absolute error and  $R^2$  are tested for added Gaussian noise to all test data. Table 27 summarizes the results for model 1 prediction with noise in the input data. As can be seen, the Image processing algorithm handle's the noise in the images and extracts correct objects from all ten movies, even with noise ratio of up to 40:1. When images were added with a Gaussian

noise ratio of 20:1 no movies were selected and no images were extracted. The prediction algorithm showed a slightly higher sensitivity to noise with an ability to cope with noise of ratio levels up to 50:1; the results for a noise ratio of 100:1 were very similar to the result with no noise, an MAE of 0.2601 compared to 0.2602 and an  $R^2$  of 0.7531 compared to 0.753 were the result of an added noise with ratio of 50:1 were an MAE of 0.4053 and an  $R^2$  of 0.4750. It must be noted that all the noise is added to the natural noise in the data.

Table 26 - Image processing result with added noise

Gaussian noise ratio of signal to noise	Number of movies selected	Percentage of correctly extracted objects from images
No noise added	10 (100%)	100%
200:1	10 (100%)	100%
100:1	10 (100%)	100%
50:1	10 (100%)	100%
40:1	10 (100%)	100%
20:1	0(0%)	Not relevant

Table 27 - Prediction results of model 1 with added noise

Gaussian noise ratio of signal to noise	MAE	R2	MAE with 'group selection'	R2 with 'group selection'
No noise added	0.2602	0.753	0.2621	0.7427
200:1	0.2602	0.753	0.2621	0.7427
100:1	0.2601	0.7531	0.2641	0.7404
90:1	0.2606	0.7533	0.2662	0.7383
80:1	0.2612	0.7527	0.2663	0.7474
70:1	0.2622	0.7522	0.2652	0.7475
60:1	0.2945	0.7004	0.3001	0.6885
50:1	0.4053	0.4750	0.5106	0.2184
40:1	1.1590	-3.2919	1.4694	-5.1569
20:1	10.3648	-348.5844	14.2413	-586.0408

### 5.11. Summary of models

The four models tested in this research have all shown good results and are all suitable for commercial applications. Each model has its advantages and disadvantages. The most suitable model is model 1, 2 or 3 depending on the preferred performance measure. Model 1 (polynomial regression with all values up to a second cubic equation on image data) has the best repeatability and will be able to reproduce consistent BCS evaluations; however, the measures showing the connection between model 1 evaluations and the manual BCS were not the best, only the median absolute error (model 1 with '*group selection*') of all the major six measures was the best in all models with a value of 0.1608.

Model 2 (polynomial regression with all values except X5 up to a second cubic equation on image data) has the lowest mean absolute error (0.259 and 0.256 for '*group selection*') and the highest percentage of correct classifications for six classes (0.6526% and 0.6316% for '*group selection*') showing that model 2 is the best model for maximizing the number of small errors; the error evaluation performed strengthens this claim, where 65% of the errors for model 2 were smaller than 0.25 (the best result in all four models).

Model 3 (polynomial regression with all values except X6 up to a second cubic equation on image data) has the highest correlation between the predicted BCS and the manual BCS (0.7575), the highest percentage of correct classifications in to four classes (81.05%) and the highest percentage of classifications within a distance of up to one class away for six classes (98.95%) and hence it is the model with best results in minimizing all errors; the same as in model 2 the error evaluation supports this claim and showing that 100% of errors in model 3 are below 1 and 96% of errors in model 3 are below 0.75. The model that is most suitable for the needs of farmers today is model 3 since they need all results to be as close as possible to

the evaluations of a trained expert (instead of most results very close and some very far). For the farmer it is more important to represent the BCS and not to reproduce results close to one another.

## 5.12. Comparison to previous research

The main contributions of this research are:

1. In this research, movie and images were captured and selected automatically ( a fully automated system) as opposed to previous studies in which images were selected manually and/or captured manually,
2. The algorithm presented here has no background dependency. Some of the previous research reported inability to handle different backgrounds (Bercovich, 2012) ;
3. This research used off-the-shelf low cost equipment (Kinect camera) as opposed to past studies that used expensive equipment, Nikon D 7000 DSLR camera (Bercovich, 2012,Bercovich et al., 2013) and InfraCAM SD thermal camera (Halachmi et al., 2008,Halachmi et al., 2013) increasing the commercial applicability.

Additionally, this research improves on most performance measures when compared to previous research. In this research, the correlation between manual and automatic BCS in the testing set was 0.7575 as compared to a correlation of 0.315 in the work of (Halachmi et al., 2008) which used a method of evaluating the BCS according to the differences between a parabolic estimation of the cow's contour and the cow's contour. The methods were improved and retested in (Halachmi et al., 2013) and they showed a Pearson correlation of 0.94 for training data and it can closely be compared to the root of  $R^2$  – in this research which achieved 0.8703 for the testing set. Mean absolute errors of 0.34 (Bercovich, 2012), when using Fourier descriptors to describe the contour, and 0.31 (Azzaro et al., 2011), when using a unified coordinate system to evaluate BCS, were found in previous research as compared to 0.2809 found in the testing set in this research. This research was also tested on a secondary

data set (the testing set) while other studies (Azzaro et al., 2011, Halachmi et al., 2008) were tested only once which strengthens the validity of the thesis results.

## 6. Conclusions and future research

### 6.1. Conclusions

An automatic 3D acquisition system for BCS was set up in the Volcani center ARO research farm; the position and configuration of the system were found to be reliable after four days of unsupervised data acquisition. In this research we developed procedures for automated image selection which resulted in 0% false positives and only 7.82% misses in selecting movies with cows backend included. 14 features were tested and used in the research for developing BCS prediction models and all of them were found useful and with added value to the evaluation of BCS. The pixel data of the 3D images and movies were found to be highly important and helpful with BCS evaluation, showing the added value of 3D computer vision over regular computer vision. Four polynomial regression prediction models were compared. The model that achieved the best results in most measures with good results in all other measures was a polynomial regression up to a second cubic equation on image data including all 14 features except the relative height of back right side of the cow's backend. The results for the testing data of the recommended model are: MAE of 0.2809  $R^2$  of 0.7575, percentage of correct classification of 47.37% percentage of classification up to one class away with 6 classes of 0.9895 and a percentage of correct classification with 4 classes of 0.8105. The model also showed good repeatability with all standard deviations under 0.26, a great error rate with 96% of the errors under 0.75 and all errors under 1.

The image processing algorithm and main regression model sensitivity was tested and they showed good ability to cope with changes in training set sizes and content and to added Gaussian noise. The results were compared to state of the art studies and showed significant improvement,  $R^2$  of 0.7575 compared to 0.315 and MAE of 0.256 compared to 0.31.

This research has several other advantages compared to past studies including automatically captured and selected images, with no background dependencies and low cost imaging using a commercial off-the-shelf Kinect camera.

## 6.2. Future research

The proof of concept for a BCS automatic system has been shown in this study and the 3D computer vision system has found to be suitable to achieve the goal. Further research is still necessary to fully implement automatic BCS in daily dairy farming.

**Data acquisition** – the configuration of the system showed good results for the applied farm the movies acquired were of good quality and were automatically captured without any human involvement. The amount of data collected was in accordance to the size of the herd. The image selection part of the image processing algorithm was able to screen out all movies with no relevant data or with irrelevant data. When installing the system in other farms there is a need for a more specific mechanical and electrical design for the system.

**Research data** – the research was a proof of concept and it is advised to train and test the system on data from several farms and herds. Furthermore it is highly recommended to obtain target values (manual BCS) from more than one expert. Target values from a group of experts will also provide a better understanding of the source of the errors.

**Image processing** – the algorithm excellent results with very good ability to cope with noise were achieved with certain parameters adjusted to the system, the herd and the farm specifically used in this research: the height of the camera, the background image and maximum height and width of the cows, all of these must be considered when installing the system in other farms. The image processing algorithm used a specific version of the Kinect camera with a specific resolution and sensitivity and all changes in the camera will require new

testing and configurations. The algorithm was programmed using Matlab which must be re-programmed in openCV and C++ for faster running time and less CPU and memory usage for any commercial application.

**Feature selection** – the features selected and tested in this research were found to be a good base for the prediction models used. Most of the features were based on the 3D data and represented the relative height of a certain region on the cow or the change in the relative heights. The features based on 3D data have shown to be good features, proving the superiority of 3D cameras over other cameras. Additionally, the features based on past information, like age and last recorded weight, were also helpful in achieving the results in this research showing the importance of using all data sources and combining the data in order to reach the best conclusion and decision.

**Prediction models** – the prediction models chosen for this research were four continuous regression models. The use of regression with a continuous output was chosen after showing better results in the testing measures and even after the partition in to classes, the continuous regression results showed more correct classification and more classification in a distance of up to 1 class in the testing set than the classification models. Some classification models showed better results in the training set but with very high over fitting. In future research or if a commercial system is being designed it is recommended to retest the classification models with a larger data set. The model that showed the best result was model 3 that had all input variable except X6 - one of the relative heights; X6 showed high correlation with X5, and the use of both of them together was probably unnecessary and added noise to the prediction model.

**Movie data vs image data** – prediction models used in this research uses both types of input data, movie data calculated from all selected frames and image data were the output data is

calculated for each movie. The image data was found to be a good choice with better results, probably due to higher ability to cope with the changes in the cows back during walking.

**System output** – the research is intended to help farmers make better decisions; BCS is a very important aspect in decisions about the cows' health. The system outputs a BCS for each cow passing under the camera at night – once a day, this is a big change in farm management that is now working with a BCS only once every few months. The integration of the BCS output with existing information systems in the farm should be thought of and planned accordingly and the way of presenting the BCS and the BCS changes should also be considered. The planning of the process should consider the presentation of the data over time and with comparison to the herds over all BCS data.

**Decision making** - future research may also include planning a decision making process using the new acquired data of daily BCS. The decision making process can consider the proper reactions for changes in BCS and for a BCS which are not corresponding to the calving cycle.

**System automation** – the research added an additional proof of concept, besides the use of 3D computer vision for BCS, which is the ability to fully automate a computer vision system for dairy cow monitoring. The system constructed automatically acquired, selected and analyzed the images and movies of dairy cows leaving the milking parlor with human involvement.

**3D computer vision for detection of specific health problems** - Although the BCS is a very effective measure, future research can also include finding connections between 3D image data and diseases and health problems in dairy cows without the need of the BCS as a mediator. Such connections will not be a replacement for a broad health indicator like the BCS but will give further data on health issues.



## 7. References

- Aguzzi J, Costa C, Fujiwara Y, Iwase R, Ramirez-Llorda E, & Menesatti P (2009). A novel morphometry-based protocol of automated video-image analysis for species recognition and activity rhythms monitoring in deep-sea fauna. *Sensors*, 9(11), 8438-8455.
- Akhloufi M (2013). 3D vision system for intelligent milking robot automation. , 90250N-90250N-10.
- Alver M O, Tennøy T, Alfredsen J A, & Øie G (2007). Automatic measurement of rotifer < i> Brachionus plicatilis densities in first feeding tanks. *Aquacultural Engineering*, 36(2), 115-121.
- Azzari G, Goulden M L, & Rusu R B (2013). Rapid characterization of vegetation structure with a Microsoft Kinect sensor. *Sensors*, 13(2), 2384-2398.
- Azzaro G, Caccamo M, Ferguson J, Battiato S, Farinella G, Guarnera G, Puglisi G, Petriglieri R, & Licitra G (2011). Objective estimation of body condition score by modeling cow body shape from digital images. *Journal of Dairy Science*, 94(4), 2126-2137.
- Bellone M, Messina A, & Reina G (2013). A new approach for terrain analysis in mobile robot applications. 2013 IEEE International Conference on Mechatronics (ICM), 225-230.
- Bercovich A (2012). Automatic cow's body condition scoring Unpublished master's, Ben Gurion University of the Negev, Beer Sheva, Israel
- Bercovich A, Edan Y, Alchanatis V, Moallem U, Parmet Y, Honig H, Maltz E, Antler A, & Halachmi I (2013). Development of an automatic cow body condition scoring using body shape signature and Fourier descriptors. *Journal of Dairy Science*, 96(12), 8047-8059.
- Bewley J, Peacock A, Lewis O, Boyce R, Roberts D, Coffey M, Kenyon S, & Schutz M (2008). Potential for estimation of body condition scores in dairy cattle from digital images. *Journal of Dairy Science*, 91(9), 3439-3453.
- Bolarinwa O A, & Adeola O (2012). Direct and regression methods do not give different estimates of digestible and metabolizable energy of wheat for pigs. *Journal of Animal Science*, 90 Suppl 4, 390-392.
- Burger J, & Geladi P (2006). Hyperspectral NIR imaging for calibration and prediction: a comparison between image and spectrometer data for studying organic and biological samples. *Analyst*, 131(10), 1152-1160.
- Chatterjee S, & Hadi A S (2013). *Regression Analysis by Example*. John Wiley & Sons.
- Chen F, Brown G M, & Song M (2000). Overview of three-dimensional shape measurement using optical methods. *Optical Engineering*, 39(1), 10-22.
- Chen Y R, Chao K, & Kim M S (2002). Machine vision technology for agricultural applications. *Computers and Electronics in Agriculture*, 36(2), 173-191.

Chen Y, Zhang W, Yan K, Li X, & Zhou G (2012). Extracting corn geometric structural parameters using Kinect. , 6673-6676.

Cohen J, Cohen P, West S G, & Aiken L S (2013). *Applied Multiple Regression/Correlation Analysis for the Behavioral Sciences*. Routledge.

Costa C, Antonucci F, Pallottino F, Aguzzi J, Sun D, & Menesatti P (2011). Shape analysis of agricultural products: a review of recent research advances and potential application to computer vision. *Food and Bioprocess Technology*, 4(5), 673-692.

Cyganek B, & Siebert J P (2011). *An Introduction to 3D Computer Vision Techniques and Algorithms*. Wiley, West Sussex UK.

Dale L M, Thewis A, Boudry C, Rotar I, Dardenne P, Baeten V, & Pierna J A F (2013). Hyperspectral imaging applications in agriculture and agro-food product quality and safety control: A review. *Applied Spectroscopy Reviews*, 48(2), 142-159.

Edan Y, Han S, & Kondo N (2009). Automation in agriculture. *Springer Handbook of Automation*, 1095-1128.

Edmonson A, Lean I, Weaver L, Farver T, & Webster G (1989). A body condition scoring chart for Holstein dairy cows. *Journal of Dairy Science*, 72(1), 68-78.

El-laithy R A, Jidong Huang, & Yeh M (2012). Study on the use of Microsoft Kinect for robotics applications. *Position Location and Navigation Symposium (PLANS), 2012 IEEE/ION*, , 1280-1288.

Ferguson J D, Galligan D T, & Thomsen N (1994). Principal descriptors of body condition score in Holstein cows. *Journal of Dairy Science*, 77(9), 2695-2703.

Ferguson J, Azzaro G, & Licitra G (2006). Body condition assessment using digital images. *Journal of Dairy Science*, 89(10), 3833-3841.

Fernández Pierna J, Michotte Renier A, Baeten V, & Dardenne P (2004). IR camera and chemometrics (S VM): The winner combination for the detection of MBM. *Stratfeed Symposium, Namur, Belgium* , 16-18.

Fern'ndez-Baena, A., Susín, A., & Lligadas, X. (2012). Biomechanical validation of upper-body and lower-body joint movements of kinect motion capture data for rehabilitation treatments. In *2012 4th International Conference on Intelligent Networking and Collaborative Systems (INCoS)*, 656-661.

Frost A, Schofield C, Beaulah S, Mottram T, Lines J, & Wathes C (1997). A review of livestock monitoring and the need for integrated systems. *Computers and Electronics in Agriculture*, 17(2), 139-159.

Fu D, Xu L, Li D, & Xin L (2014). Automatic detection and segmentation of stems of potted tomato plant using kinect. *Sixth International Conference on Digital Image Processing. International Society for Optics and Photonics* , 915905-915905-5.

Hady P, Domecq J, & Kaneene J (1994). Frequency and precision of body condition scoring in dairy cattle. *Journal of Dairy Science*, 77(6), 1543-1547.

Halachmi I, Polak P, Roberts D, & Klopčič M (2008). Cow body shape and automation of condition scoring. *Journal of Dairy Science*, 91(11), 4444-4451.

Halachmi I, Klopčič M, Polak P, Roberts D, & Bewley J (2013). Automatic assessment of dairy cattle body condition score using thermal imaging. *Computers and Electronics in Agriculture*, 99, 35-40.

Henry P, Krainin M, Herbst E, Ren X, & Fox D (2014). RGB-D mapping: Using depth cameras for dense 3D modeling of indoor environments. In the 12th International Symposium on Experimental Robotics (ISER), 477-491.

Huang J (2011). Kinerehab: A kinect-based system for physical rehabilitation: A pilot study for young adults with motor disabilities. *Research in developmental disabilities*, 32(6), 319-320.

Imran M, Zurita-Milla R, & Stein A (2013). Modeling Crop Yield in West-African Rainfed Agriculture Using Global and Local Spatial Regression. *Agronomy Journal*, 105(4), 1177-1188.

Izadi S, Kim D, Hilliges O, Molyneaux D, Newcombe R, Kohli P, Shotton J, Hodges S, Freeman D, & Davison A (2011). KinectFusion: real-time 3D reconstruction and interaction using a moving depth camera. In Proceedings of the 24th annual ACM symposium on User interface software and technology, 559-568. JOURNAL

Jeliński T, Du C, Sun D, & Fornal J (2007). Inspection of the distribution and amount of ingredients in pasteurized cheese by computer vision. *Journal of Food Engineering*, 83(1), 3-9.

Ji B, Zhu W, Liu B, Ma C, & Li X (2009). Review of Recent Machine-Vision Technologies in Agriculture. 3, 330-334.

