Segmentation algorithms for processing color images of Arctic Char

Elena Doronina

28.06. 2005

University of Joensuu

Department of Computer Science

Pro gradu

Abstract

Two kinds of segmentation technique were applied for the purpose of processing color images of fishes: object-oriented segmentation – for finding location and extraction of a fish from the image and color-oriented segmentation – for extraction red colored area within fish.

First one uses a priori information about image background color and is based on the global thresholding of the blue color component of initial *RGB* representation of the image, combined with edge detection and followed by morphological post-processing operations.

Second one involves transformation from *RGB* to *L*a*b* and then clustering by k-means in the chromaticity plane. Pre-processing includes intensity correction by calculating average intensity using blue patches from sufficiently big number of well lightened images and equalization in Value component of *HSV* if needed. Post-processing includes interpretation of the clustering results by back conversion to *RGB*.

This project was done in the collaboration with the Finnish Game and Fishery research center in Enonkoski, Finland.

Keywords: color image segmentation, thresholding, k-means clustering in CIELAB chromaticity plane.

Acknowledgements

I would like to sincerely thank people who really supported me during my studies and especially during my research work. At first, many thanks to my supervisors Jussi Parkkinen and Birgitta Martinkauppi for their guidance and help that could not be overestimated. Then, of course, I would like to express my gratitude to Jorma Piironen for giving me the opportunity to work on this interesting project. Next, I would like to thank people from Color research group of the University of Joensuu. I highly appreciated the possibility to be a member of your team even for such a short time.

Finally, I would thank my family and my friends who provided me with a confidence that I can cope with any problem and do everything I want.

CONTENTS

CHAPTER 1: INTRODUCTION

With the development of powerful computational resources, systems for digital image processing became popular, inexpensive and widely used tools in different applied fields of science and industry, such as medicine, forestry, architecture, satellite imagery processing etc. Wide range of algorithms has been developed to achieve particular goals depending on tasks and on the type of images provided.

The problem of segmentation arises in many tasks of image processing applications and usually plays a fundamental role as a low-level operation for further processing as far as it allows simplifying consequent analysis of segmented homogeneous regions, their geometrical and brightness characteristics [4],[22],[24]. Therefore the process of segmentation is often considered as a starting point for constructing formal description of the scene, on the quality of which the process of further recognition, interpretation of identification is dependent. On the other hand, it can be considered as an independent problem if we, for example, have to deal with the task of extracting regions of certain colors from image and measuring them.

Presented study aims to achieve next goals. Given with the set of color images of fish, the very first goal was to segment the body of the fish from the image and quantify it. The second objective was to classify colors in the fish's coloration and extract areas that are supposed to be red. The ambiguity in the formulation is conditioned by physical property of fish that is able to produce red color of various hues – from pink to deep red and orange.

More then three decades in the past an opinion that almost all image segmentation techniques are ad hoc in nature and no general algorithm can be adapted to work for all purposes on any kind of images has been stated [11]. Moreover, as a rule, techniques are usually dependent on parameters, constants and thresholds, which are usually fixed on the basis of few experiments. Despite a number of efforts for constructing more or less general segmentation algorithm for common need, still the most clearly specified, algorithmically efficient and robust are methods designed for the particular small applications assuming well-specified knowledge about the scene [38].

Due to these all prerequisites it seems to be a good point to start from analysis of the data that should be processed.

1.1 Formulation of the problem

The problem that is underneath described study is the project together with Finnish Game and Fisheries Research Institute (FGFRI) in Enonkoski, Finland [12]. Fisheries Research Unit of the Center produces scientific data, population estimates and expert services for the management of fisheries. The Aquaculture Unit maintains the genetic diversity of the endangered indigenous fish populations through aquaculture when other conservation methods cannot ensure this.

Among other research projects scientists from FGFRI are studying how they can restore the population of Arctic Char. These beautiful and mysterious fish are now rare and populations are disappearing at an alarming rate, because this species is very sensitive to environmental changes [17]. Acidification, eutrophication, engineering works (gravel abstraction, road building) and so on are those threats that affect the populations of this fish.

Figure 1: Image of Arctic Char [35]

In general Arctic Char (Figure 1) have an elongated body and an adipose fin, and notably have very small scales and an easily distinguished skull structure. Freshwater or land locked specimens are green-brown with reddish to white spots along the side and an orange to red belly [16], [31]. In many, but not all, populations males and in some cases females also, become more brightly colored at spawning time - often very deep pink to brownish red in color.

The former peculiarity led to the hypothesis that red color in fish coloration can be a factor that defines more or less the behavior of fish during mating period – its activity and ability to produce healthy posterity. Thus redness can be considered as a factor of successful individuals.

Consequently a need of quantitative measure of fish’s “redness” appeared. Thereto 135 color images of different fishes were taken. They were intended for digital image processing and analysis, during which necessary measure can be created.

One of the possibilities is to consider red area in the fish (homogeneous continuous region in the belly of the fish) in relation to the area occupied by its body or, in other words, as the amount of red pixels in respect to the amount of whole fish pixels.

Basically the problem consists of 3 parts:

1. Locate the fish in the image, quantify its area.
2. Find red colored area inside the fish, quantify it.
3. Find the relation of these two measures in order to have a quantitative description of the fish's redness.