Jungong Han, Ling Shao, Dong Xu, & Shotton J (2013). Enhanced Computer Vision With Microsoft Kinect Sensor: A Review. *IEEE Transactions on Cybernetics*, 43(5), 1318-1334.

Keller J M, & Gader P (1995). Fuzzy logic and the principle of least commitment in computer vision. In Systems, Man and Cybernetics, 1995. IEEE International Conference on Intelligent Systems for the 21st Century., 5, 4621-4625.

Kemphorne D M, Barry M, Zabkiewicz J A, & Young J (2014). 3D digitisation of plant leaves. *ANZIAM Journal*, 55, C138-C152.

Kertz A, Reutzel L, Barton B, & Ely R (1997). Body weight, body condition score, and wither height of prepartum Holstein cows and birth weight and sex of calves by parity: A database and summary. *Journal of Dairy Science*, 80(3), 525-529.

Kim I, Kim M, Chen Y, & Kong S (2004). Detection of skin tumors on chicken carcasses using hyperspectral fluorescence imaging. *Transactions-American Society of Agricultural Engineers*, 47(5), 1785-1792.

Long J S, & Freese J (2006). *Regression Models for Categorical Dependent Variables using Stata*. Stata press.

Lucey T (2004). *Management Information Systems*. Cengage Learning EMEA, London UK.

- Marinello F, Pezzuolo A, Gasparini F, & Sartori L (2013). Three-dimensional sensor for dynamic characterization of soil microrelief. In *Precision agriculture'13*, 71-78.
- Otsu N (1975). A threshold selection method from gray-level histograms. *Automatica*, 11(285-296), 23-27.
- Pepe M S (2003). *The Statistical Evaluation of Medical Tests for Classification and Prediction*. New York:Oxford University Press.
- Pietersma D, Lacroix R, & Wade K (1998). A framework for the development of computerized management and control systems for use in dairy farming. *Journal of Dairy Science*, 81(11), 2962-2972.
- Rafibakhsh N, Lee H F, Gong J, Ko H S, & Mohammadi A (2013). Applying fuzzy logic for optimal placement of XBOX Kinect sensors for industrial applications. *Advanced Materials Research*, 628, 433-439.
- Ren Z, Meng J, Yuan J, & Zhang Z (2011). Robust hand gesture recognition with kinect sensor. In *Proceedings of the 19th ACM international conference on Multimedia*, 759-760.
- Riccioli C, Perez-Marin D, Guerrero-Ginel J E, Saeys W, & Garrido-Varo A (2011). Pixel selection for near-infrared chemical imaging (NIR-CI) discrimination between fish and terrestrial animal species in animal protein by-product meals. *Applied Spectroscopy*, 65(7), 771-781.
- Roche J, Dillon P, Stockdale C, Baumgard L, & VanBaale M (2004). Relationships among international body condition scoring systems. *Journal of Dairy Science*, 87(9), 3076-3079.
- Rodenburg J (2000). *Body Condition Scoring of Dairy Cattle*. 2012(12/04).
- Rossing W, Hogewerf P, Ipema A, Ketelaar-De Lauwere C, & De Koning C (1997). Robotic milking in dairy farming. *NJAS Wageningen Journal of Life Sciences*, 45(1), 15-31.
- Rutten C, Velthuis A, Steeneveld W, & Hogeveen H (2013). Invited review: Sensors to support health management on dairy farms. *Journal of Dairy Science*, 96(4), 1928-1952.
- Schowengerdt R A (2006). *Remote Sensing: Models and Methods for Image Processing*. Academic press.
- Shao B, & Xin H (2008). A real-time computer vision assessment and control of thermal comfort for group-housed pigs. *Computers and Electronics in Agriculture*, 62(1), 15-21.
- Singh T P, Chatli M K, Singh P, & Kumar P (2013). Advances in computer vision technology for foods of animal and aquatic origin-a. *Journal of Meat Science and Technology/ July-September*, 1(2), 40-49.
- Smisek J, Jancosek M, & Pajdla T (2013). 3D with Kinect. *Consumer Depth Cameras for Computer Vision*, 3-25. Springer London.
- Szeliski R (2010). *Computer Vision: Algorithms and Applications*. Springer, New york USA.

- Tanabe R, Cao M, Murao T, & Hashimoto H (2012). Vision based object recognition of mobile robot with Kinect 3D sensor in indoor environment. *SICE 2012 Proceedings of Annual Conference*, 2203-2206.
- Tillett R (1991). Image analysis for agricultural processes: a review of potential opportunities. *Journal of Agricultural Engineering Research*, 50, 247-258.
- Tillett R, Onyango C, & Marchant J (1997). Using model-based image processing to track animal movements. *Computers and Electronics in Agriculture*, 17(2), 249-261.
- Torisawa S, Kadota M, Komeyama K, Suzuki K, & Takagi T (2011). A digital stereo-video camera system for three-dimensional monitoring of free-swimming Pacific bluefin tuna, *Thunnus orientalis*, cultured in a net cage. *Aquatic Living Resources*, 24(02), 107-112.
- Tscharke M, & Banhazi T (2013). Review of methods to determine weight and size of livestock from images. *Australian Journal of Multi-Disciplinary Engineering*, 10(1), 1.
- Vadivambal R, & Jayas D S (2011). Applications of thermal imaging in agriculture and food industry—a review. *Food and Bioprocess Technology*, 4(2), 186-199.
- Vajda F (1994). Techniques and trends in digital image processing and computer vision. *Mathematical Modelling and Simulation of Industrial and Economic Processes*, 1/1.
- Van den Putte A, Govers G, Diels J, Gillijns K, & Demuzere M (2010). Assessing the effect of soil tillage on crop growth: A meta-regression analysis on European crop yields under conservation agriculture. *European Journal of Agronomy*, 33(3), 231-241.
- Van der Stuyft E, Schofield C, Randall J M, Wambacq P, & Goedseels V (1991). Development and application of computer vision systems for use in livestock production. *Computers and Electronics in Agriculture*, 6(3), 243-265.
- Viazzi S, Bahr C, Van Hertem T, Schlageter-Tello A, Romanini C E B, Halachmi I, Lokhorst C, & Berckmans D (2014). Comparison of a three-dimensional and two-dimensional camera system for automated measurement of back posture in dairy cows. *Computers and Electronics in Agriculture*, 100(0), 139-147.
- Wildman E, Jones G, Wagner P, Boman R, Troutt H, & Lesch T (1982). A dairy cow body condition scoring system and its relationship to selected production characteristics. *Journal of dairy science*, 65(3), 495-501.
- Xiong G, Lee D, Moon K R, & Lane R M (2010). Shape similarity measure using turn angle cross-correlation for oyster quality evaluation. *Journal of Food Engineering*, 100(1), 178-186.
- Yoshida K, & Kawasue K (2014). Compact Three-dimensional Vision for Ubiquitous Sensing. *UBICOMM 2014, The Eighth International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies*, 157-163.
- Zhang Z (2012). Microsoft kinect sensor and its effect. *MultiMedia, IEEE*, 19(2), 4-10.

Zion B, Alchanatis V, Ostrovsky V, Barki A, & Karplus I (2008). Classification of guppies' (*Poecilia reticulata*) gender by computer vision. *Aquacultural Engineering*, 38(2), 97-104.

Zion B (2012). The use of computer vision technologies in aquaculture—A review. *Computers and Electronics in Agriculture*, 88, 125-132.

## 8. Appendices

### Appendix A. All models and results

Model	Sub model	Missing feature	Data	'Group selection'	Training / Testing	MAE	MDAE	R2	A6	A6+1	A4
Regression	Polynomial	-	Image	No	Training	0.3302	0.2457	0.7866	0.5306	0.915	0.6633
Regression	Polynomial of roots	-	Image	No	Training	0.3685	0.2723	0.7457	0.4592	0.881	0.6122
Regression	Polynomial	Num 1	Image	No	Training	0.3389	0.2271	0.77	0.5136	0.8915	0.6565
Regression	Polynomial	Num 2	Image	No	Training	0.3445	0.2645	0.7691	0.5136	0.8878	0.6429
Regression	Polynomial	Num 3	Image	No	Training	0.3381	0.2375	0.7836	0.5068	0.9116	0.6735
Regression	Polynomial	Num 4	Image	No	Training	0.3483	0.2712	0.7696	0.5034	0.9014	0.6429
Regression	Polynomial	Num 5	Image	No	Training	0.3684	0.2614	0.7386	0.7558	0.8707	0.5714
Regression	Polynomial	Num 6	Image	No	Training	0.3432	0.2599	0.7713	0.5136	0.9014	0.6497
Regression	Polynomial	Num 7	Image	No	Training	0.344	0.2596	0.7737	0.5136	0.898	0.6395
Regression	Polynomial	Num 8	Image	No	Training	0.3523	0.2456	0.7613	0.5136	0.8912	0.6463
Regression	Polynomial	Num 9	Image	No	Training	0.34	0.2453	0.7766	0.4932	0.915	0.6633
Regression	Polynomial	Num 10	Image	No	Training	0.3341	0.238	0.7796	0.534	0.9116	0.6599
Regression	Polynomial	Num 11	Image	No	Training	0.3307	0.2444	0.7842	0.5272	0.9116	0.6599
Regression	Polynomial	Num 12	Image	No	Training	0.3783	0.2516	0.7297	0.4694	0.8741	0.6667
Regression	Polynomial	Num 13	Image	No	Training	0.3586	0.2623	0.7553	0.4694	0.8878	0.6395
Regression	Polynomial	Num 14	Image	No	Training	0.3359	0.2358	0.7824	0.5034	0.915	0.6565
Regression	linear	-	Image	No	Training	0.4582	0.3901	0.6488	0.3367	0.8503	0.6599
Regression	3 degree polynomial	-	Image	No	Training	0.1955	0.1415	0.9182	0.6497	0.966	0.7415
Regression	Stepwise polynomial	-	Image	No	Training	0.326	0.2384	0.7903	0.534	0.9184	0.6599
Regression	Stepwise 3 degree polynomial	-	Image	No	Training	0.2126	0.1462	0.9041	0.6463	0.9626	0.74115

Regression tree	polynomial	-	Image	No	Training	0.0033	0	0.9991	0.9286	1	0.8741
Regression tree	linear	-	Image	No	Training	0	0	1	0.9218	1	0.8673
Classification tree	Polynomial to values	-	Image	No	Training	0.0016	0	0.9994	1	1	0.881
Classification tree	Linear to values	-	Image	No	Training	0.0028	0	0.9989	0.9932	1	0.8741
Classification tree	Polynomial to 6 classes	-	Image	No	Training	0.0068	0	0.9953	0.9558	0.9558	-
Classification tree	Linear to 6 classes	-	Image	No	Training	0	0	1	0.9595	0.9592	-
ANN	Polynomial 10 nudes	-	Image	No	Training	-	-	-	0.6667	0.8231	-
ANN	Linear 10 nudes	-	Image	No	Training	-	-	-	0.7449	0.8537	-
ANN	Polynomial 100 nudes	-	Image	No	Training	-	-	-	0.7755	0.8333	-
ANN	Linear 100 nudes	-	Image	No	Training	-	-	-	0.8231	0.8844	-
ANN	Polynomial 200 nudes	-	Image	No	Training	-	-	-	0.881	0.9218	-
ANN	Linear 200 nudes	-	Image	No	Training	-	-	-	0.8912	0.9082	-
Regression	Polynomial	-	Mean movie	No	Training	0.2725	0.1926	0.8235	0.5918	0.9592	0.6905
Regression	Polynomial	-	Max movie	No	Training	0.3095	0.2291	0.7657	0.551	0.9354	0.6633
Regression	Polynomial	-	Min movie	No	Training	0.3419	0.2389	0.7172	0.534	0.898	0.6701
Regression	Polynomial	-	Median movie	No	Training	0.2785	0.2028	0.8195	0.5578	0.949	0.6701
Regression	Polynomial	-	Image	Yes	Training	0.3302	0.2304	0.7717	0.5544	0.9082	0.6769
Regression	Polynomial of roots	-	Image	Yes	Training	0.3723	0.2701	0.7284	0.4728	0.881	0.6156
Regression	Polynomial	Num 1	Image	Yes	Training	0.352	0.2474	0.7518	0.517	0.8878	0.6361
Regression	Polynomial	Num 2	Image	Yes	Training	0.3503	0.2542	0.7517	0.5442	0.9116	0.6735
Regression	Polynomial	Num 3	Image	Yes	Training	0.345	0.2396	0.7744	0.5	0.9048	0.6701
Regression	Polynomial	Num 4	Image	Yes	Training	0.3566	0.2763	0.7519	0.4966	0.9014	0.6327
Regression	Polynomial	Num 5	Image	Yes	Training	0.3746	0.2629	0.723	0.483	0.8673	0.5918
Regression	Polynomial	Num 6	Image	Yes	Training	0.3532	0.2611	0.7555	0.5408	0.9048	0.6599
Regression	Polynomial	Num 7	Image	Yes	Training	0.3511	0.2707	0.7584	0.5442	0.9116	0.6599
Regression	Polynomial	Num	Image	Yes	Training	0.3551	0.2532	0.7455	0.534	0.9116	0.6599

		8									
Regression	Polynomial	Num 9	Image	Yes	Training	0.3507	0.2604	0.7614	0.4966	0.898	0.6429
Regression	Polynomial	Num 10	Image	Yes	Training	0.3395	0.2375	0.7636	0.5408	0.9048	0.6497
Regression	Polynomial	Num 11	Image	Yes	Training	0.3427	0.2514	0.7713	0.534	0.9048	0.6497
Regression	Polynomial	Num 12	Image	Yes	Training	0.3877	0.2658	0.7163	0.4796	0.8741	0.6599
Regression	Polynomial	Num 13	Image	Yes	Training	0.3741	0.2876	0.7308	0.4592	0.899	0.6497
Regression	Polynomial	Num 14	Image	Yes	Training	0.3471	0.2528	0.7658	0.5306	0.898	0.6395
Regression	linear	-	Image	Yes	Training	0.4737	0.3843	0.6293	0.3299	0.8469	0.6395
Regression	3 degree polynomial	-	Image	Yes	Training	0.2054	0.1487	0.9097	0.6531	0.9626	0.7449
Regression	Stepwise polynomial	-	Image	Yes	Training	0.334	0.2499	0.7755	0.5476	0.9116	0.6633
Regression	Stepwise 3 degree polynomial	-	Image	Yes	Training	0.2209	0.1611	0.8994	0.6361	0.966	0.7245
Regression tree	polynomial	-	Image	Yes	Training	0.0033	0	0.9991	0.9286	1	0.8741
Regression tree	linear	-	Image	Yes	Training	0	0	1	0.9218	1	0.8673
Classification tree	Polynomial to values	-	Image	Yes	Training	0.0016	0	0.9994	1	1	0.881
Classification tree	Linear to values	-	Image	Yes	Training	0.0028	0	0.9989	0.99932	1	0.8741
Classification tree	Polynomial to 6 classes	-	Image	Yes	Training	0.0068	0	-	-	-	-
Classification tree	Linear to 6 classes	-	Image	Yes	Training	0	0	-	-	-	-
Regression	Polynomial	-	Image	No	Testing	0.2602	0.1921	0.753	0.6316	0.9474	0.7684
Regression	Polynomial of roots	-	Image	No	Testing	0.5774	0.3155	-0.0607	0.4316	0.7368	0.6211
Regression	Polynomial	Num 1	Image	No	Testing	0.2833	0.1954	0.7305	0.5895	0.9263	0.7263
Regression	Polynomial	Num 2	Image	No	Testing	0.2794	0.1894	0.7546	0.4842	0.9684	0.7474
Regression	Polynomial	Num 3	Image	No	Testing	0.2927	0.2791	0.713	0.5579	0.9263	0.6947
Regression	Polynomial	Num 4	Image	No	Testing	0.2884	0.2354	0.7242	0.5684	0.9474	0.6947
Regression	Polynomial	Num 5	Image	No	Testing	0.259	0.1819	0.7404	0.6526	0.9474	0.7789
Regression	Polynomial	Num 6	Image	No	Testing	0.2809	0.2294	0.7575	0.4737	0.9895	0.8105
Regression	Polynomial	Num 7	Image	No	Testing	0.2702	0.2059	0.7528	0.6105	0.9368	0.7895
Regression	Polynomial	Num	Image	No	Testing	0.2612	0.1911	0.756	0.5684	0.9474	0.8

		8									
Regression	Polynomial	Num 9	Image	No	Testing	0.2803	0.1906	0.7176	0.6	0.9368	0.6842
Regression	Polynomial	Num 10	Image	No	Testing	0.2638	0.2069	0.7439	0.6105	0.9368	0.7263
Regression	Polynomial	Num 11	Image	No	Testing	0.2615	0.1854	0.7514	0.6421	0.9474	0.7684
Regression	Polynomial	Num 12	Image	No	Testing	0.4054	0.3973	0.5319	0.3789	0.9053	0.6632
Regression	Polynomial	Num 13	Image	No	Testing	0.3192	0.2118	0.6678	0.6316	0.8842	0.7263
Regression	Polynomial	Num 14	Image	No	Testing	0.2676	0.2112	0.7564	0.6	0.9789	0.7579
Regression	linear	-	Image	No	Testing	0.462	0.4819	0.4253	0.3263	0.8526	0.5158
Regression	3 degree polynomial	-	Image	No	Testing	0.6356	0.5986	- 0.2651	0.2737	0.5368	0.4
Regression	Stepwise polynomial	-	Image	No	Testing	0.2647	0.1817	0.7464	0.5895	0.9474	0.7789
Regression	Stepwise 3 degree polynomial	-	Image	No	Testing	0.6186	0.5915	- 0.1385	0.2947	0.5475	0.4211
Regression tree	polynomial	-	Image	No	Testing	0.6335	0.5	- 0.2953	0.1789	0.7158	0.5684
Regression tree	linear	-	Image	No	Testing	0.6889	0.6	- 0.5204	0.1895	0.6947	0.5053
Classification tree	Polynomial to values	-	Image	No	Testing	0.6875	0.6286	- 0.4542	0.3684	0.6211	0.5579
Classification tree	Linear to values	-	Image	No	Testing	0.6963	0.5	-0.499	0.1579	0.5579	0.4632
Classification tree	Polynomial to 6 classes	-	Image	No	Testing	1.3173	1	- 1.0582	0.2632	0.5789	-
Classification tree	Linear to 6 classes	-	Image	No	Testing	1.4526	1	- 1.2929	0.2632	0.4947	-
ANN	Polynomial 10 nodes	-	Image	No	Testing	-	-	-	0.2421	0.6842	-
ANN	Linear 10 nodes	-	Image	No	Testing	-	-	-	0.2842	0.5263	-
ANN	Polynomial 100 nodes	-	Image	No	Testing	-	-	-	0.2421	0.6105	-
ANN	Linear 100 nodes	-	Image	No	Testing	-	-	-	0.4737	0.6526	-
ANN	Polynomial 200 nodes	-	Image	No	Testing	-	-	-	0.2737	0.7368	-
ANN	Linear 200 nodes	-	Image	No	Testing	-	-	-	0.3579	0.7158	-
Regression	Polynomial	-	Mean movie	No	Testing	0.4464	0.2141	- 0.2039	0.4211	0.8211	0.6947
Regression	Polynomial	-	Max movie	No	Testing	0.6444	0.547	- 0.5495	0.2842	0.7579	0.3895
Regression	Polynomial	-	Min movie	No	Testing	1.0037	0.3421	- 30.205	0.3158	0.7579	0.5368

								2			
Regression	Polynomial	-	Media n movie	No	Testing	0.4894	0.4026	0.1056	0.4	0.8316	0.5684
Regression	Polynomial	-	Image	Yes	Testing	0.2621	0.1608	0.7427	0.5895	0.9263	0.7684
Regression	Polynomial of roots	-	Image	Yes	Testing	0.5958	0.2984	- 0.1085	0.4526	0.7158	0.6105
Regression	Polynomial	Num 1	Image	Yes	Testing	0.2955	0.2213	0.6991	0.5895	0.9263	0.7158
Regression	Polynomial	Num 2	Image	Yes	Testing	0.2771	0.1822	0.7386	0.5263	0.9474	0.7579
Regression	Polynomial	Num 3	Image	Yes	Testing	0.3043	0.2533	0.6914	0.5474	0.9053	0.6842
Regression	Polynomial	Num 4	Image	Yes	Testing	0.2994	0.2396	0.6967	0.5579	0.9474	0.6842
Regression	Polynomial	Num 5	Image	Yes	Testing	0.256	0.1752	0.7211	0.6316	0.9579	0.7789
Regression	Polynomial	Num 6	Image	Yes	Testing	0.2895	0.2102	0.754	0.4737	0.9474	0.8
Regression	Polynomial	Num 7	Image	Yes	Testing	0.2824	0.1999	0.732	0.5895	0.9368	0.7579
Regression	Polynomial	Num 8	Image	Yes	Testing	0.2818	0.2019	0.7316	0.5474	0.9368	0.7895
Regression	Polynomial	Num 9	Image	Yes	Testing	0.2861	0.2004	0.6812	0.5895	0.9158	0.6737
Regression	Polynomial	Num 10	Image	Yes	Testing	0.2711	0.1798	0.7201	0.6211	0.9263	0.7263
Regression	Polynomial	Num 11	Image	Yes	Testing	0.2619	0.1763	0.7392	0.6105	0.9368	0.7684
Regression	Polynomial	Num 12	Image	Yes	Testing	0.397	0.3662	0.5369	0.3789	0.9053	0.6632
Regression	Polynomial	Num 13	Image	Yes	Testing	0.3288	0.2099	0.6458	0.6211	0.8842	0.7474
Regression	Polynomial	Num 14	Image	Yes	Testing	0.2658	0.1971	0.7492	0.5789	0.9684	0.7474
Regression	linear	-	Image	Yes	Testing	0.4572	0.4622	0.4081	0.3158	0.8526	0.5263
Regression	3 degree polynomial	-	Image	Yes	Testing	0.6222	0.5694	- 0.1953	0.2947	0.5579	0.4105
Regression	Stepwise polynomial	-	Image	Yes	Testing	0.2656	0.1673	0.7339	0.5789	0.9368	0.7789
Regression	Stepwise 3 degree polynomial	-	Image	Yes	Testing	0.6092	0.6186	- 0.0912	0.3263	0.5263	0.4316
Regression tree	polynomial	-	Image	Yes	Testing	0.6433	0.5	- 0.3648	0.1789	0.7158	0.5684
Regression tree	linear	-	Image	Yes	Testing	0.6889	0.6	- 0.5204	0.1895	0.6947	0.5053
Classificatio n tree	Polynomial to values	-	Image	Yes	Testing	0.6832	0.5	- 0.4546	0.3684	0.6316	0.5684
Classificatio n tree	Linear to values	-	Image	Yes	Testing	0.6963	0.5	-0.499	0.1579	0.5579	0.4632

Classification tree	Polynomial to 6 classes	-	Image	Yes	Testing	1.3173	1	-	-	-	-
Classification tree	Linear to 6 classes	-	Image	Yes	Testing	1.4744	1.6667	-	-	-	-

## Appendix B. Image processing pseudo code

- Define input folder
- Load background images
- Get list of sub folders
- Get number of sub folders
- For folder number from 3 to number of sub folders
  - Get sub folder name
    - Delete old outputs from sub folder
    - Get cow number from sub folder name
    - Get list of oni files in sub folder
    - Get number of oni files in sub folder
    - For oni file number from 1 to number of oni files in sub folder
      - Get oni file name
      - Connect to Kinect C++
      - Get time from file name
      - Chose background image according to time
      - Get RGB part from background image
      - Get distance part from background image
      - Cover holes (using function) in background distance image
      - Threshold with predetermined threshold the background distance image
      - Get specific threshold for the background distance image
      - Use specific threshold to turn background distance image into black and white image
      - For number of frame from 1 to 100
        - Get frame
        - Get RGB image part from frame

- Get distance image part from frame
- Display RGB part and distance part of frame
- Cover holes (using function) in distance image
- Threshold with predetermined threshold the distance image
- Get specific threshold for the distance image
- Use specific threshold to turn distance image into black and white image
- Subtract background black and white image from black and white image
- Label all black objects in subtracted image
- Get all objects' sizes
- Get largest object label
- Turn all other objects into background (0)
- Convert original frame into double
- Get original frame distance values where the largest object is found
- Perform a [1;-1] convolution on the object to detect edges (other cows)
- Set a threshold for edges
- Perform a [1,-1] convolution on the object to detect edges (other cows)
- Set a threshold for edges
- Turn all edges into background (0)
- Turn image into black and white image
- Label all black objects in subtracted image
- Get all objects' sizes
- If the far left column has values
  - Use second black and white image
- Else use first black and white image
- Get largest object
- Turn all other objects into background (0)

- Set image as distance value where object is found
- Set filter [0.1 0.1 0.1; 0.1 1 0.1; 0.1 0.1 0.1]
- Use filter on image (to get contour)
- Where filter output is bigger than 1 and smaller than 1.8 set value to 1, else zero
- Set filter [0 1 0; 1 10 1; 0 1 0]
- Use filter on contour image (smooth contour)
- If filter output is bigger than 11 then set value to 1, else 0
- Set a matrix of the same size as the image with row numbers as values
- Insert row numbers to every contour pixel
- Get the maximum and minimum rows of the contour for each column
- Get the difference between the maximum and the minimum for all columns
- Get the maximum difference
- Set flag to 1 if there is no columns with contours else set to 0
- Get the number of all contour pixels in every column
- If flag is 0 then
  - Get the contour row difference for the first column with a contour
  - Get the contour row difference for the middle column with a contour
- Get minimum distance in object
- If first column does not have contour and flag=0 and the difference of first column is

lower than the difference of the middle column and the minimum distance is higher than 1100

and the last column has a contour and maximum difference is smaller than 320 and the area of the object is larger than 60000

- Set filter [0.1 0.1 0.1; 0.1 0.2 0.1; 0.1 0.1 0.1]
- Use filter on black and white object
- If filter output is bigger than 0.4 set value to 1 else 0
- Get all added pixels from filter output

- Set added pixels value to average value of their neighbors in the object
- Add pixels in to object distances image
- Sum the object distances image columns
- Get all columns with sum bigger than 0
- Get differences between maximum and minimum row index for all columns
- Get the maximum column index with a difference of less than 50
- Get all columns with an index larger than 10 less of maximum index found
- Sum the image rows
- Get all rows with sum bigger than 0
- If image has more than 20 columns get the 20'st column, else get the 1'st as start

column

- If image has more than 60 columns get the 30'st column, else get the last as end

column

- Get the mean of indexes of the columns were value is not 0
- Subtract the means
- Get the number of columns in the image
- Calculate the arctangent of difference between the means divided by the number of

columns

- Rotate the image by the angle received
- For angle from -3 to 3 with leaps of 0.1
  - Rotate the image by that angle
  - get image symmetry (using function)
- get maximum symmetry and maximum symmetry angle
- Get the minimum value for each column of the object image
- Sort the values for each column of the object image
- Subtract the minimum from all sorted values for each column

- Were the subtracted value is not zero set the index to NAN
- Calculate the mean for each column
- Perform a linear regression on the values
- Calculate the arctangent of second regression coefficient
- Rotate the image by the angle received
- For angle from -3 to 3 with leaps of 0.1
  - Rotate the image by that angle
  - get image symmetry (using function)
- get maximum symmetry and maximum symmetry angle
- Decide which method got the best symmetry and set rotated image to be the winning

method output

- Sum the object distances image columns
- Get all columns with sum bigger than 0
- Transform the image to black and white
- Sum the black and white image columns
- Get the maximum column index with a sum of less than 50 and with an index number

not larger than the number of columns

- Get all columns with an index larger 10 less of maximum index found
- Sum the image rows
- Get all rows with sum bigger than 0
- If number of columns is bigger than 200 than Get 200 first columns else fill in missing

columns with zeroes

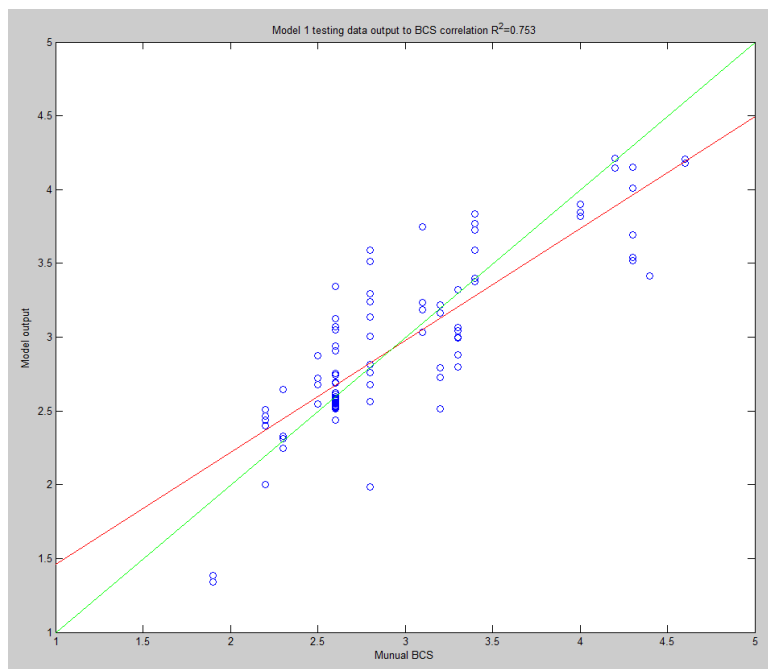
- If number of rows is bigger than 150 than
  - Get columns 31 to 130
  - Get the minimum for each column
  - Sort the values of each column

- Subtract the minimum from all sorted values
- Perform round up mean on all subtracted values
- If the mean is less than 75 set the mean to be 75 if the number of rows minus the mean is less than 75 set the mean to the number of rows minus 75
- Get only the rows that are up 75 rows away from the mean
- Else fill in the missing rows with zeroes half in the begging and half in the end
- Set all zero values to the threshold preformed at the beginning and subtract that threshold from all values
- Divide all values with the maximum value
- Save related data in matrix (maximum height, time, date, frame)
- Save image as jpg
- Save image as matrix

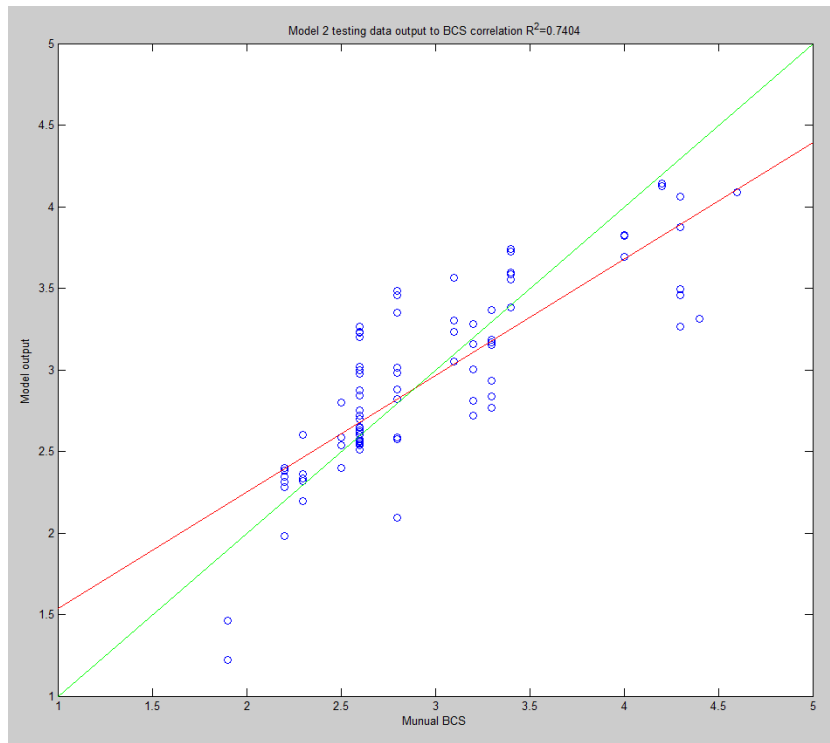
## Appendix C. Correlations between models output and manual BCS

Correlations between the model output (Y axis) and the manual BCS (X axis). The red line is the linear fit line and the green line is 1:1.