1.2 What color images are all about

Phenomenon of color

Color is connected to the ability of objects to reflect the electromagnetic waves of different wavelengths. Chromatic spectrum spans the electromagnetic one in rather narrow range - from approximately 400 *nm*[1] to 700 *nm* (Figure 2).

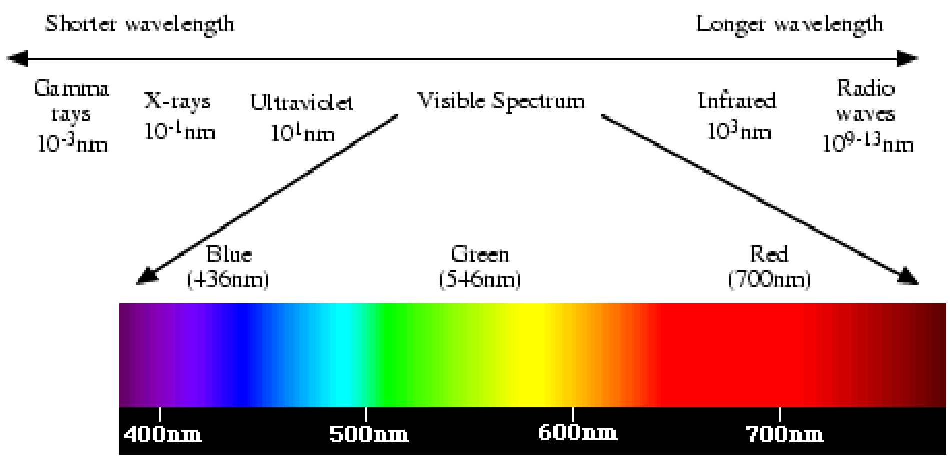


Figure 2: The visible spectrum in respect to electromagnetic spectrum (courtesy of [7])

Due to the construction of the human eye we are able to see the color as the combination of three primary colors: red, green and blue. This proposition is based on the classical model of human color vision postulated by Thomas Young (1802), according to which the human visual system acquires color imagery by means of three band pass filters S_X, X = R, G, B on light radiance $E(\lambda)$. In other words, these filters are basically three different kinds of photoreceptors situated in the retina and called cones, which are grouped by their sensitivity to the short, medium and long wavelengths. Their spectral responses are tuned to wavelengths of red, green and blue [38].

$$R = \int_\lambda E(\lambda) S_R(\lambda) d\lambda \quad G = \int_\lambda E(\lambda) S_G(\lambda) d\lambda \quad B = \int_\lambda E(\lambda) S_B(\lambda) d\lambda \qquad \textbf{[F. 1]}$$

1 *nm* – nanometer, equals to 10^{-9} m

In spite of the fact that photoreceptors in the eye respond actually to a wide range of wavelength (Figure 3), for the purpose of standardization they have been defined as 435.8 *nm* (blue), 546.1 *nm* (green) and 700 *nm* (red) [33]. Note that peak of red response curve is approximately 580 *nm*, but fixed value is 700 *nm*, as far as this peak value actually belongs to yellow zone.

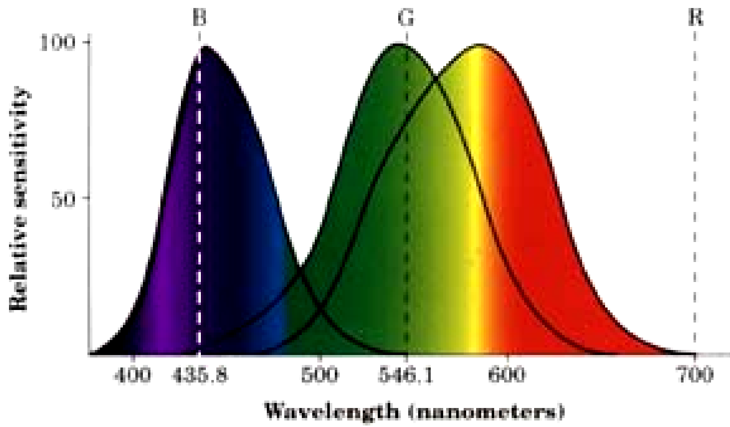


Figure 3: Normalized spectral response curves for each cone type (courtesy of [29])

In general, digital image is a set of points, which are named pixels. Each pixel in addition to information on its horizontal and vertical position can be characterized by the value of its brightness. Hence they can be represented as a 3d curve *f(x,y)*. (Figure 4)

While grayscale and black-and-white (binary) images are 2d arrays of integers, color image usually is a 2d array of (*R,G,B*) integer triplets. These triplets encode the intensity of each color (see Figure 12).