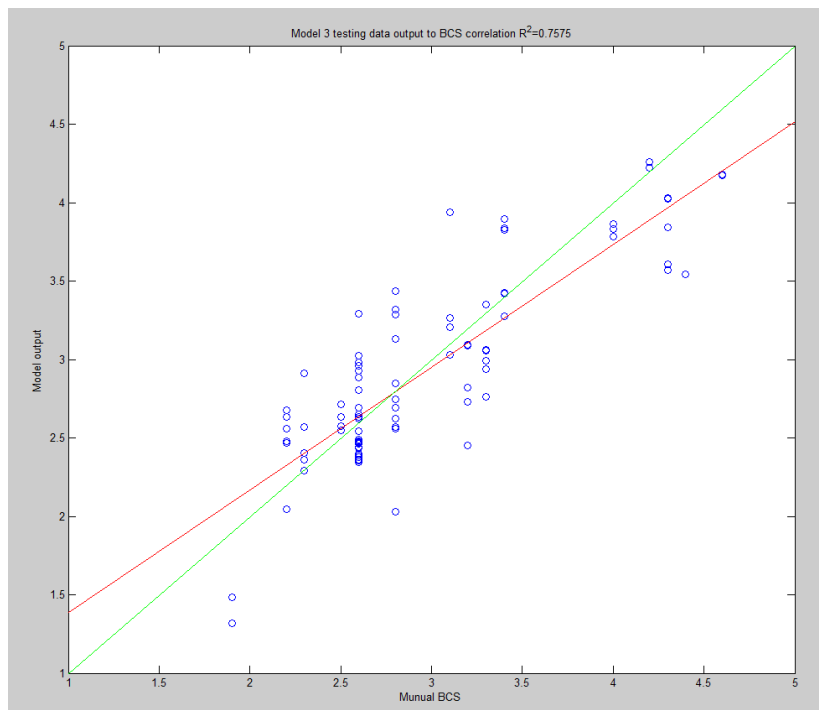
Model 1



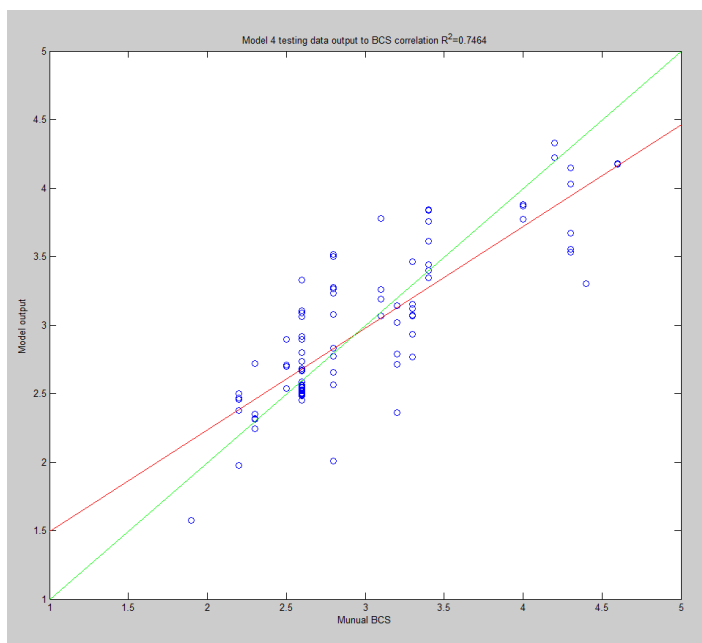
## Model 2



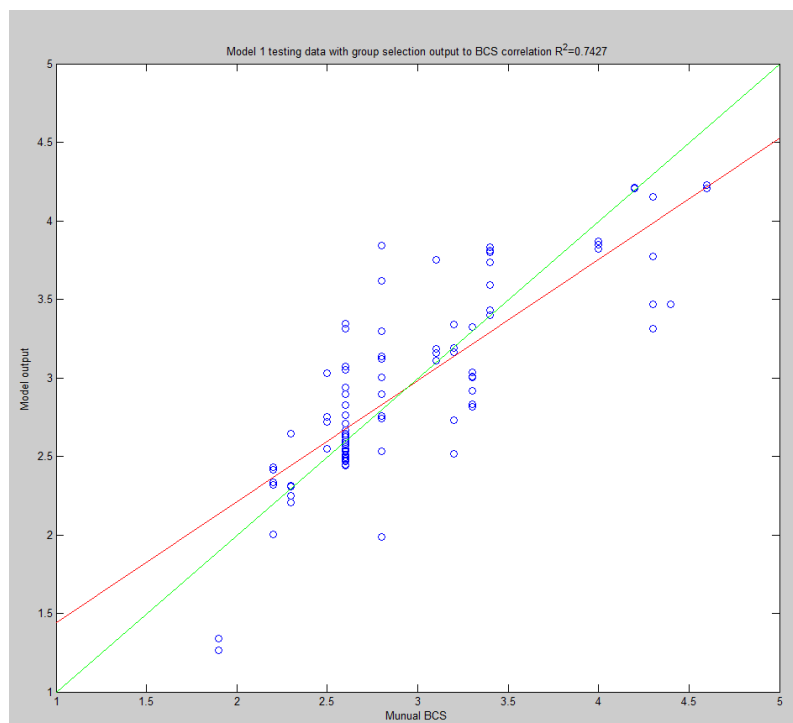
## Model 3



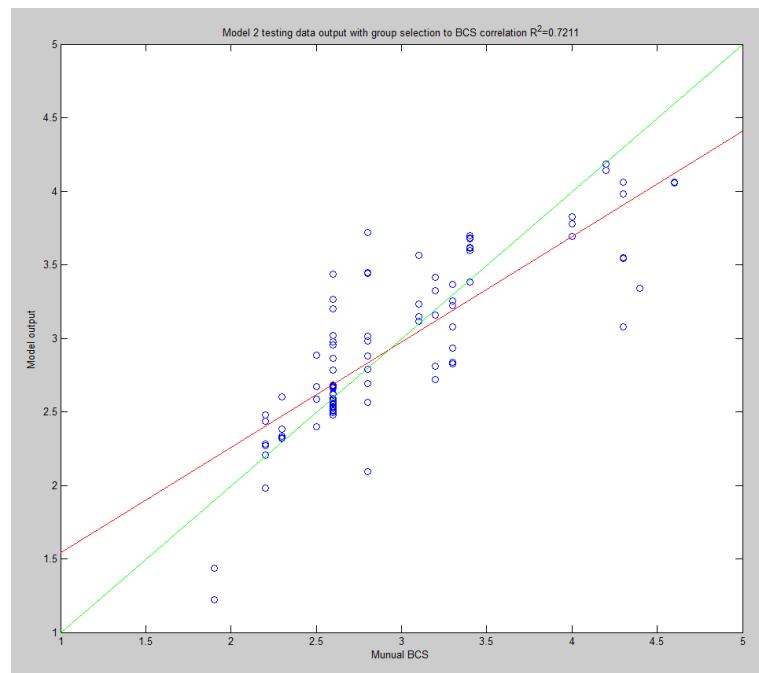
## Model 4



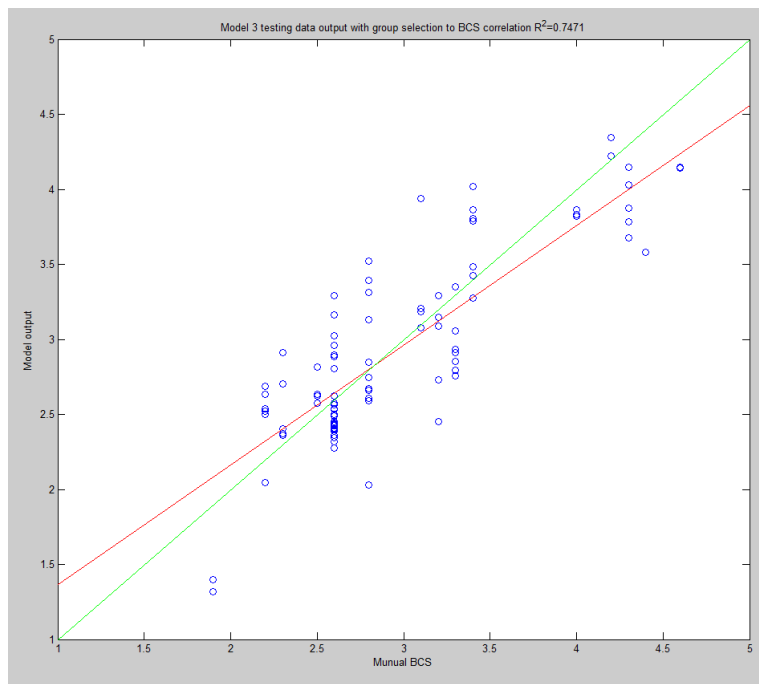
## Model 1 with 'group selection'



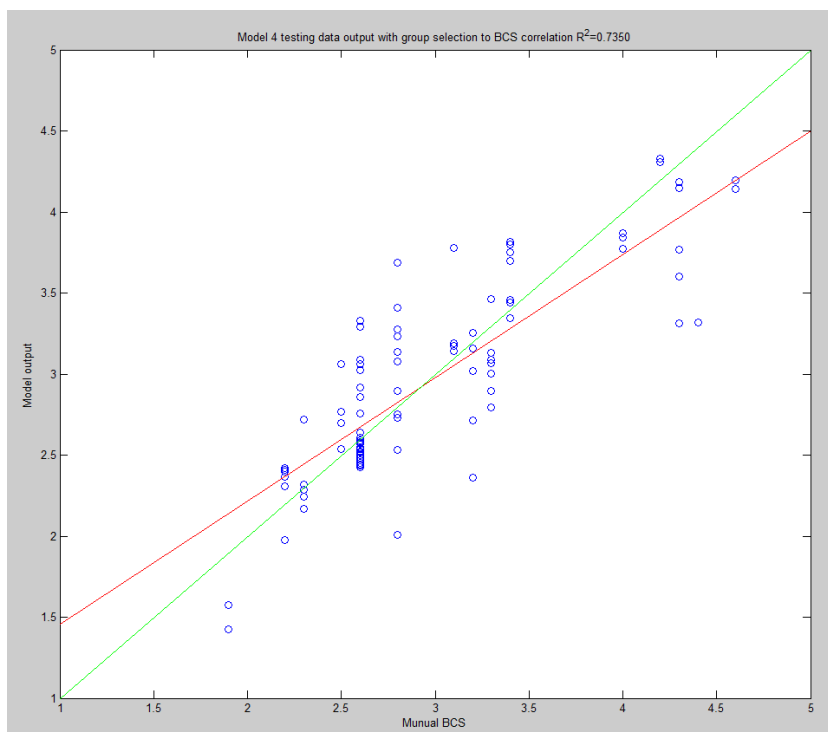
## Model 2 with 'group selection'



## Model 3 with 'group selection'



## Model 4 with 'group selection'

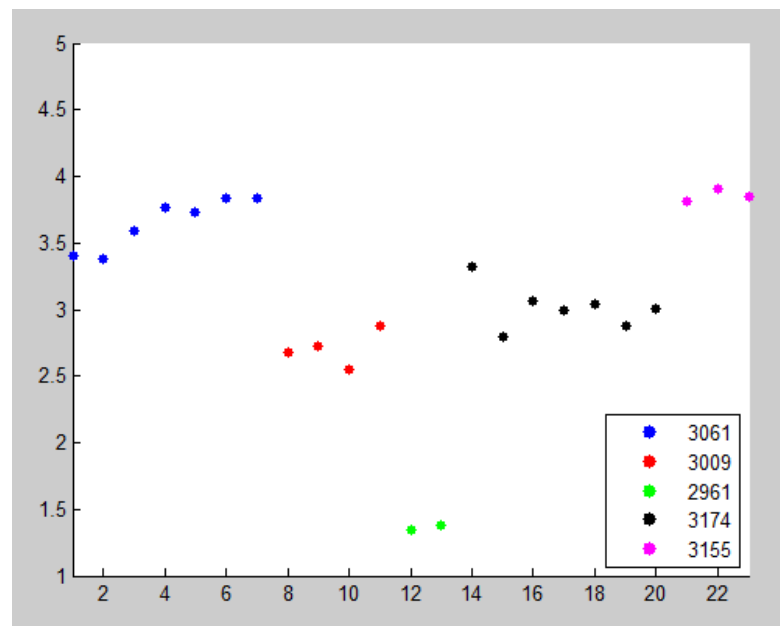


## Appendix D. Models repeatability tables and figures

Model 1 testing data repeatability table

Num	3061	3009	2961	3174	3155
Movie 1	3.4009	2.6793	1.3403	3.3217	3.8184
Movie 2	3.3781	2.7205	1.3843	2.7993	3.9013
Movie 3	3.5921	2.5459	-	3.0638	3.8475
Movie 4	3.7703	2.8734	-	2.9957	-
Movie 5	3.7269	-	-	3.0411	-
Movie 6	3.8356	-	-	2.8826	-
Movie 7	3.8340	-	-	3.004	-
Std	0.195	0.135	0.031	0.164	0.042

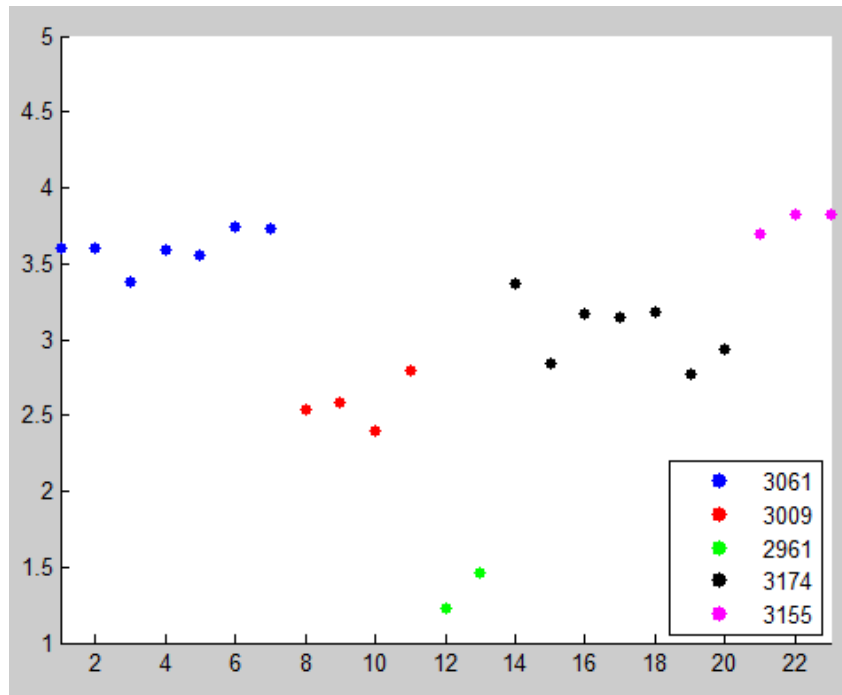
Model 1 testing data repeatability figure



Model 2 testing data repeatability table

Num	3061	3009	2961	3174	3155
Movie 1	3.5977	2.5348	1.2246	3.3676	3.6913
Movie 2	3.5980	2.5879	1.4647	2.8385	3.8223
Movie 3	3.3807	2.3996	-	3.1693	3.8248
Movie 4	3.5857	2.7971	-	3.1514	-
Movie 5	3.5539	-	-	3.1838	-
Movie 6	3.7391	-	-	2.7691	-
Movie 7	3.7268	-	-	2.9317	-
Std	0.12	0.165	0.17	0.216	0.076

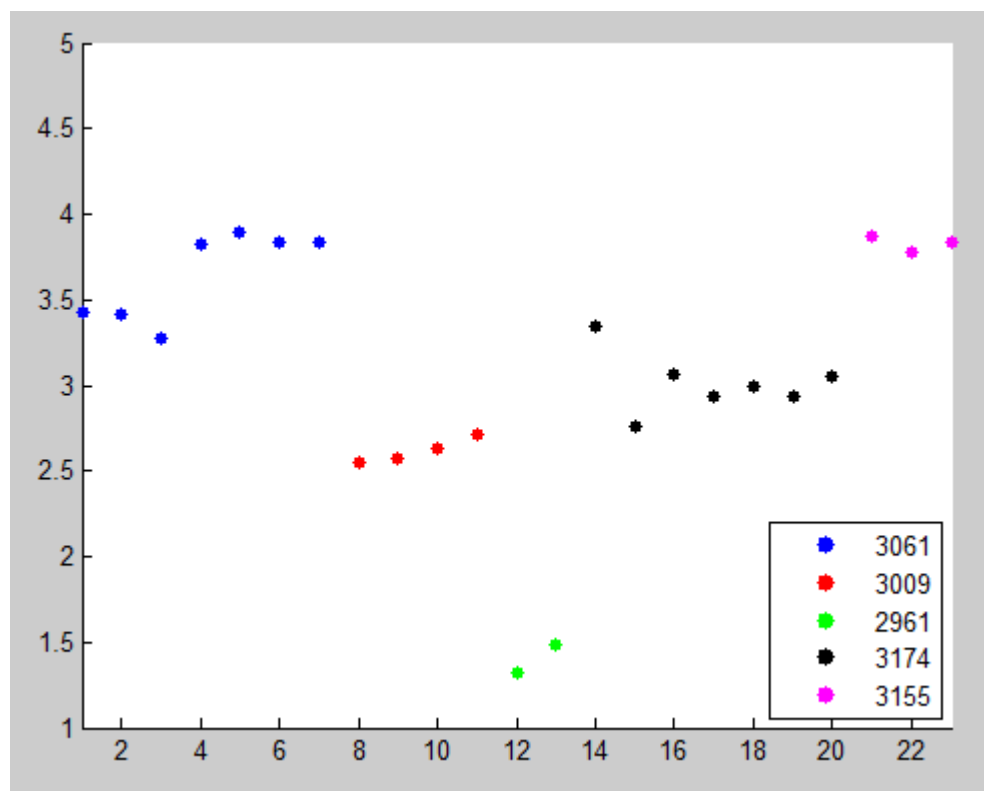
Model 2 testing data repeatability figure



Model 3 testing data repeatability table

Num	3061	3009	2961	3174	3155
Movie 1	3.4259	2.5496	1.3169	3.3489	3.8661
Movie 2	3.4204	2.5747	1.4824	2.7647	3.7813
Movie 3	3.2774	2.6358	-	3.0615	3.8318
Movie 4	3.8272	2.7134	-	2.9398	-
Movie 5	3.8953	-	-	2.9937	-
Movie 6	3.8358	-	-	2.9400	-
Movie 7	3.8363	-	-	3.0583	-
Std	0.259	0.073	0.117	0.178	0.043

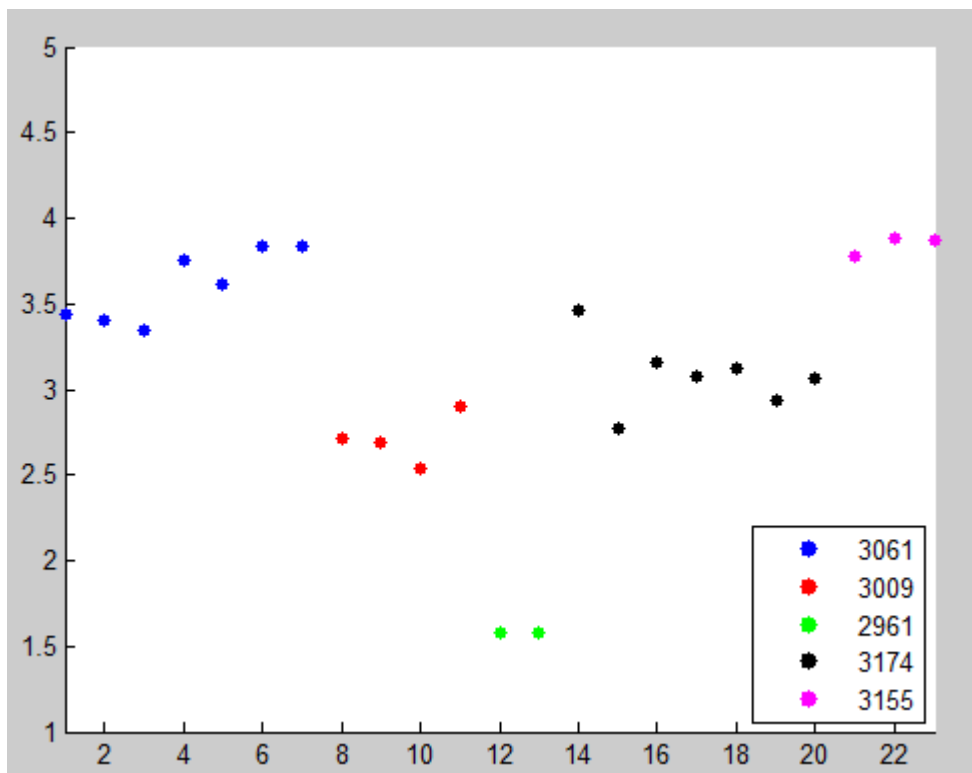
Model 3 testing data repeatability figure



Model 4 testing data repeatability table

Num	3061	3009	2961	3174	3155
Movie 1	3.4398	2.7112	1.5735	3.4620	3.7733
Movie 2	3.4005	2.6959	1.5754	2.7668	3.8788
Movie 3	3.3445	2.5365	-	3.1538	3.8707
Movie 4	3.7543	2.8948	-	3.0702	-
Movie 5	3.6118	-	-	3.1184	-
Movie 6	3.8395	-	-	2.9335	-
Movie 7	3.8399	-	-	3.0659	-
Std	0.212	0.147	0.001	0.213	0.059

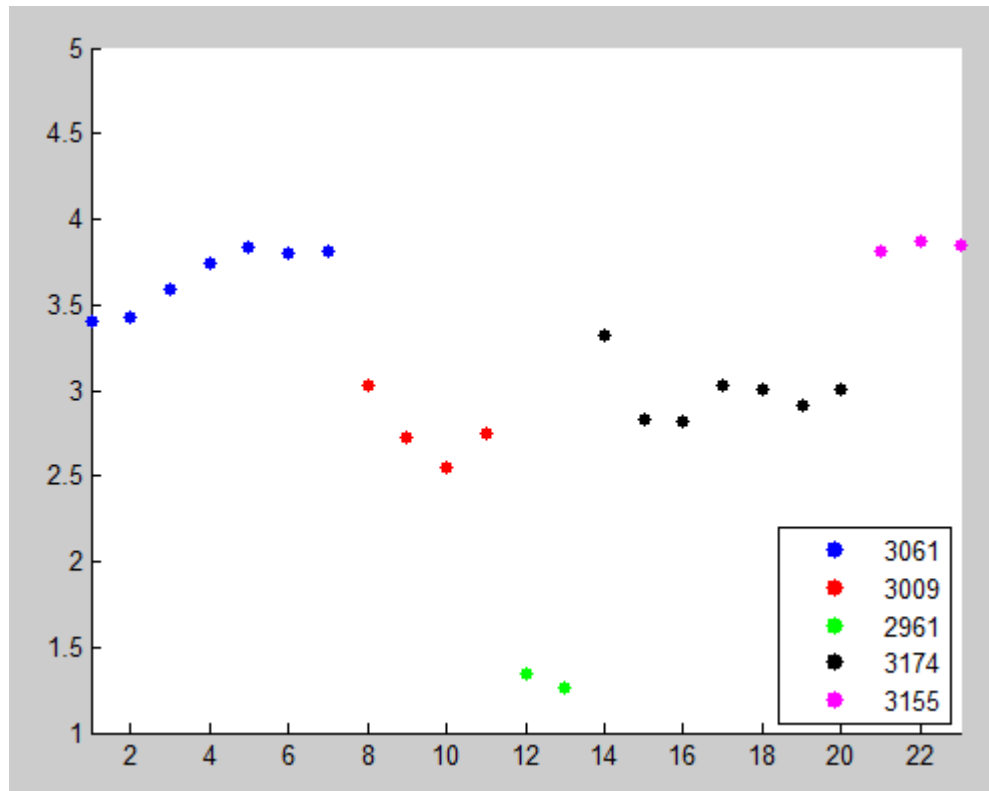
Model 4 testing data repeatability figure



Model 1 testing data repeatability table with 'group selection'

Num	3061	3009	2961	3174	3155
Movie 1	3.4009	3.0270	1.3403	3.3217	3.8184
Movie 2	3.4285	2.7205	1.2636	2.8309	3.8671
Movie 3	3.5921	2.5459	-	2.8130	3.8475
Movie 4	3.7378	2.7525	-	3.0332	-
Movie 5	3.8306	-	-	3.0088	-
Movie 6	3.7980	-	-	2.9153	-
Movie 7	3.8084	-	-	3.0004	-
Std	0.183	0.199	0.054	0.171	0.025

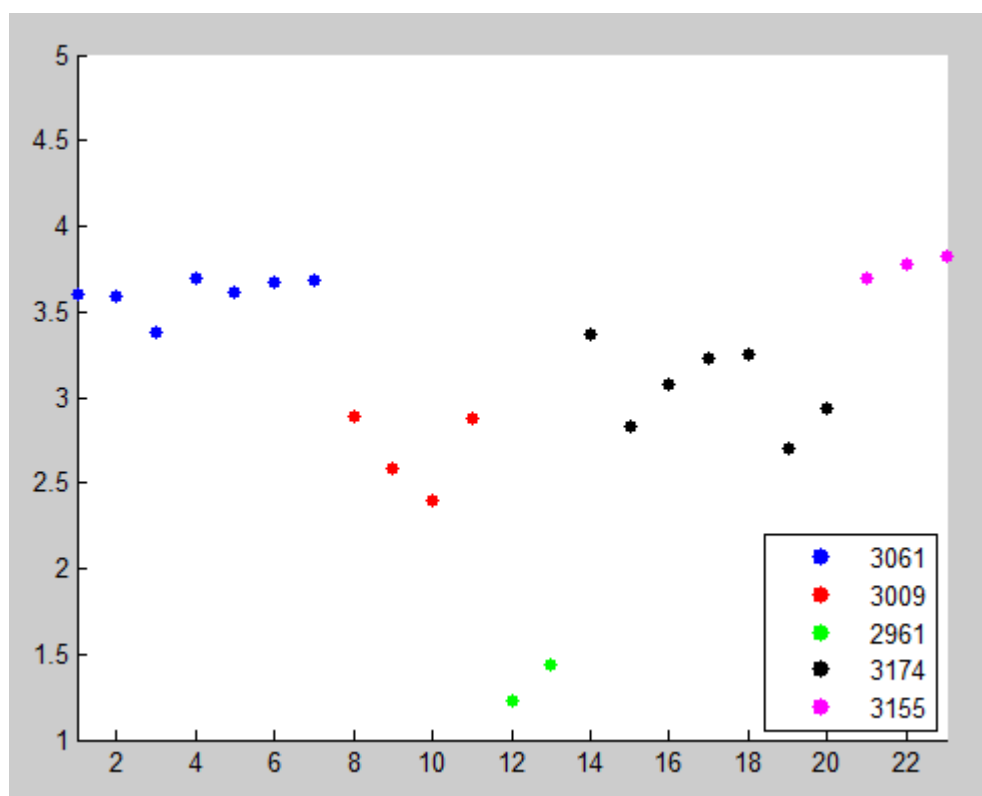
Model 1 testing data repeatability figure with 'group selection'



Model 2 testing data repeatability table with 'group selection'

Num	3061	3009	2961	3174	3155
Movie 1	3.5977	2.8855	1.2246	3.3674	3.6913
Movie 2	3.5851	2.5879	1.4366	2.8253	3.7773
Movie 3	3.3807	2.3996	-	3.0762	3.8248
Movie 4	3.6985	2.8766	-	3.2240	-
Movie 5	3.6164	-	-	3.2564	-
Movie 6	3.6759	-	-	2.7027	-
Movie 7	3.6806	-	-	2.9317	-
Std	0.119	0.236	0.15	0.244	0.068

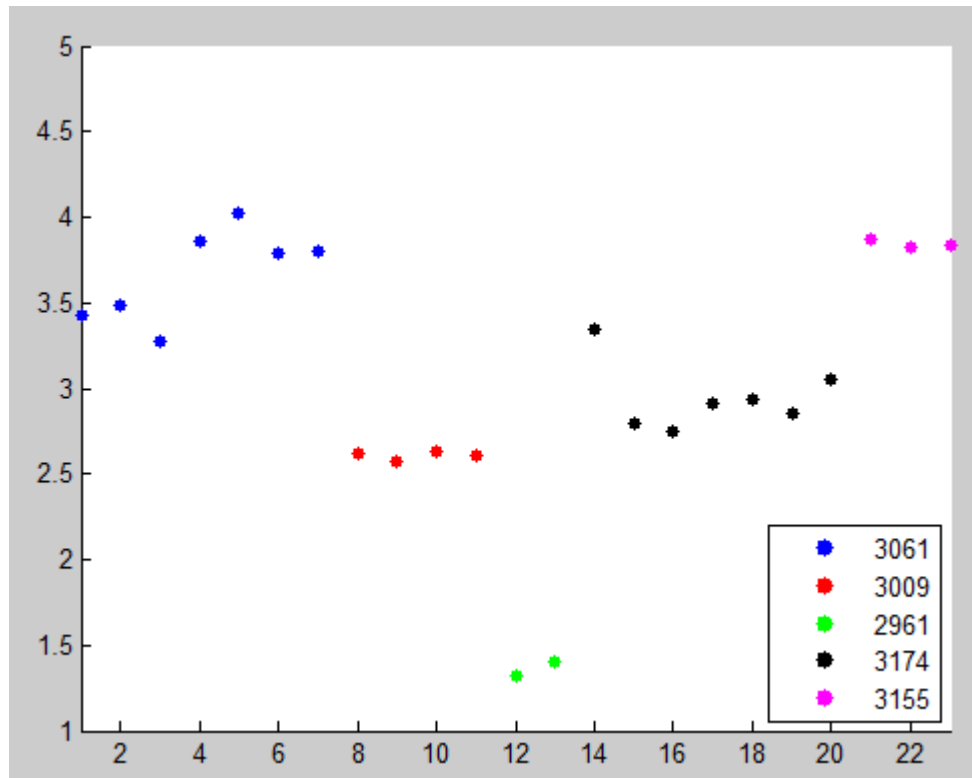
Model 2 testing data repeatability figure with 'group selection'



Model 3 testing data repeatability table with 'group selection'

Num	3061	3009	2961	3174	3155
Movie 1	3.4259	2.6246	1.3169	3.3489	3.8661
Movie 2	3.4837	2.5747	1.3995	2.7915	3.8191
Movie 3	3.2774	2.6358	-	2.7542	3.8318
Movie 4	3.8654	2.6122	-	2.9135	-
Movie 5	4.0177	-	-	2.9340	-
Movie 6	3.7904	-	-	2.8514	-
Movie 7	3.8062	-	-	3.0583	-
Std	0.271	0.027	0.058	0.202	0.024

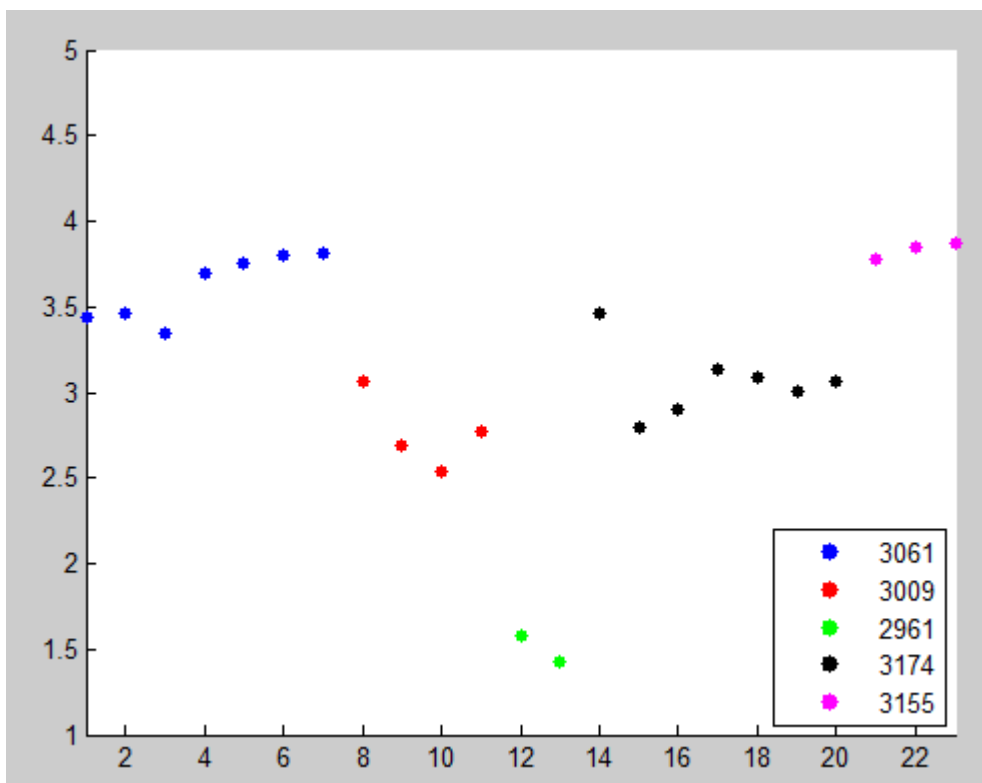
Model 3 testing data repeatability figure with 'group selection'



Model 4 testing data repeatability table with 'group selection'

Num	3061	3009	2961	3174	3155
Movie 1	3.4398	3.0602	1.5735	3.4620	3.7733
Movie 2	3.4573	2.6959	1.4238	2.7949	3.8445
Movie 3	3.3445	2.5365	-	2.8958	3.8707
Movie 4	3.6993	2.7678	-	3.1290	-
Movie 5	3.7517	-	-	3.0885	-
Movie 6	3.8008	-	-	3.0024	-
Movie 7	3.8142	-	-	3.0659	-
Std	0.195	0.219	0.106	0.211	0.05

Model 4 testing data repeatability figure with 'group selection'



## Appendix E. Research data

All 422 movies with cow's back end are stored in disc 1 and disc 2 in folder – moviedata.

All 100 movies without cow's back end are stored in disc 3 in folder – irdata.

Matlab matrix with additional data (number of cow, BCS, age and weight) is stored in disc 3 in folder – additionaldata.

## Appendix F. Matlab codes used in research

Image processing code:

```
%Roi BCS image processing algorithm
%clear
clc;
clear all
%input folder + load data
folder='G:\New folder\';
back=[folder,'background1'];
load(back);
back=[folder,'background11'];
load(back);
dirname=[folder,'test\'];
dirname1=strcat(dirname,'*');
listofcows=dir(dirname1);
numofcows=size(listofcows,1);
jc=0;
j1=0;
j2=0;
cowsh=zeros(200,4);
%run over all folders
for ip1=3:numofcows
    cowname=listofcows(ip1).name;
    if size(cowname,2)>9
        cownum2=cowname(9:12);
    end
    cownum=cowname(1:4);
    subdirname=strcat(dirname,cowname,'\*.mat');
    delete(subdirname);
    subdirname=strcat(dirname,cowname,'\*.jpeg');
    delete(subdirname);
    subdirname=strcat(dirname,cowname,'\*.oni');
    listofoni=dir(subdirname);
    numofoni=size(listofoni,1);
    %run over all movies
    for ip2=1:numofoni
        jc=jc+1;
        oniname=listofoni(ip2).name;
        filename=strcat(dirname,cowname,'\ ',oniname);
```

```

addpath('Mex')
SAMPLE_XML_PATH='Config/SamplesConfig.xml';

% Start the Kinect Process

KinectHandles=mxNiCreateContext(SAMPLE_XML_PATH,filename);
disip=[ip1 ip2];
disp(disip);
figure(1);
I=mxNiPhoto(KinectHandles); I=permute(I,[3 2 1]);
D=mxNiDepth(KinectHandles); D=permute(D,[2 1]);
subplot(1,2,1),h1=imshow(I);
subplot(1,2,2),h2=imshow(D,[0 9000]); colormap('jet');

sp=0;
flags=0;
i=sp;
%run over all images
while i<100

    i=i+1;
    I=mxNiPhoto(KinectHandles); I=permute(I,[3 2 1]);
    D=mxNiDepth(KinectHandles); D=permute(D,[2 1]);
    mxNiUpdateContext(KinectHandles);
    %show images
    set(h1,'CDATA',I);
    set(h2,'CDATA',D);
    titles=[cownum, ' ',int2str(i)];
    figure(1); title (titles);
    drawnow;
    Dt=D;
    %add noise
    %for adding noise
    %Dt=double(Dt);
    %Dt0=Dt;
    %Dt0(Dt~=0)=1;
    %Dt=awgn(Dt,40,'measured');
    %Dt=Dt.*Dt0;
    %Dt=uint16(Dt);
    if str2num(oniname(10))==0 || (str2num(oniname(10))==1 &&
str2num(oniname(12))==0)
        D1t=D11;
    else
        D1t=D1;
    end
    [mt,nt]=size(D1t);
    %clean images
    if i==sp+1
        [mt,nt]=size(Dt);
        Dt=cover_holes(Dt,zeros(mt,nt));
    else
        Dt=cover_holes(Dt,d_e);
    end
    %clear beackgroud
    Dt(Dt>1800)=5000;
    Dt(:,1:7)=5000;
    level=graythresh(Dt);
    Dt_BW=im2bw(Dt,level);
    D1t(D1t>1800)=5000;