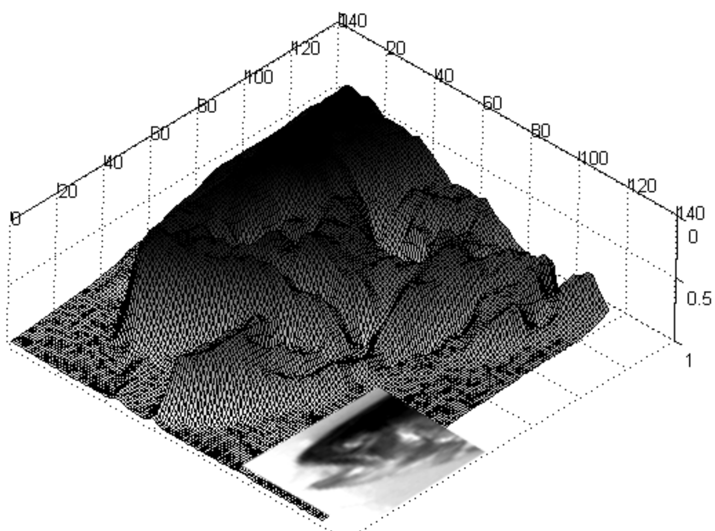


Figure 4: Unusual image representation of grayscale image as a 3d curve

1.3 Color spaces

For images acquired by digital cameras the most popular is *RGB* space, where colors are represented by their red, green and blue components in an orthogonal Cartesian space (Figure 5). This is in agreement with the tristimulus theory of color mentioned above.

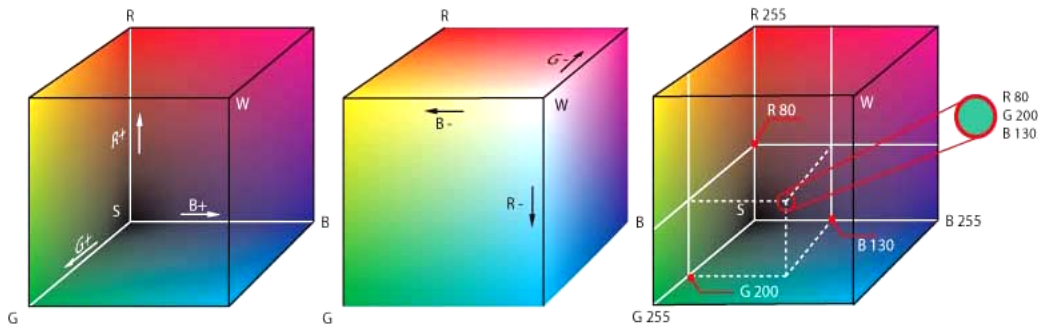


Figure 5: *RGB* color cube (courtesy of [42])

But the analysis of the pixel color distribution in a color space is not restricted to the (*R,G,B*) space. Actually, there are a large number of spaces that can be used to represent the color [33]. In [41] authors suggested categorizing the variety of them into 4 main groups (see gray rectangles in Figure 6), which contain subfamilies that are more specific in a way. In the chart they are outlined with dotted rectangles inside gray ones.

As it was already said, commonly color images are acquired through *R, G* and *B* components and hence (*R,G,B*) color space is defined. All the other color spaces can be obtained from it through linear or non-linear transformations. Arrows in the chart indicate ways of these transformations. Review on the most widely used transformations can be found in [4],[25]. Let us consider few the most popular color spaces and look what are their main features that can be utilized for segmentation.

One of desirable goals in segmentation is reducing of dependence of changing in lighting intensities [38]. If variations of intensities are uniform across the spectrum, then Normalized Color Coordinates can help to achieve this goal. Components of *NCC* are calculated as following:

$$x = \frac{X}{I} \quad x = r, g, b;\ X = R, G, B;\ I = R + G + B \qquad \textbf{[F. 2]}$$

Also $I = \sqrt{R^2 + G^2 + B^2}$ can be used for normalization [25]. Obvious shortcoming of *NCCrgb* color coordinates that they are very noisy if they are under low intensities due to nonlinearity of this transformation [4].