```

```

D1t(:,1:7)=5000;
level1=graythresh(D1t);
D1t_BW=im2bw(D1t,level1);
sub=Dt_BW-D1t_BW;
D_lab1=bwlabel(sub,8);
lab_area=regionprops(D_lab1);
big_l=find([lab_area.Area]==max([lab_area.Area]));
big_l=big_l(1);
D_lab=D_lab1;
D_lab(D_lab~=big_l)=0;
D_lab(D_lab==big_l)=1;
flag3=0;
[m,n]=size(D_lab);
Dd=double(Dt);
D2=Dd.*D_lab;
f=[1;-1];
D2t=conv2(D2,f,'same');
D2t1(abs(D2t)>90)=0;
D2t1=D2;
D2t1(abs(D2t)>90 & abs(D2t)<500)=0;
f=[1 -1];
D2t=conv2(D2,f,'same');
D2t1(abs(D2t)>90 & abs(D2t)<500)=0;
D2t1(D2==0)=0;
D2=D2t1;
tempm=1;
D2t_BW=im2bw(D2);
D_lab2=bwlabel(D2t_BW,4);
lab_area1=regionprops(D_lab2);
if sum(D2(:,end))>0
    tempm=0;
    lab_area=lab_area1;
    D_lab=D_lab2;
else
    D_lab=D_lab1;
end
slarge=sort([lab_area.Area]);
isArea=slarge(end-tempm);
big_l=find([lab_area.Area]==slarge(end-tempm));
big_l=big_l(1);
D_lab(D_lab~=big_l)=0;
D_lab(D_lab==big_l)=1;
flag3=0;
[m,n]=size(D_lab);
D2=Dd.*D_lab;
D_c=D_lab;
D_c(2:m-1,2:n-1)=D_lab(1:m-2,1:n-2)*0.1+D_lab(1:m-2,2:n-
1)*0.1+D_lab(1:m-2,3:n)*0.1+D_lab(2:m-1,1:n-2)*0.1+D_lab(2:m-1,2:n-1)+D_lab(2:m-
1,3:n)*0.1+D_lab(3:m,1:n-2)*0.1+D_lab(3:m,2:n-1)*0.1+D_lab(3:m,3:n)*0.1;
D_c(D_c>1 & D_c<1.8)=1;
D_c(D_c<1)=0;
D_c(D_c>=1.8)=0;

D_c3=D_c;
D_c3(2:m-1,2:n-1)=D_c(1:m-2,2:n-1)+D_c(2:m-1,1:n-2)+D_c(2:m-1,2:n-
1)*10+D_c(2:m-1,3:n)+D_c(3:m,2:n-1);
D_c3(m,2:n-1)=D_c(m-1,2:n-1)+D_c(m,1:n-2)+D_c(m,2:n-1)*10+D_c(m,3:n);
D_c3(1,2:n-1)=D_c(1,1:n-2)+D_c(1,2:n-1)*10+D_c(1,3:n)+D_c(2,2:n-1);
D_c3(2:m-1,n)=D_c(2:m-1,n-1)+D_c(1:m-2,n)+D_c(2:m-1,n)*10+D_c(3:m,n);
D_c3(2:m-1,1)=D_c(1:m-2,1)+D_c(2:m-1,1)*10+D_c(3:m,1)+D_c(2:m-1,2);
D_c3(1,1)=D_c(1,1)*10+D_c(1,2)+D_c(2,1);

```

```

D_c3(1,n)=D_c(1,n)*10+D_c(1,n-1)+D_c(2,n);
D_c3(m,1)=D_c(m,1)*10+D_c(m,2)+D_c(m-1,1);
D_c3(m,n)=D_c(m,n)*10+D_c(m,n-1)+D_c(m-1,n);
D_c(D_c3<12)=0;
D_c(D_c3>=12)=1;
xi=1:m;
yi=1:n;
idx=meshgrid(xi,yi)';
D_i=D_c.*idx;
mxD_i=max(D_i);
D_i(D_i==0)=m+1;
mnD_i=min(D_i);
C_s=mxD_i-mnD_i;
C_s(C_s==-(m+1))=0;
[mC_s,imC_s]=max(C_s);
sD_c=sum(D_c);
C_sn0=C_s(C_s>0);
fflag=0;
if isempty(C_sn0)
    fflag=1;
else
    cbw=C_sn0(1);
    iC_sn0=max(floor(size(C_sn0,2)/2),1);
    cmw=C_sn0(iC_sn0);
end
Dtfr=D2;
Dtfr(D2==0)=5000;
%check if image has important data
if sD_c(8)==0 && fflag==0 && cbw<cmw && isArea>60000 &&
min(Dtfr(:))>1100 && sD_c(end)>0 && mC_s<320
    c_pic=D;
    c_D_c=D_c;
    flags=1;
    Dm=D_c;
    Dm(2:m-1,2:n-1)=D_lab(1:m-2,1:n-2)*0.1+D_lab(1:m-2,2:n-
1)*0.1+D_lab(1:m-2,3:n)*0.1+D_lab(2:m-1,1:n-2)*0.1+D_lab(2:m-1,2:n-
1)*0.2+D_lab(2:m-1,3:n)*0.1+D_lab(3:m,1:n-2)*0.1+D_lab(3:m,2:n-
1)*0.1+D_lab(3:m,3:n)*0.1;
    Dm(Dm>0.4)=1;
    Dm(Dm<=0.4)=0;
    Dm_c=Dm-D_lab;
    Dm_c(Dm_c==-1)=0;
    Dm2=zeros(m,n);
    Dm3=zeros(m,n);
    Dm4=zeros(m,n);
    Dm2(2:m-1,2:n-1)=D_lab(1:m-2,1:n-2).*Dd(1:m-2,1:n-2)+D_lab(1:m-
2,2:n-1).*Dd(1:m-2,2:n-1)+D_lab(1:m-2,3:n).*Dd(1:m-2,3:n)+D_lab(2:m-1,1:n-
2).*Dd(2:m-1,1:n-2)+D_lab(2:m-1,3:n).*Dd(2:m-1,3:n)+D_lab(3:m,1:n-2).*Dd(3:m,1:n-
2)+D_lab(3:m,2:n-1).*Dd(3:m,2:n-1)+D_lab(3:m,3:n).*Dd(3:m,3:n);
    Dm3(2:m-1,2:n-1)=D_lab(1:m-2,1:n-2)+D_lab(1:m-2,2:n-1)+D_lab(1:m-
2,3:n)+D_lab(2:m-1,1:n-2)+D_lab(2:m-1,3:n)+D_lab(3:m,1:n-2)+D_lab(3:m,2:n-
1)+D_lab(3:m,3:n);
    Dm4=Dm2./Dm3;
    Dm4(isnan(Dm4))=0;
    Dm5=Dm4.*Dm_c;
    D2=D2+Dm5;
    D_c2=D_c;
    num_c=sum(sum(D_c));
    i2=1;
    j=1;
    flag=0;

```

```

D_c_j=D;
D_c_j(D_c==1)=10000;
cmap = colormap('jet');

%crop image
SD=sum(D2);
D3=D2(:,SD~=0);
D3_BW=im2bw(D3);
D3c=conv2(double(D3_BW),[-1;1],'same');
xi=1:size(D3,1);
yi=1:size(D3,2);
idx=meshgrid(xi,yi)';
D3ci=D3c.*idx;
mxD3c=max(D3ci);
mnD3c=min(D3ci);
gD3c=mxD3c-mnD3c;
[SDs,SDsi]=sort(gD3c);
SDsi(SDs>50)=0;
mSDi=max(SDsi);
D3=D3(:,max(1,mSDi-10):end);
SD=sum(D3,2);
D3=D3(SD~=0,:);
%rotate image
if size(D3,2)>20
    [b1,ix1]=sort(D3(:,20));
else
    [b1,ix1]=sort(D3(:,1));
end
if size(D3,2)>60
    [b2,ix2]=sort(D3(:,end-30));
else
    [b2,ix2]=sort(D3(:,end));
end
ix1_c=ix1(b1~=0);
ix2_c=ix2(b2~=0);
m1=mean(ix1_c);
m2=mean(ix2_c);
h=m1-m2;
w=size(D3,2);
ang=atan(h/w);
D51=imrotate(D2,radtodeg(ang));
D31=D3;
D31(D31==0)=5000;
[am,iam]=min(D31);
[soD3,soD3i]=sort(D31);
soD3=soD3-repmat(am,size(soD3,1),1);
soD3i(soD3~=0)=NaN;
iam=nanmean(soD3i);
xD3=1:size(D31,2);
xD3=[ones(size(xD3')) xD3'];
b=regress(iam',xD3);
ang=atan(b(2));
D53=imrotate(D2,radtodeg(ang));
j3=0;
for i3=-3:0.1:3
    j3=j3+1;
    [angout(j3),msD4(j3) diff(j3)]=rotandsemt(i3,D51);
end
[semt imn]=min(msD4);
[diff1 imnn]=min(diff);
angout1=angout(imn);

```

```

j3=0;
for i3=-3:0.1:3
    j3=j3+1;
    [angout(j3),msD4(j3) diff(j3)]=rotandsemt(i3,D53);
end
[semt2 imn]=min(msD4);
[diff2 imnn]=min(diff);
angout2=angout(imn);
%check for best angle
if semt2<semt
    D51=D53;
    angout1=angout2;
    semt=semt2;
end
D51=imrotate(D51,angout1);
SD=sum(D51);
D41=D51(:,SD~=0);
D3_BW=im2bw(D41);
SD=sum(D3_BW);
[SDs,SDsi]=sort(SD);
SDsi(SDs>50)=0;
SDsi=SDsi(SDsi>0);
SDsi(SDsi>(size(SDsi,2)+10))=0;
mSDi=max(SDsi);
D41=D41(:,max(1,mSDi-10):end);
SD=sum(D41');
D41=D41(SD~=0,:);
%crop image
[mD4,nD4]=size(D41);
if nD4>=200
    D41=D41(:,1:200);
else
    D41=[D41 zeros(mD4,200-nD4)];
end
if mD4>=150
    D4t=D41;
    D4t(D41==0)=5000;
    D4t=D4t(:,31:130);
    [maxD,maxDi]=min(D4t);
    [sortD,sortDi]=sort(D4t);
    sortD=sortD-repmat(maxD,size(sortD,1),1);
    maxDi=ceil(mean(sortDi(sortD==0)));
    if maxDi<75
        maxDi=75;
    elseif mD4-maxDi<75
        maxDi=mD4-75;
    end
    D41=D41(maxDi-74:maxDi+75,:);
    if size(D41,1)>150
        D41=D41(1:150,:);
    end
else
    D41=[zeros(ceil((150-mD4)/2),200);D41;zeros(floor((150-
mD4)/2),200)];
end
D4=D41;
%normalize and write image and pixel data to file
D4(D4==0)=1800;
D4=abs(D4-1800);
Dtt=D4./max(D4(:));
Dtt1=Dtt(1:75,:);

```

```

Dtt2=Dtt(end:-1:76,:);
if sum(sum(Dtt1-Dtt2))<2000
    date=str2num(oniname(5:10));
    time=str2num(oniname(12:17));
    j1=j1+1;
    cowsh(j1,1)=date;
    cowsh(j1,2)=time;
    cowsh(j1,3)=i;
    cowsh(j1,4)=max(D4(:));
    if j1==200
        j2=j2+1;
        j1=0;
        disp(j2);
    end
    D4=D4./max(D4(:));
    jpgname=[filename(1:(end-4)),'_f_',int2str(i),'.jpeg'];
    matname=[filename(1:(end-4)),'_f_',int2str(i),'.mat'];
    imwrite(im2uint8(D4),cmap,jpgname,'jpeg');
    save(matname,'D4');
end
%end
end
d_e=D;
end
% Stop the Kinect Process
mxNiDeleteContext(KinectHandles);
end
end
cowsh=cowsh(1:j1,:);
j2=j2+1;

```

#### Optrotate (function):

```

function [D41 semt angout1]=optrotate(D5)
j=0;
flag=0;
for i=-5:0.1:5
    j=j+1;
    [angout(j),msD4(j)]=rotandsemt(i,D5);
end
[semt imn]=min(msD4);
angout1=angout(imn);
D51=imrotate(D5,angout1);
SD=sum(D51);
D41=D51(:,SD~=0);
D3_BW=im2bw(D41);
SD=sum(D3_BW);
[SDs,SDsi]=sort(SD);
SDsi(SDs>50)=0;
SDsi=SDsi(SDsi>0);
SDsi(SDsi>(size(SDsi,2)+10))=0;
mSDi=max(SDsi);
D41=D41(:,max(1,mSDi-10):end);
SD=sum(D41');
D41=D41(SD~=0,:);
[mD4,nD4]=size(D41);
if nD4>=200
    D41=D41(:,1:200);
else
    D41=[D41 zeros(mD4,200-nD4)];
end

```

```

    if mD4>=150
        D4t=D41;
        D4t(D41==0)=5000;
        D4t=D4t(:,1:100);
        [maxD,maxDi]=min(D4t);
        [sortD,sortDi]=sort(D4t);
        sortD=sortD-repmat(maxD,size(sortD,1),1);
        maxDi=ceil(mean(sortDi(sortD==0)));
        if maxDi<75
            maxDi=75;
        elseif mD4-maxDi<75
            maxDi=mD4-75;
        end
        D41=D41(maxDi-74:maxDi+75,:);
        if size(D41,1)>150
            D41=D41(1:150,:);
        end
    else
        D41=[zeros(150-mD4,200);D41;zeros(150-mD4,200)];
    end
figure(2); imagesc(D41);
    title(sem);
end

```

Cover\_holes(function):

```

function [d_no_h]=cover_holes(d_h1,d_e)
    d_h=d_h1(:,8:end);
    [m,n]=size(d_h);
    md=min(d_h(:));
    while md==0
        for i1=1:m
            for i2=1:n
                temp=0;
                d_temp=0;
                if d_h(i1,i2)==0
                    if d_e(i1,i2)~=0
                        d_h(i1,i2)=d_e(i1,i2);
                    else
                        if i1~=1 && d_h(i1-1,i2)~=0
                            temp=temp+d_h(i1-1,i2);
                            d_temp=d_temp+1;
                        end
                        if i1~=m && d_h(i1+1,i2)~=0
                            temp=temp+d_h(i1+1,i2);
                            d_temp=d_temp+1;
                        end
                        if i2~=1 && d_h(i1,i2-1)~=0
                            temp=temp+d_h(i1,i2-1);
                            d_temp=d_temp+1;
                        end
                        if i2~=n && d_h(i1,i2+1)~=0
                            temp=temp+d_h(i1,i2+1);
                            d_temp=d_temp+1;
                        end
                        d_h(i1,i2)=temp/d_temp;
                    end
                end
            end
        end
        md=min(d_h(:));
    end
end

```

```

end
d_no_h=d_h;
end

```

Get data from images:

```

clc;
clear all
folder='F:\New folder\';
dirname=[folder,'test\'];
dirname1=strcat(dirname,'*');
listofcows=dir(dirname1);
numofcows=size(listofcows,1);
jc=0;
j1=0;
j2=0;
count=0;
count2=0;
matrix=0;
for ip1=3:numofcows
    cowname=listofcows(ip1).name;
    if size(cowname,2)>9
        cownum2=str2num(cowname(10:13));
        bcs2=str2num(cowname(15:17));
    else
        cownum2=0;
    end
    cownum=str2num(cowname(1:4));
    if size(cowname,2)==9
        bcs=str2num(cowname(6:9));
    else
        bcs=str2num(cowname(6:8));
    end
    subdirname=strcat(dirname,cowname,'\*.mat');
    listofmat=dir(subdirname);
    numofmat=size(listofmat,1);
    for ip2=1:numofmat
        lastframenum=0;
        jc=jc+1;
        matname=listofmat(ip2).name;
        filename=strcat(dirname,cowname,'\',matname);
        date=str2num(matname(5:10));
        time=str2num(matname(12:17));
        framenum=str2num(matname(21:end-4));
        load(filename);
        count=count+1;
        D4=D4/D4(76,100);
        if min(D4(:,100:200))>0
            if cownum2>0 && ((framenum>(lastframenum+40) &&
size(cowname,2)==size(lastcowname,2) && min(cowname==lastcowname)) ||
((framenum>60) && (lastframenum==0)))
                matrixrow=[date time framenum cownum2 bcs2 D4(:)'];
            else
                matrixrow=[date time framenum cownum bcs D4(:)'];
            end
            lastframenum=framenum;
            lastcowname=cowname;
            count2=count2+1;

            if matrix==0

```

```

        matrix=matrixrow;
    else
        matrix=[matrix;matrixrow];
    end
    j1=j1+1;
    if j1==200
        j2=j2+1;
        j1=0;
        matrixfilename=[ 'F:\New
folder\test\datamatrix_',int2str(j2),'.mat'];
        save(matrixfilename,'matrix');
        matrix=0;
    end
    end
    disp('count');
    disp(count);
    disp('count2');
    disp(count2);
    disp(size(matrix,1));
end
end
j2=j2+1;
j1=0;
matrixfilename=[ 'F:\New folder\test\datamatrix_',int2str(j2),'.mat'];
save(matrixfilename,'matrix');

```

#### Get distances of contours

```

folder='C:\Users\Roi\Dropbox\stuff\mats\datamat\';
xi=1:200;
yi=1:150;
zi=ones(1,200);
idx=meshgrid(xi,yi,zi);
dis2=[75:-1:1 1:75]';
dis3=repmat(dis2,1,200);
for j=1:81
    matrixname=[folder,'datamatrix_',num2str(j),'.mat'];
    load(matrixname);
    numofmin=zeros(size(matrix,1),1);
    numofmax=zeros(size(matrix,1),1);
    m=size(matrix,1);
    Dall=reshape(matrix(:,6:end)',150,200,m);
    D2all=Dall;
    D2all(Dall>0)=1;
    if m~=200
        zi=ones(1,m);
        idx=meshgrid(xi,yi,zi);
        dis3=repmat(dis2,1,m);
    end
    D2all=D2all.*idx;

    D2all(D2all==0)=201;
    dis=min(D2all,[],2);
    dis=reshape(dis,150,m);
    if j==1
        fftdis2= mean(dis)' ;
    else
        fftdis2=[fftdis2; mean(dis)'];
    end
    disp(j);
end