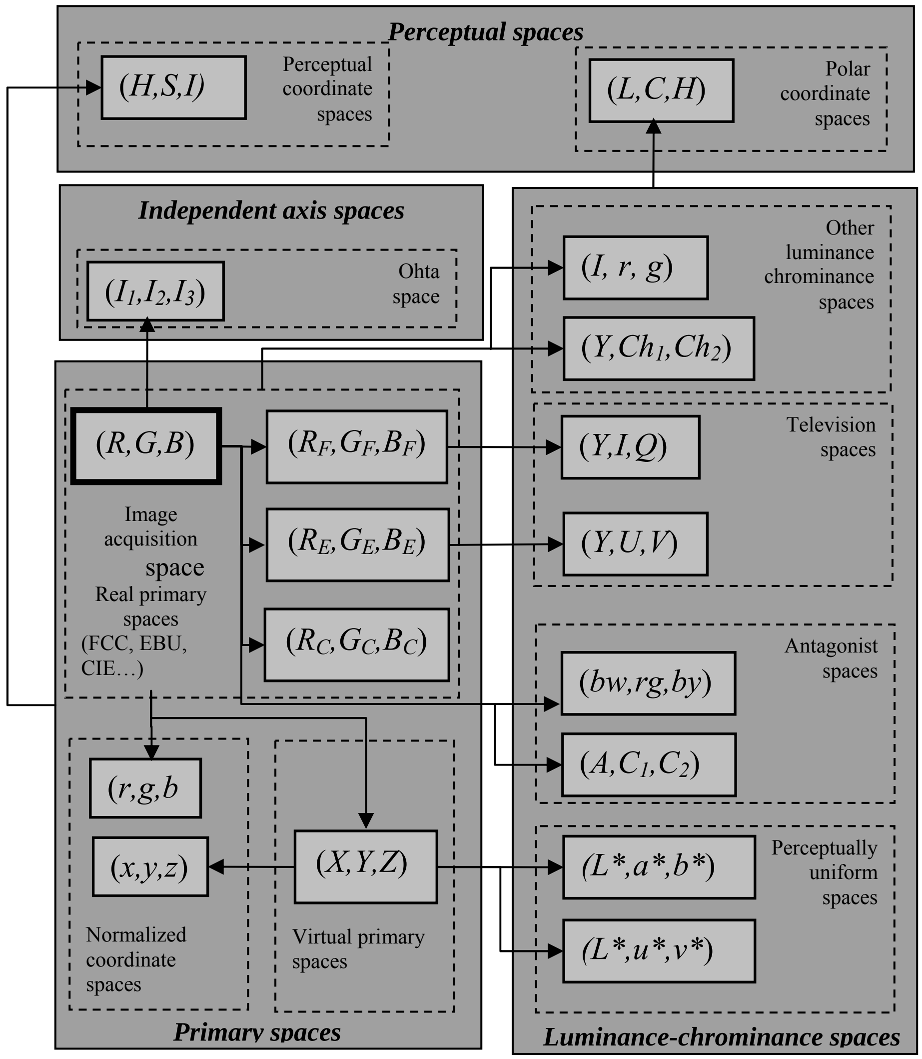


Figure 6: Four main color space families (adopted from [41])

Another space that aims to decrease the correlation between color components is Ohta [27] space, which is not color space in common sense, but presents color features and is basically an approximation of Karhunen - Loeve transformation for principal axes.

$$I_1 = \frac{R+G+B}{3}, I_2 = R-B, I_3 = \frac{2G-R-B}{2} \qquad \textbf{[F. 3]}$$

As we can see from previous example, not only particular colors can be taken as a base for the model of color representation, it can be also parameters that characterize the color. In general, any color can be gained by combining color from visible spectrum with white and black. For humans this way of thinking is more natural: we usually treat colors in terms of its hue, saturation and brightness. This is a basis for constructing color spaces that model a human color vision system, such as *HSI* [13]:

$$I = \frac{R+G+B}{3}, \quad S = 1 - \frac{\min(R,G,B)}{I}$$

$$H = \begin{cases} \arccos'\left(\dfrac{0.5((R-G)+(R-B))}{\sqrt{(R-G)^2+(R-B)(G-B)}}\right), & \text{if } S \neq 0 \\ \text{undefined otherwise} \end{cases}$$

[F. 4]

if $B>G$ then H=360+H, at last since H is an angle in degrees it should be normalized to 0..1: $H=H/360$.

Main advantage of this color space is that Intensity is separated from the color information. Hence this model can be useful in case if illumination level in image varies, because hue is invariant to certain types of highlights and shadows. Additionally, Hue and Saturation components are intimately related to the way in which human beings perceive the color. However the remaining disadvantage lies in non-removable singularity and numerically instability at low saturation, which are conditioned by non-linearity of the transformation [5],[32].

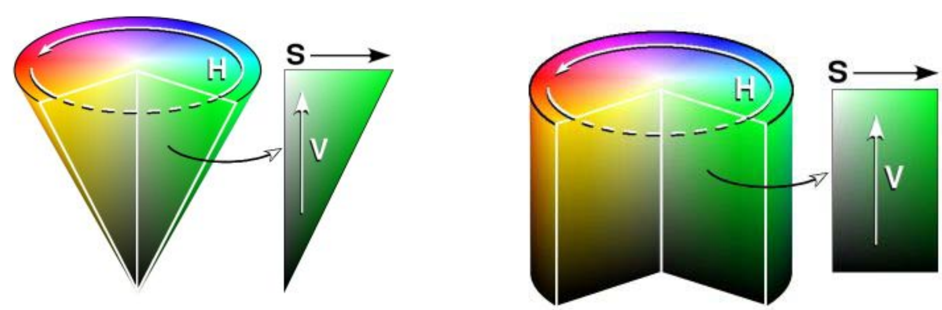


Figure 7: Two spatial representation of *HSV* color model (courtesy of [42])

There are a number of such intuitive color definition spaces that differs only with how components are calculated, but not in the essence. For example *HSB*: hue-saturation-brightness; *HSL*: hue-saturation-ligthness. Often in the model mentioned above Intensity component is substituted to Value, which is the maximum of (*R,G,B*) values of a pixel. This forms *HSV* (Hue, Saturation, Value) space [9], which spatial representation is presented on Figure 7.

$$V = \max(R,G,B), \quad S = \frac{\max(R,G,B) - \min(R,G,B)}{\max(R,G,B)}$$

if $S = 0$ then H undefined, otherwise $\Delta = \max(R,G,B) - \min(R,G,B)$ and

$$H = 60 * \begin{cases} \dfrac{G-B}{\Delta}, \ if \max(R,G,B) = R \\ 2 + \dfrac{B-R}{\Delta}, \ if \max(R,G,B) = G \\ 4 + \dfrac{R-G}{\Delta}, \ if \max(R,G,B) = B \end{cases} \qquad if \ H < 0 \ then \ H = H + 360$$

[F. 5]

The same idea of separation color information from brightness is used in *YUV* color system and in slightly different *YIQ*. Corresponding components are:

$$Y = 0.3R + 0.6G + 0.1B, \quad U = B - Y, \ V = R - Y,$$

[F. 6]

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.275 & -0.321 \\ 0.212 & -0.528 & 0.311 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \quad ,$$

[F. 7]

where *Y* represents luminance and can be considered as a grayscale version of the color image, while *U, V* and *I, Q* consist of the color information, i.e. chrominance. One more advantage of these color spaces is that they partly get rid of the correlation between color components, though it still exists, due to linear nature of the transformation from *RGB*.

Among well-known properties of *RGB* tristimulus coordinates are its dependence on physical sensors, exclusion of some visible colors and non-uniformity. In 1976 CIE (Committee International d'Eclairage) developed *XYZ* tristimulus coordinates, which are device independent and on base of which another useful coordinates can be calculated, for example *CIELAB* uniform color space (Figure 8). *XYZ* tristimulus coordinates can be produced from *RGB* by a linear transformation. However, the transformation matrix must be determined empirically. For instance, the matrix for the NTSC receiver primary system, which is based on C illuminant, is [21]:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.607 & 0.174 & 0.200 \\ 0.229 & 0.587 & 0.114 \\ 0 & 0.066 & 1.116 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

[F. 8]

If *XYZ* coordinates are calculated for the given illuminant, then *CIELAB* color components can be achieved as follows:

$$L^* = 116 f(Y/Y_n) - 16$$
$$a^* = 500[f(X/X_n) - f(Y/Y_n)] \qquad f(x) = \begin{cases} \sqrt[3]{x}, x > 0.008856 \\ 7.787x + 16/116, 0 < x \le 0.008856 \end{cases}$$
$$b^* = 200[f(Y/Y_n) - f(Z/Z_n)]$$

[F. 9]

where X_n, Y_n, Z_n are coordinates of the reference white point [34].

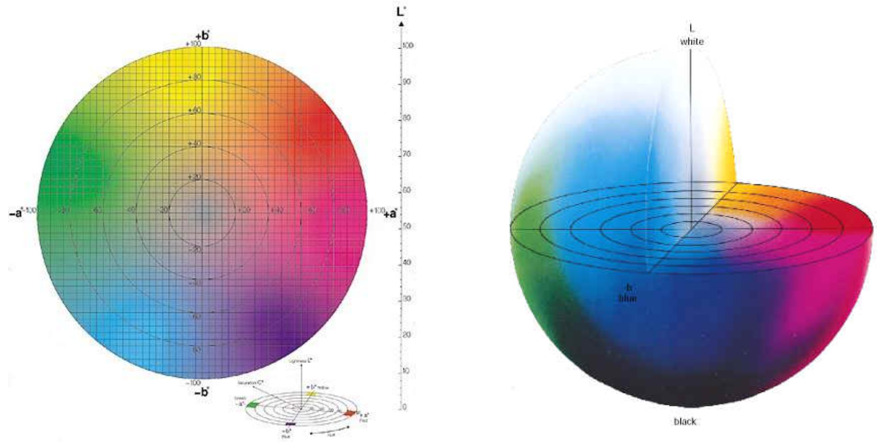


Figure 8: $L^*a^*b^*$ color space model: L^* - Luminance; a^* and b^* - chrominance; a^* ranges from green to red, b^* ranges from blue to yellow

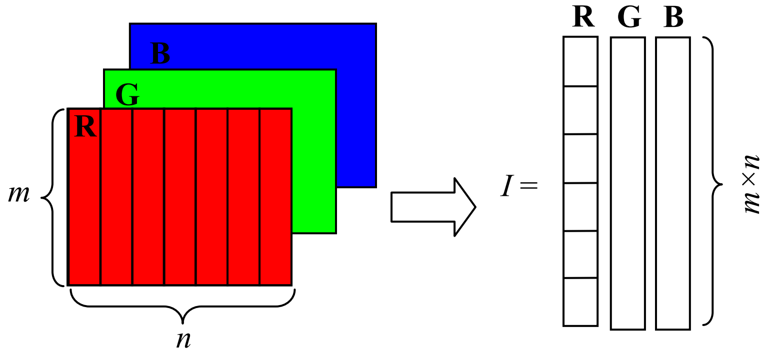
Among main advantages of this color space is its perceptual uniformity, which means that a small perturbation to a component value is approximately equally perceptible across the range of that value.

Although we have deal with color images, reducing a color image to gray level one is useful in many situations and can be accomplished in many ways in addition to obvious methods of selecting just one of the color component (*R, G, B, H, S, I* etc) or their combination. One of the possible ways that can give high contrast gray-scale images - which means a potentially good separability between structures in the image - is the projection of all color pixels onto the least-squared-fit line that provide the greatest separation of various pixel color values [36].

Use of principal component analysis (PCA) can also be utilized in order to calculate the principal axis to which all the points from color space will be projected [24],[34].

Algorithm for transforming color image to gray scale using PCA:

1. Represent color image ($m \times n \times 3$) in the form of 3 vectors, each of length ($m \times n$). Form a matrix *I*, where each column is one of the vectors. (see Figure below)



2. Calculate covariance matrix for matrix I (it'll be 3×3), its eigenvalues λ_i, i=1,2,3 and corresponding eigenvectors $e_i = (e_{i1}, e_{i2}, e_{i3})^T$. Note that eigenvalues should be sorted in descending order $\lambda_1 \geq \lambda_2 \geq \lambda_3$.