```

Calculate correlations:

```
for i=1:30
    for j=1:81

matrixname=['C:\Users\Roi\Dropbox\stuff\mats\mats\datamatrix_',num2str(j),'.mat'];
        load(matrixname);
        if j==1
%strtemp=matrix(:,1)*10000000000000+matrix(:,2)*10000000+matrix(:,3)*10000+matrix(:,4);
            %tempforcor=[strtemp matrix(:,5) matrix(:,i)];
            tempforcor=[matrix(:,4) matrix(:,5) matrix(:,(i-1)*1000+6:i*1000+5)];
        else

%strtemp=matrix(:,1)*10000000000000+matrix(:,2)*10000000+matrix(:,3)*10000+matrix(:,4);
            %temp=[strtemp matrix(:,5) matrix(:,i)];
            temp=[matrix(:,4) matrix(:,5) matrix(:,(i-1)*1000+6:i*1000+5)];
            tempforcor=[tempforcor; temp];
        end
    end
    tempforcor=tempforcor(not(ismember(tempforcor(:,1),cowstest)),2:end);
    [r,p] = corrcoef(tempforcor);
    corrmatrix((i-1)*1000+1:i*1000)=r(1,2:end);
    disp(i);
end
save('corr.mat','corrmatrix');
```

Find and analyze regression models:

```
load('bcs.mat');
load('ch.mat');
load('cowstest2.mat');
load('fbo60a.mat');
load('fbozo.mat');
folder='C:\Users\Roi\Dropbox\stuff\mats\datamat\';
for j=1:81
    matrixname=[folder,'datamatrix_',num2str(j),'.mat'];
    load(matrixname);
    numofmin=zeros(size(matrix,1),1);
    numofmax=zeros(size(matrix,1),1);
    for i1=1:size(matrix,1)
        D4=reshape(matrix(i1,6:end),150,200);
        m1=150;
        n1=200;
        mnpD=zeros(m1,n1);
        mnpD(2:m1-1,2:n1-1)=(D4(2:m1-1,2:n1-1)<D4(1:m1-2,2:n1-1));
        mnpD(2:m1-1,2:n1-1)=mnpD(2:m1-1,2:n1-1).*(D4(2:m1-1,2:n1-1)<D4(3:m1,2:n1-1));
        mnpD(2:m1-1,2:n1-1)=mnpD(2:m1-1,2:n1-1).*(D4(2:m1-1,2:n1-1)<D4(2:m1-1,1:n1-2));
        mnpD(2:m1-1,2:n1-1)=mnpD(2:m1-1,2:n1-1).*(D4(2:m1-1,2:n1-1)<D4(2:m1-1,3:n1));
        numofmin(i1)=sum(mnpD(:));
        D4(D4==0)=1;
        mxpD=zeros(m1,n1);
        mxpD(2:m1-1,2:n1-1)=(D4(2:m1-1,2:n1-1)>D4(1:m1-2,2:n1-1));
        mxpD(2:m1-1,2:n1-1)=mxpD(2:m1-1,2:n1-1).*(D4(2:m1-1,2:n1-1)>D4(3:m1,2:n1-1));
        mxpD(2:m1-1,2:n1-1)=mxpD(2:m1-1,2:n1-1).*(D4(2:m1-1,2:n1-1)>D4(2:m1-1,1:n1-2));
    end
end
```

```

mxpD(2:m1-1,2:n1-1)=mxpD(2:m1-1,2:n1-1).*(D4(2:m1-1,2:n1-1)>D4(2:m1-
1,3:n1));
numofmax(i1)=sum(mxpD(:));
end
tempch=ch((j-1)*200+1:min(j*200,end),:);
tempfft=fft2((j-1)*200+1:min(j*200,end),:);
matrix=[matrix numofmax numofmin tempch];
temp=matrix(not(ismember(matrix(:,4),cowstest)),:);
temp2=tempfft(not(ismember(matrix(:,4),cowstest)),:);
strtemp=temp(:,4)*1000000000000+temp(:,1)*1000000+temp(:,2);
if j==1
    movienum=strtemp;
    cownumtd=temp(:,4);
    ytd=temp(:,5);
    temp=temp(:,6:end);
    tdbo60a=temp(:,fbo60a);
    tdbozo=temp(:,fbozo);
    chtd=temp(:,end);
    tdnummin=temp(:,end-1);
    tdnummax=temp(:,end-2);
    temp=temp(:,1:end-3);
    temp(temp==0)=NaN;
    mtd=nanmean(temp,2);
    stdtd=nanstd(temp,0,2);
    ffttd=temp2;
    %cowbcs=matrix(:,5);
else
    movienum=[movienum;strtemp];
    cownumtd=[cownumtd;temp(:,4)];
    ytd=[ytd;temp(:,5)];
    temp=temp(:,6:end);
    tdbo60a=[tdbo60a;temp(:,fbo60a)];
    tdbozo=[tdbozo;temp(:,fbozo)];
    chtd=[chtd;temp(:,end)];
    tdnummin=[tdnummin;temp(:,end-1)];
    tdnummax=[tdnummax;temp(:,end-2)];
    temp=temp(:,1:end-3);
    temp(temp==0)=NaN;
    mtd=[mtd;nanmean(temp,2)];
    stdtd=[stdtd;nanstd(temp,0,2)];
    ffttd=[ffttd;temp2];
    %cowbcs=[cowbcs;matrix(:,5)];
end
temp=matrix(ismember(matrix(:,4),cowstest),:);
temp2=tempfft(ismember(matrix(:,4),cowstest),:);
strtemp=temp(:,4)*1000000000000+temp(:,1)*1000000+temp(:,2);
if j==1
    movienumts=strtemp;
    cownumtsd=temp(:,4);
    ytsd=temp(:,5);
    temp=temp(:,6:end);
    tsdbo60a=temp(:,fbo60a);
    tsdbozo=temp(:,fbozo);
    chtsd=temp(:,end);
    tsdnummin=temp(:,end-1);
    tsdnummax=temp(:,end-2);
    temp=temp(:,1:end-3);
    temp(temp==0)=NaN;
    mtsd=nanmean(temp,2);
    stdtsd=nanstd(temp,0,2);
    ffttsd=temp2;

```

```

        %cowbcs=matrix(:,5);
    else
        movienumts=[movienumts;strtemp];
        cownumtsd=[cownumtsd;temp(:,4)];
        ytsd=[ytsd;temp(:,5)];
        temp=temp(:,6:end);
        tsdbo60a=[tsdbo60a;temp(:,fbo60a)];
        tsdbozo=[tsdbozo;temp(:,fbozo)];
        chtsd=[chtsd;temp(:,end)];
        tsdnummin=[tsdnummin;temp(:,end-1)];
        tsdnummax=[tsdnummax;temp(:,end-2)];
        temp=temp(:,1:end-3);
        temp(temp==0)=NaN;
        mtsd=[mts;nanmean(temp,2)];
        stdtsd=[stdtsd;nanstd(temp,0,2)];
        ffttsd=[ffttsd;temp2];
        %cowbcs=[cowbcs;matrix(:,5)];
    end
    disp(j);
end
xtd=[mean(tdbozo(:,1:10),2) mean(tdbozo(:,11:20),2) mean(tdbozo(:,21:30),2)
mean(tdbozo(:,31:40),2) mean(tdbozo(:,41:50),2) mean(tdbozo(:,51:60),2) ...
mean(tdbozo(:,61:70),2) mean(tdbozo(:,71:80),2)
    mtd stdtd chtd tdnummin tdnummax];
y6cl=zeros(size(ytd,1),6);
y6cl(ytd<=2,1)=1;
y6cl(ytd>2 & ytd<=2.5,2)=1;
y6cl(ytd>2.5 & ytd<=3,3)=1;
y6cl(ytd>3 & ytd<=3.5,4)=1;
y6cl(ytd>3.5 & ytd<=4,5)=1;
y6cl(ytd>4,6)=1;
bcsds=dataset(bcs(:,1),bcs(:,3));
tdw=dataset(cownumtd,'VarNames',{'Var1'});
jtda=join(tdw,bcsds);
jtdage=jtda.Var2;
bcsds=dataset(bcs(:,1),bcs(:,5));
jtdw=join(tdw,bcsds);
jtdwe=jtdw.Var2;
xtd=[xtd jtdwe jtdage ffttd];
xtdl=xtd;
xtd(xtd(:,9)==0,9)=mean(xtd(:,9));
xtd(xtd(:,10)==0,10)=mean(xtd(:,10));
%xtd=xtd(:,[1:5 7:end]);
xtdsize=size(xtd,2);
for i=1:xtdsize
    for j=i:xtdsize
        xtd=[xtd xtd(:,i).*xtd(:,j)];
    end
end
%{
xtdsize2=size(xtd,2);
for i=1:xtdsize
    for j=xtdsize+1:xtdsize2
        xtd=[xtd xtd(:,i).*xtd(:,j)];
    end
end
%}
mxtd=mean(xtd(:,9:end));
mxxtd=max(xtd(:,9:end));
mnxtd=min(xtd(:,9:end));

```

```

xtd(:,9:end)=(xtd(:,9:end)-
repmat(mnxttd,size(xtd,1),1))./repmat(mxxttd,size(xtd,1),1);
xtsd=[mean(tsdbozo(:,1:10),2) mean(tsdbozo(:,11:20),2) mean(tsdbozo(:,21:30),2)
mean(tsdbozo(:,31:40),2) mean(tsdbozo(:,41:50),2) mean(tsdbozo(:,51:60),2) ...
mean(tsdbozo(:,61:70),2) mean(tsdbozo(:,71:80),2)
mtsd stdtsd chtsd tsdnummin tsdnummax];
ys6cl=zeros(size(ytsd,1),6);
ys6cl(ytsd<=2,1)=1;
ys6cl(ytsd>2 & ytsd<=2.5,2)=1;
ys6cl(ytsd>2.5 & ytsd<=3,3)=1;
ys6cl(ytsd>3 & ytsd<=3.5,4)=1;
ys6cl(ytsd>3.5 & ytsd<=4,5)=1;
ys6cl(ytsd>4,6)=1;
bcsds=dataset(bcs(:,1),bcs(:,3));
tsdw=dataset(cownumtsd,'VarNames',{ 'Var1' });
jtsda=join(tsdw,bcsds);
jtsdage=jtsda.Var2;
bcsds=dataset(bcs(:,1),bcs(:,5));
jtsdw=join(tsdw,bcsds);
jtsdwe=jtsdw.Var2;
xtd=[xtd jtsdwe jtsdage ffttsd];
xtdl=xtd;
%xtsd=xtsd(:,[1:5 7:end]);
xtsdsz=size(xtd,2);
for i=1:xtsdsz
    for j=i:xtsdsz
        xtd=[xtd xtd(:,i).*xtd(:,j)];
    end
end
%{
xtsdsz2=size(xtd,2);
for i=1:xtsdsz
    for j=xtsdsz+1:xtsdsz2
        xtd=[xtd xtd(:,i).*xtd(:,j)];
    end
end
%}
xtd(:,9:end)=(xtd(:,9:end)-
repmat(mnxttd,size(xtd,1),1))./repmat(mxxttd,size(xtd,1),1);
xtd=[ones(size(xtd,1),1) xtd (xtd(:,12)./(max(xtd(:,9),0.001))))];
xtsd=[ones(size(xtd,1),1) xtd (xtsd(:,12)./(max(xtd(:,9),0.001))))];
b=regress(ytd,xtd);
yhat=xtd*b;
e=abs(ytd-yhat);
disp(mean(e));
disp(median(e));
xtsd=awgn(xtd,60,'measured');
yhatl=xtsd*b;
e=abs(ytsd-yhatl);
disp(mean(e));
disp(median(e));
movienums=unique(movienum);
numofmovies=size(movienums,1);
movienums=[movienums (1:numofmovies)'];
movienumsds=dataset(movienums(:,1),movienums(:,2));
movieoutputds=dataset(movienum,'VarNames',{ 'Var1' });
moviedata=join(movieoutputds,movienumsds,'Var1');
moviegroups=moviedata.Var2;
testf=[moviegroups ytd yhat zeros(size(ytd))];
for i=1:size(ytd,1)
    if i==1 || testf(i,1)~=testf(i-1,1)

```

```

        testf(i,4)=1;
    else
        testf(i,4)=testf(i-1,4)+1;
    end
end
testf=[testf zeros(size(ytd)) zeros(size(ytd))];
for i=size(ytd,1):-1:1
    if i==size(ytd,1) || testf(i,1)~=testf(i+1,1)
        testf(i,5)=testf(i,4);
        testf(i,6)=10;
    else
        testf(i,5)=testf(i+1,5);
        for il=9:-1:0
            if testf(i,4)>=(il/10)*testf(i+1,5) &&
testf(i,4)<((il+1)/10)*testf(i+1,5)

                testf(i,6)=il+1;
            end
        end
    end
end
for i=1:size(testf,1)
    if i==1 || testf(i,6)<testf(i-1,6)
        testf(i,6)=1;
    end
end
testf2=testf(testf(:,6)==1 ,:);
meanresultmovie=accumarray(testf2(:,1),testf2(:,3),[],@mean);
meanresultmovie2=zeros(1,max(testf2(:,1)));
for i=1:max(testf2(:,1))
    if not(isempty(testf2(testf2(:,1)==i,1))) &&
size(testf2(testf2(:,1)==i,1),1)>2
        valus=testf2(testf2(:,1)==i,3);
        if size(unique(round(valus.*1000)),1)>2
            IDX = kmeans(valus,2);
            cluster=(sum(IDX==2)>sum(IDX==1))+1;
            meanresultmovie2(i)=mean(valus(IDX==cluster));
        else
            meanresultmovie2(i)=mean(testf2(testf2(:,1)==i,3));
        end
    elseif not(isempty(testf2(testf2(:,1)==i,1)))
        meanresultmovie2(i)=mean(testf2(testf2(:,1)==i,3));
    end
end
meanresultmovie2=meanresultmovie2';
meantargetmovie=accumarray(testf2(:,1),testf2(:,2),[],@mean);
disp(mean(abs(meanresultmovie-meantargetmovie)));
disp(median(abs(meanresultmovie-meantargetmovie)));
disp(mean(abs(meanresultmovie2-meantargetmovie)));
disp(median(abs(meanresultmovie2-meantargetmovie)));
movienumsts=unique(movienumsts);
numofmoviests=size(movienumsts,1);
movienumsts=[movienumsts (1:numofmoviests)'];
movienumstsd=dataset(movienumsts(:,1),movienumsts(:,2));
movieoutputts=dataset(movienumsts,'VarNames', {'Var1'});
moviedatats=join(movieoutputts,movienumstsd,'Var1');
moviegroupsts=moviedatats.Var2;
testfs=[moviegroupsts ytsd yhat1 zeros(size(ytsd))];
for i=1:size(ytsd,1)
    if i==1 || testfs(i,1)~=testfs(i-1,1)
        testfs(i,4)=1;
    end
end

```

```

        else
            testfs(i,4)=testfs(i-1,4)+1;
        end
    end
end
testfs=[testfs zeros(size(ytsd)) zeros(size(ytsd))];
for i=size(ytsd,1):-1:1
    if i==size(ytsd,1) || testfs(i,1)~=testfs(i+1,1)
        testfs(i,5)=testfs(i,4);
        testfs(i,6)=10;
    else
        testfs(i,5)=testfs(i+1,5);
        for il=9:-1:0
            if testfs(i,4)>=(il/10)*testfs(i+1,5) &&
testfs(i,4)<((il+1)/10)*testfs(i+1,5)
                testfs(i,6)=il+1;
            end
        end
    end
end
end
for i=1:size(testfs,1)
    if i==1 || testfs(i,6)<testfs(i-1,6)
        testfs(i,6)=1;
    end
end
testf2=testfs(testfs(:,6)==1 ,:);
meanresultmoviets=accumarray(testf2(:,1),testf2(:,3),[],@mean);
meanresultmoviets2=zeros(1,max(testf2(:,1)));
for i=1:max(testf2(:,1))
    if not(isempty(testf2(testf2(:,1)==i,1))) &&
size(testf2(testf2(:,1)==i,1),1)>2
        valus=testf2(testf2(:,1)==i,3);
        if size(unique(round(valus.*1000)),1)>2
            IDX = kmeans(valus,2);
            cluster=(sum(IDX==2)>sum(IDX==1))+1;
            meanresultmoviets2(i)=mean(valus(IDX==cluster));
        else
            meanresultmoviets2(i)=mean(testf2(testf2(:,1)==i,3));
        end
    elseif not(isempty(testf2(testf2(:,1)==i,1)))
        meanresultmoviets2(i)=mean(testf2(testf2(:,1)==i,3));
    end
end
meanresultmoviets2=meanresultmoviets2';
meantargetmoviets=accumarray(testf2(:,1),testf2(:,2),[],@mean);
disp(mean(abs(meanresultmoviets-meantargetmoviets)));
disp(median(abs(meanresultmoviets-meantargetmoviets)));
disp(mean(abs(meanresultmoviets2-meantargetmoviets)));
disp(median(abs(meanresultmoviets2-meantargetmoviets)));
[ymx,ycl]=max(y6cl,[],2);
[ymx,yscl]=max(ys6cl,[],2);
sse=sum((yhat-ytd).^2);
sst=sum((ytd-mean(ytd)).^2);
R2=1-(sse/sst);
sses=sum((yhat1-ytsd).^2);
ssts=sum((ytsd-mean(ytsd)).^2);
R2s=1-(sses/ssts);
ssem=sum((meanresultmovie-meantargetmovie).^2);
sstm=sum((meantargetmovie-mean(meantargetmovie)).^2);
R2m=1-(ssem/sstm);
ssemss=sum((meanresultmoviets-meantargetmoviets).^2);
sstmss=sum((meantargetmoviets-mean(meantargetmoviets)).^2);

```

```

R2ms=1-(ssems/sstms);
ssem2=sum((meanresultmovie2-meantargetmovie).^2);
R2m2=1-(ssem2/sstm);
ssems2=sum((meanresultmoviets2-meantargetmoviets).^2);
R2ms2=1-(ssems2/sstms);
disp('R2')
disp(R2);
disp(R2s);
disp(R2m);
disp(R2ms);
disp(R2m2);
disp(R2ms2);