3. Determine axes of the new coordinate space v_i:

$$\begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} = \begin{bmatrix} e_{11} & e_{12} & e_{13} \\ e_{21} & e_{22} & e_{23} \\ e_{31} & e_{32} & e_{33} \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \qquad \textbf{[F. 10]}$$

Theory states that main information will be concentrated along the principal axis, which corresponds to the maximum eigenvalue of the covariance matrix, and thus we can take this projection as a gray scale version of color image with confidence that no considerable loss of the information will occur.

CHAPTER 2: IMAGE SEGMENTATION

2.1 Definitions and classifications

In order to be able to analyze the content of an image, one needs first to locate and isolate objects within it. This procedure is referred to as image segmentation.

Segmentation is a process of partitioning an image into disjoint and homogeneous regions. Formal definition in terms of image algebra is as follows [30]: Let I denotes an image and H stands for homogeneous predicate, then segmentation of I is a partition P of I into a set of N regions R_n, $P : I \to R_n$ n=1...N such as:

1. $\bigcup_{n=1}^{N} R_n = I,\ R_n \bigcap R_m = 0 \Leftrightarrow n \neq m$
2. $H(R_n) = true \quad \forall n$
3. $H(R_n \bigcap R_m) = false, \quad \forall\ R_n\ and\ R_m\ adjacent$

In [14] authors have proposed a kind of qualitative definition for good image segmentation: "*Regions of an image segmentation should be uniform and homogeneous with respect to some characteristics such as gray tone or texture (or color). Region interiors should be simple and without small holes. Adjacent regions of segmentation should have significantly different values with respect to the characteristic on which they are uniform. Boundaries of each segment should be simple, not ragged and must be spatially accurate*".

In spite of some proposed heuristic measures for quantitative evaluation of segmentation, unfortunately no one single measure can capture all the factors that affect the segmentation results: homogeneity, uniformity, spatial compactness, correspondence to visual perception etc [22].

There are many segmentation techniques available in the literature. Exhaustive surveys on recent algorithms can be found in [3],[4],[22],[38]. There are also a number of classifications of these algorithms. The most common divides them into 3 subparts: histogram-based (thresholding), edge-based and region-based methods [13],[39]. Another classification emerged in [22] is like this: feature space based techniques, image domain based techniques and physics-based techniques. More general classification can be represented as in the Figure 9.

In the chart below the most widely used approaches to the image segmentation are grouped into 4 main categories. This division is based on the type of feature that can be used for separation of different regions in the image.

We can either try to find homogeneous areas by agglomeration of "similar" pixels (methods in the first group) or treat regions as areas bounded by edge points (second group). Another approach is to work with pixels entirely, treating them as points in 1, 2 or 3 dimensional spaces with certain properties. These properties provide the instrument for classification pixels, combining them in different groups and hence distinguish objects (or colors, textures) in the image. Techniques in the last group try to make advantage of the capability of different materials to reflect or absorb the light.

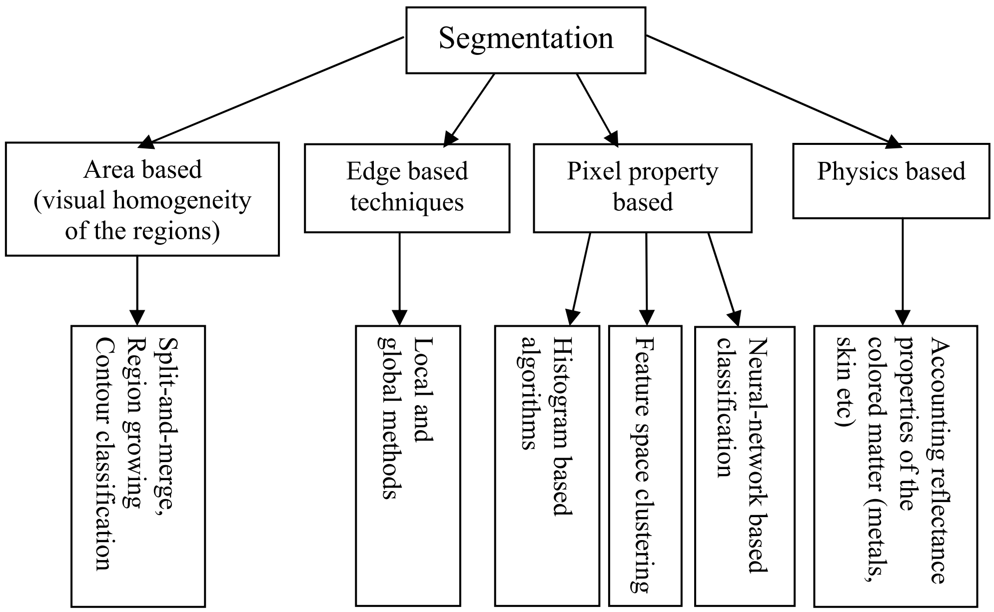


Figure 9: Classification of segmentation algorithms

It is important to understand [43] that there is no universally applicable segmentation technique that will work for all images and, on the contrary, most techniques are tailored for particular applications and might not work if certain condition will not be satisfied.

One more aspect, which is not of the last importance, is the completeness of the desirable segmentation [39]. In partial segmentation the image is divided into the set of separate homogeneous regions with respect to chosen property, i.e. brightness, color, texture, etc. To achieve a complete segmentation in which result regions correspond directly to the objects that should be separated, additional techniques, which use specific knowledge about the image, should be used. Of course complete segmentation can be achieved by further processing results of partial one.

2.2 Gray scale and color image segmentation techniques

Until a few years ago image segmentation techniques were usually proposed for gray scale images. But as far as the problem of "computational costs" is not crucial nowadays, a remarkable growth of algorithms for color images appeared. For the most part they are kind of dimensional extensions of those devised for gray scale images, although there are a number of methods that work directly with chromaticity, i.e. color characteristics.

This section gives a brief introduction to the most popular techniques. To preserve the logic of evolved approaches the presentation starts from techniques for gray scale images, and then followed by analogous (or advanced) for color images.

As it was mentioned above, color images can be reduced to gray scale. This reduction along with the seeming loss of the information can give an advantage for segmentation, as far as most information is usually concluded in the intensity component, which is suitable for detection of objects using fast, rather simple and the oldest, but still widely used segmentation method for gray scale images - histogram thresholding.

Histogram thresholding

Histogram is a vector that contains the information about the distribution of pixels in respect to brightness levels occurred in the image and usually is displayed as a bar graph.

Histogram thresholding is essentially a pixel classification problem in which the main objective is to separate the pixels of a given image into two classes, namely foreground (or object) from background [43].

The technique is based upon a simple concept. Those pixels, which values are below given brightness threshold θ are supposed to belong to object, while others are labeled as background. Here we assume that dark object is situated on a light background, though there is no strict demand about how to treat the parties. Let us denote the gray scale image as I. Then the procedure can be written as:

$$\text{If } I[n, m] < \theta \text{ then } I[n, m] = 1, \text{ else } I[n, m] = 0.$$

In principle, the test condition can be based upon some other property than just pixel brightness, for example redness, but the idea stays the same.

Thresholding can either be bi-level, as stated above, or multilevel, when multiple regions are detected through several threshold values. The main question of thresholding algorithm is how to determine the threshold value.

There are a variety of alternatives, however the selection of appropriate one for the given problem can be a difficult task [1],[43].

Two essential ways to determine the threshold are manual and automatic. Manual thresholding often provides good results, because the choice of threshold value is usually conducted by visual perception of the effect it makes and hence it can be tuned to provide better results. Automatic determination of the meaningful threshold is more desirable goal. An exhaustive survey on existing thresholding algorithms with automatic determination of the threshold value is presented in [37]. Among them there are methods that provide optimal threshold. The most referencing thresholding algorithm proposed by Otsu [28], which is based on minimizing weighted sum of within-class variances of the foreground and background pixels, is exactly such a method.

A common problem with the histogram-based techniques is the essential property of histograms that usually have many local minima and maxima; they often ragged due to noise that can produce spurious peaks and ambiguity in the segmentation results. These algorithms can benefit from smoothing the raw data of histogram to remove small fluctuations. But it is important that smoothing technique should not shift the peaks positions [39],[43]. Such smoothing techniques, for example, are:

Local averaging of neighboring histogram elements

$$h'(x) = \frac{1}{2k+1}\sum_{i=-k}^{k} h(x+i) \qquad \textbf{[F. 11]}$$

where k is usually odd and defines the size of the window, inside which the values will be averaged (k =3,5…).

One dimensional Gaussian blurring (convolution with the Gaussian function)

$$\sigma(x) = \frac{e^{-\frac{(x-\mu)^2}{2\sigma^2}}}{\sigma\sqrt{2\pi}}, \quad x \in [-k, k] \qquad \textbf{[F. 12]}$$

where σ is the parameter that defines the curvature of the smoothing function, μ is the shift value. The size of window k also should be chosen properly. The bigger the size, the more smooth effect can be reached.

If objects in the image do not touch each other and their gray levels are clearly distinct from background, thresholding is a suitable segmentation method. The global thresholding is used to isolate objects of interest that have values rather different from background, but it can give suitable results rather rarely. Local (or adaptive) thresholding has advantages in the situation when the illumination is non-uniform. This approach is sufficiently simple – divide the image into blocks and calculate thresholds for them independently. As it usually happens in divide-and-conquer approaches, the process of assembling processed parts can be complicated, as far as one has to take care about blocks that conclude only object parts, and hence should not be segmented (so called control of the histogram range is usually used for solving this task). The same situation with blocks that contain only background – suitable values for these regions should be calculated with the help of neighboring areas (interpolation).

Feature space clustering

Another popular approach is pixel clustering, through which image segmentation can be effectively performed. Cluster analysis allows partitioning data into meaningful subgroups and it can be applied for image segmentation and classification purposes. Clustering of characteristic features applied to image segmentation is the multidimensional extension of the concept of thresholding [11].

The use of clustering is based on the assumption that each region in the image forms a separate cluster in the feature space. For gray scale images typical features are intensity level, mean, standard deviation, entropy, sharpness etc, while for color images the most essential features are color components. Thus applying clustering for color image segmentation is a straightforward approach, as far as colors supposed to form clusters in the color/feature space. Features are image dependent and how to select them in order to obtain satisfactory segmentation remains unclear [4].