```

Calculate movie data regression:

```

for i1=1:4
    if i1==1
        func=@mean;
    elseif i1==2
        func=@max;
    elseif i1==3
        func=@min;
    else
        func=@median;
    end
    xtdtf=xtdd;% (testf(:,6)==6,:);
    moviegroupstf=moviegroups;% (testf(:,6)==6,:);
    meanmovie1=accumarray(moviegroupstf,xtdtf(:,1),[],func);
    meanmovie2=accumarray(moviegroupstf,xtdtf(:,2),[],func);
    meanmovie3=accumarray(moviegroupstf,xtdtf(:,3),[],func);
    meanmovie4=accumarray(moviegroupstf,xtdtf(:,4),[],func);
    meanmovie5=accumarray(moviegroupstf,xtdtf(:,5),[],func);
    meanmovie6=accumarray(moviegroupstf,xtdtf(:,6),[],func);
    meanmovie7=accumarray(moviegroupstf,xtdtf(:,7),[],func);
    meanmovie8=accumarray(moviegroupstf,xtdtf(:,8),[],func);
    meanmovie9=accumarray(moviegroupstf,xtdtf(:,9),[],func);
    meanmovie10=accumarray(moviegroupstf,xtdtf(:,10),[],func);
    meanmovie11=accumarray(moviegroupstf,xtdtf(:,11),[],func);
    meanmovie12=accumarray(moviegroupstf,xtdtf(:,12),[],func);
    meanmovie13=accumarray(moviegroupstf,xtdtf(:,13),[],func);
    xtdmovie=[meanmovie1 meanmovie2 meanmovie3 meanmovie4 meanmovie5 meanmovie6
meanmovie7 meanmovie8 meanmovie9 meanmovie10 meanmovie11 meanmovie12 meanmovie13];
    ytdmovie=accumarray(moviegroups,ytd,[],@mean);
    for i=2:13
        for j=i:13
            xtdmovie=[xtdmovie xtdmovie(:,i).*xtdmovie(:,j)];
        end
    end
    xtsdtdf=xtsd;% (testfs(:,6)==6,:);
    moviegroupststf=moviegroupsts;% (testfs(:,6)==6,:);
    meanmovie1s=accumarray(moviegroupststf,xtsdtdf(:,1),[],func);
    meanmovie2s=accumarray(moviegroupststf,xtsdtdf(:,2),[],func);
    meanmovie3s=accumarray(moviegroupststf,xtsdtdf(:,3),[],func);
    meanmovie4s=accumarray(moviegroupststf,xtsdtdf(:,4),[],func);
    meanmovie5s=accumarray(moviegroupststf,xtsdtdf(:,5),[],func);
    meanmovie6s=accumarray(moviegroupststf,xtsdtdf(:,6),[],func);
    meanmovie7s=accumarray(moviegroupststf,xtsdtdf(:,7),[],func);
    meanmovie8s=accumarray(moviegroupststf,xtsdtdf(:,8),[],func);
    meanmovie9s=accumarray(moviegroupststf,xtsdtdf(:,9),[],func);
    meanmovie10s=accumarray(moviegroupststf,xtsdtdf(:,10),[],func);
    meanmovie11s=accumarray(moviegroupststf,xtsdtdf(:,11),[],func);

```

```

meanmovie12s=accumarray(moviegroupststf,xtsdtf(:,12),[],func);
meanmovie13s=accumarray(moviegroupststf,xtsdtf(:,13),[],func);
xtsdmovie=[meanmovie1s meanmovie2s meanmovie3s meanmovie4s meanmovie5s
meanmovie6s meanmovie7s meanmovie8s meanmovie9s meanmovie10s meanmovie11s
meanmovie12s meanmovie13s];
ytsdmovie=accumarray(moviegroupsts,ytsd,[],@mean);
for i=2:13
    for j=i:13
        xtsdmovie=[xtsdmovie xtsdmovie(:,i).*xtsdmovie(:,j)];
    end
end
y6clm=zeros(size(ytdmovie,1),6);
y6clm(ytdmovie<=2,1)=1;
y6clm(ytdmovie>2 & ytdmovie<=2.5,2)=1;
y6clm(ytdmovie>2.5 & ytdmovie<=3,3)=1;
y6clm(ytdmovie>3 & ytdmovie<=3.5,4)=1;
y6clm(ytdmovie>3.5 & ytdmovie<=4,5)=1;
y6clm(ytdmovie>4,6)=1;
ys6clm=zeros(size(ytsdmovie,1),6);
ys6clm(ytsdmovie<=2,1)=1;
ys6clm(ytsdmovie>2 & ytsdmovie<=2.5,2)=1;
ys6clm(ytsdmovie>2.5 & ytsdmovie<=3,3)=1;
ys6clm(ytsdmovie>3 & ytsdmovie<=3.5,4)=1;
ys6clm(ytsdmovie>3.5 & ytsdmovie<=4,5)=1;
ys6clm(ytsdmovie>4,6)=1;
b=regress(ytdmovie,xtsdmovie);
yhat=xtsdmovie*b;
e=abs(ytdmovie-yhat);
disp(mean(e));
disp(median(e));
yhat1=ytsdmovie*b;
e=abs(ytsdmovie-yhat1);
disp(mean(e));
disp(median(e));
y6clmr=zeros(size(yhat,1),6);
y6clmr(yhat<=2,1)=1;
y6clmr(yhat>2 & yhat<=2.5,2)=1;
y6clmr(yhat>2.5 & yhat<=3,3)=1;
y6clmr(yhat>3 & yhat<=3.5,4)=1;
y6clmr(yhat>3.5 & yhat<=4,5)=1;
y6clmr(yhat>4,6)=1;
ys6clmr=zeros(size(yhat1,1),6);
ys6clmr(yhat1<=2,1)=1;
ys6clmr(yhat1>2 & yhat1<=2.5,2)=1;
ys6clmr(yhat1>2.5 & yhat1<=3,3)=1;
ys6clmr(yhat1>3 & yhat1<=3.5,4)=1;
ys6clmr(yhat1>3.5 & yhat1<=4,5)=1;
ys6clmr(yhat1>4,6)=1;
ssem=sum((yhat-ytdmovie).^2);
sstm=sum((ytdmovie-mean(ytdmovie)).^2);
R2=1-(ssem/sstm);
ssem1=sum((yhat1-ytsdmovie).^2);
sstm1=sum((ytsdmovie-mean(ytsdmovie)).^2);
R2s=1-(ssem1/sstm1);
disp('R2');
disp(R2);
disp(R2s);
[ymx,ymcl]=max(y6clm,[],2);
[ymx,ymscl]=max(ys6clm,[],2);
[ymx,ymclr]=max(y6clmr,[],2);
[ymx,ymsclr]=max(ys6clmr,[],2);

```

```

cm=zeros(6);
for i=1:6
    for j=1:6
        cm(i,j)=sum((ymcl==i) & (ymclr==j));
    end
end
cms=zeros(6);
for i=1:6
    for j=1:6
        cms(i,j)=sum((ymscl==i) & (ymsclr==j));
    end
end
ey=eye(6);
ey1=ey+conv2(ey,[1 0 1],'same');
ym4cl=zeros(size(ytdmovie,1),4);
ym4cl(ytdmovie<=2,1)=1;
ym4cl(ytdmovie>2 & ytdmovie<=3,2)=1;
ym4cl(ytdmovie>3 & ytdmovie<=4,3)=1;
ym4cl(ytdmovie>4,3)=1;
yms4cl=zeros(size(ytsdmovie,1),4);
yms4cl(ytsdmovie<=2,1)=1;
yms4cl(ytsdmovie>2 & ytsdmovie<=3,2)=1;
yms4cl(ytsdmovie>3 & ytsdmovie<=4,3)=1;
yms4cl(ytsdmovie>4,4)=1;
ym4clr=zeros(size(yhat,1),4);
ym4clr(yhat<=2,1)=1;
ym4clr(yhat>2 & yhat<=3,2)=1;
ym4clr(yhat>3 & yhat<=4,3)=1;
ym4clr(yhat>4,4)=1;
yms4clr=zeros(size(yhat1,1),4);
yms4clr(yhat1<=2,1)=1;
yms4clr(yhat1>2 & yhat1<=3,2)=1;
yms4clr(yhat1>3 & yhat1<=4,3)=1;
yms4clr(yhat1>4,4)=1;
[ymx,ymcl4]=max(ym4cl,[],2);
[ymx,ymscl4]=max(yms4cl,[],2);
[ymx,ymclr4]=max(ym4clr,[],2);
[ymx,ymsclr4]=max(yms4clr,[],2);
cm4=zeros(4);
for i=1:4
    for j=1:4
        cm4(i,j)=sum((ymcl4==i) & (ymclr4==j));
    end
end
cms4=zeros(4);
for i=1:4
    for j=1:4
        cms4(i,j)=sum((ymscl4==i) & (ymsclr4==j));
    end
end
ey2=eye(4);
disp('classes');
disp(sum(sum(ey.*cm))/size(ymcl,1));
disp(sum(sum(ey1.*cm))/size(ymcl,1));
disp(sum(sum(ey2.*cm4))/size(ymcl4,1));
disp(sum(sum(ey.*cms))/size(ymscl,1));
disp(sum(sum(ey1.*cms))/size(ymscl,1));
disp(sum(sum(ey2.*cms4))/size(ymscl4,1));
disp('end');

end

```

Calculate movie classes and classes results:

```
ym6cl=zeros(size(meantargetmovie,1),6);
ym6cl(meantargetmovie<=2,1)=1;
ym6cl(meantargetmovie>2 & meantargetmovie<=2.5,2)=1;
ym6cl(meantargetmovie>2.5 & meantargetmovie<=3,3)=1;
ym6cl(meantargetmovie>3 & meantargetmovie<=3.5,4)=1;
ym6cl(meantargetmovie>3.5 & meantargetmovie<=4,5)=1;
ym6cl(meantargetmovie>4,6)=1;
yms6cl=zeros(size(meantargetmoviets,1),6);
yms6cl(meantargetmoviets<=2,1)=1;
yms6cl(meantargetmoviets>2 & meantargetmoviets<=2.5,2)=1;
yms6cl(meantargetmoviets>2.5 & meantargetmoviets<=3,3)=1;
yms6cl(meantargetmoviets>3 & meantargetmoviets<=3.5,4)=1;
yms6cl(meantargetmoviets>3.5 & meantargetmoviets<=4,5)=1;
yms6cl(meantargetmoviets>4,6)=1;
ym6clr=zeros(size(meanresultmovie2,1),6);
ym6clr(meanresultmovie2<=2,1)=1;
ym6clr(meanresultmovie2>2 & meanresultmovie2<=2.5,2)=1;
ym6clr(meanresultmovie2>2.5 & meanresultmovie2<=3,3)=1;
ym6clr(meanresultmovie2>3 & meanresultmovie2<=3.5,4)=1;
ym6clr(meanresultmovie2>3.5 & meanresultmovie2<=4,5)=1;
ym6clr(meanresultmovie2>4,6)=1;
yms6clr=zeros(size(meanresultmoviets2,1),6);
yms6clr(meanresultmoviets2<=2,1)=1;
yms6clr(meanresultmoviets2>2 & meanresultmoviets2<=2.5,2)=1;
yms6clr(meanresultmoviets2>2.5 & meanresultmoviets2<=3,3)=1;
yms6clr(meanresultmoviets2>3 & meanresultmoviets2<=3.5,4)=1;
yms6clr(meanresultmoviets2>3.5 & meanresultmoviets2<=4,5)=1;
yms6clr(meanresultmoviets2>4,6)=1;
[ymx,ymcl]=max(ym6cl,[],2);
[ymx,ymscl]=max(yms6cl,[],2);
[ymx,ymclr]=max(ym6clr,[],2);
[ymx,ymsclr]=max(yms6clr,[],2);
cm2=zeros(6);
for i=1:6
    for j=1:6
        cm2(i,j)=sum((ymcl==i) & (ymclr==j));
    end
end
cms2=zeros(6);
for i=1:6
    for j=1:6
        cms2(i,j)=sum((ymscl==i) & (ymsclr==j));
    end
end
ey=eye(6);
ey1=ey+conv2(ey,[1 0 1],'same');
ym4cl=zeros(size(meantargetmovie,1),4);
ym4cl(meantargetmovie<=2,1)=1;
ym4cl(meantargetmovie>2 & meantargetmovie<=3,2)=1;
ym4cl(meantargetmovie>3 & meantargetmovie<=4,3)=1;
ym4cl(meantargetmovie>4,3)=1;
yms4cl=zeros(size(meantargetmoviets,1),4);
yms4cl(meantargetmoviets<=2,1)=1;
yms4cl(meantargetmoviets>2 & meantargetmoviets<=3,2)=1;
yms4cl(meantargetmoviets>3 & meantargetmoviets<=4,3)=1;
yms4cl(meantargetmoviets>4,4)=1;
ym4clr=zeros(size(meanresultmovie2,1),4);
ym4clr(meanresultmovie2<=2,1)=1;
```

```

ym4clr(meanresultmovie2>2 & meanresultmovie2<=3,2)=1;
ym4clr(meanresultmovie2>3 & meanresultmovie2<=4,3)=1;
ym4clr(meanresultmovie2>4,4)=1;
yms4clr=zeros(size(meanresultmoviets2,1),4);
yms4clr(meanresultmoviets2<=2,1)=1;
yms4clr(meanresultmoviets2>2 & meanresultmoviets2<=3,2)=1;
yms4clr(meanresultmoviets2>3 & meanresultmoviets2<=4,3)=1;
yms4clr(meanresultmoviets2>4,4)=1;
[ymx,ymcl4]=max(ym4cl,[],2);
[ymx,ymscl4]=max(yms4cl,[],2);
[ymx,ymclr4]=max(ym4clr,[],2);
[ymx,ymsclr4]=max(yms4clr,[],2);
cm4=zeros(4);
for i=1:4
    for j=1:4
        cm4(i,j)=sum((ymcl4==i) & (ymclr4==j));
    end
end
cms4=zeros(4);
for i=1:4
    for j=1:4
        cms4(i,j)=sum((ymscl4==i) & (ymsclr4==j));
    end
end
ey2=eye(4);
disp(sum(sum(ey.*cm2))/size(ymcl,1));
disp(sum(sum(ey1.*cm2))/size(ymcl,1));
disp(sum(sum(ey2.*cm4))/size(ymcl4,1));
disp(sum(sum(ey.*cms2))/size(ymscl,1));
disp(sum(sum(ey1.*cms2))/size(ymscl,1));
disp(sum(sum(ey2.*cms4))/size(ymscl4,1));
ym6clr1=zeros(size(meanresultmovie,1),6);
ym6clr1(meanresultmovie<=2,1)=1;
ym6clr1(meanresultmovie>2 & meanresultmovie<=2.5,2)=1;
ym6clr1(meanresultmovie>2.5 & meanresultmovie<=3,3)=1;
ym6clr1(meanresultmovie>3 & meanresultmovie<=3.5,4)=1;
ym6clr1(meanresultmovie>3.5 & meanresultmovie<=4,5)=1;
ym6clr1(meanresultmovie>4,6)=1;
yms6clr1=zeros(size(meanresultmoviets,1),6);
yms6clr1(meanresultmoviets<=2,1)=1;
yms6clr1(meanresultmoviets>2 & meanresultmoviets<=2.5,2)=1;
yms6clr1(meanresultmoviets>2.5 & meanresultmoviets<=3,3)=1;
yms6clr1(meanresultmoviets>3 & meanresultmoviets<=3.5,4)=1;
yms6clr1(meanresultmoviets>3.5 & meanresultmoviets<=4,5)=1;
yms6clr1(meanresultmoviets>4,6)=1;
[ymx,ymclr1]=max(ym6clr1,[],2);
[ymx,ymsclr1]=max(yms6clr1,[],2);
cm=zeros(6);
for i=1:6
    for j=1:6
        cm(i,j)=sum((ymcl==i) & (ymclr1==j));
    end
end
cms=zeros(6);
for i=1:6
    for j=1:6
        cms(i,j)=sum((ymscl==i) & (ymsclr1==j));
    end
end
ym4clr1=zeros(size(meanresultmovie,1),4);
ym4clr1(meanresultmovie<=2,1)=1;

```

```

ym4clr1(meanresultmovie>2 & meanresultmovie<=3,2)=1;
ym4clr1(meanresultmovie>3 & meanresultmovie<=4,3)=1;
ym4clr1(meanresultmovie>4,4)=1;
yms4clr1=zeros(size(meanresultmoviets2,1),4);
yms4clr1(meanresultmoviets<=2,1)=1;
yms4clr1(meanresultmoviets>2 & meanresultmoviets<=3,2)=1;
yms4clr1(meanresultmoviets>3 & meanresultmoviets<=4,3)=1;
yms4clr1(meanresultmoviets>4,4)=1;
[ymx,ymclr41]=max(ym4clr1,[],2);
[ymx,ymsclr41]=max(yms4clr1,[],2);
cm4=zeros(4);
for i=1:4
    for j=1:4
        cm4(i,j)=sum((ymcl4==i) & (ymclr41==j));
    end
end
cms4=zeros(4);
for i=1:4
    for j=1:4
        cms4(i,j)=sum((ymscl4==i) & (ymsclr41==j));
    end
end
disp(sum(sum(ey.*cm))/size(ymcl,1));
disp(sum(sum(ey1.*cm))/size(ymcl,1));
disp(sum(sum(ey2.*cm4))/size(ymcl4,1));
disp(sum(sum(ey.*cms))/size(ymscl,1));
disp(sum(sum(ey1.*cms))/size(ymscl,1));
disp(sum(sum(ey2.*cms4))/size(ymscl4,1));

```