Two main questions that should be solved for clustering task are: how many clusters are the best for the given data and how to determine the validity of clusters. One more aspect about clustering is that it either requires providing the seeds for the regions to be segmented or uses non-parametric methods for finding the salient regions without the need for seed point. Often the number of clusters should be specified in advance, although there are techniques that do not require this step, as for example in [23], where the algorithm starts with the only one seed, which is in fact is a baricenter of the data and the number of clusters changes during the process. The opponent approach is to start with a large number of clusters in order to reduce the sensitivity to the initialization step and then, during the agglomeration step, merge them on competitive base [10]. This is the example of hierarchical clustering.

In non-hierarchical clustering at the initial stage an arbitrary number and actual positions of clusters should be chosen. The members belonging to each cluster will be checked by selected parameters or distance and relocated into the more appropriate clusters with higher separability. Examples of such methods are ISODATA (Iterative Self-Organizing Data Analysis Technique) and k-means reported by McQueen as early as in 1967.

The main framework of non-hierarchical clustering algorithm is usually as follows [18]:

(1) Initialization of cluster centers
(2) Partitioning: All data points are relocated into the closest clusters by computing the distance between them and the cluster centers.
(3) Repositioning of cluster centers: The center of gravity for each cluster is recalculated and the procedure above is repeated until convergence.

Convergence usually means that cost function – sum of all point-to-centroid distances – does not change much after completed iteration. Also there is another stopping criterion: maximum number of iterations allowed.

Mentioned algorithm, if it starts from randomly generated cluster's centers, does not use any a priori knowledge. On the contrary, such knowledge, if we do know some information about the data in process, can help to solve the problem of initialization and thus affect the result of clustering considerably.

Region based approaches

Within this approach the following definition of region is used: Region is a maximal connected set of pixels, for which uniformity condition is satisfied [38].

Homogeneous regions can be obtained for example through region growing process, which starts from a pre-selected point, called seed, and progressively performs agglomeration of similar neighbors. The term "similar" means that they satisfy chosen homogeneity criterion. The process of growth stops when no more points can be added to the region. As one can see this approach is oriented to production single region inside which specified demand of homogeneity is holds, nevertheless a goal of segmentation can be achieved by repeating growing process with different seeds and combining results.

Region growing procedure, described above, usually needs post-processing phase that includes merging of small or meaningless regions with the bigger ones that have similar attributes.

Disadvantages are: dependence of the results on the choice of seed point, on the homogeneity criterion and on the order in which image points are processed. The main advantage is that regions obtained by this kind of techniques are certainly spatially connected and rather compact.

Another way for gaining homogeneous regions is split-and-merge approach. The main idea of this strategy is to construct a hierarchical partition of the image into the blocks inside which a certain homogeneity criterion holds. Usually each block is divided into by 4 square sub-blocks if it does not satisfy homogeneity criterion. In other words the algorithm starts with the wittingly inhomogeneous region, aiming to find smaller but resemble ones. The idea is that inside rather small regions pixel properties are very similar. Thus after splitting phase there usually exist a lot of fragments that should be agglomerated to obtain meaningful results. For this purpose the merging phase is usually performed. The goal is reached by associating neighboring regions and guaranteeing that homogeneity requirements are met until maximal possible uniform segments can be created [22].

Main disadvantage of this method is that results can be too ragged (boundaries are not smooth) due to data structures used in division part – quad tree.

Homogeneous criteria for gray scale and color images can vary. For monochrome images it can be for example threshold of the intensity histogram. For color images such criterion can be color similarity within the block or for example spatial proximity [40].

Edge detection

The notion of region can be equivalently defined as connected set of pixels surrounded by closed boundary. Hence the problem of finding homogeneous regions can be reduced to the search of closed boundary that consists of edge points. But in reality closed boundary are difficult to discover. In [5] authors emphasize the fact that image segmentation can not be accomplished by using only edge detection techniques, which nevertheless provide useful information about region boundaries that can be utilized in high level systems or can be combined with other approaches.

For gray scale images rather many studies on this topic have been accomplished [11],[30]. Mainstream is to use scalar functions for gradient approximation – Sobel, Prewitt, Roberts (first difference operators) or Laplacian (second difference operator) [13]. Edge detection in color images is usually done by considering them as a set of separate grayscale images corresponding to color planes. Edges are detected in each of them and then combined. Instead 3d space analysis the analysis of 3 projections is used. Example of such approach is described in [4].

In the edge detection there are also local and global approaches. Local technique needs only information of the neighborhood of the point in consideration, while global techniques make a sort of global optimization involving changes in large areas. This can be done for example by using different approaches to Markov Random fields.

Edge detecting is very essential way in which humans distinguish objects. It gives rather satisfactory results for images with good contrast between regions and oppositely does not work well with images where edges are ill-defined or there are too many of them; moreover edge detectors are often very sensitive to noise. One more challenge is how to get closed boundary that described certain object.

Neural network based methods and other

Other approaches such as segmentation using Markov Random Fields, variational approach based on minimization Mumford-Shah functional and treating images as 3d curves, active contours – so called “color snakes” and different applications of neural networks can be found in the literature [2],[20],[26] and were not considered here since they had not been applied for the current problem anyhow.

CHAPTER 3: TASK TAILORED SEGMENTATION

3.1 Stages of segmentation

The process of segmentation is usually carried out in three steps: at first we need to pre-process the image; then the actual segmentation phase is usually followed by post-processing procedures (Figure 10).

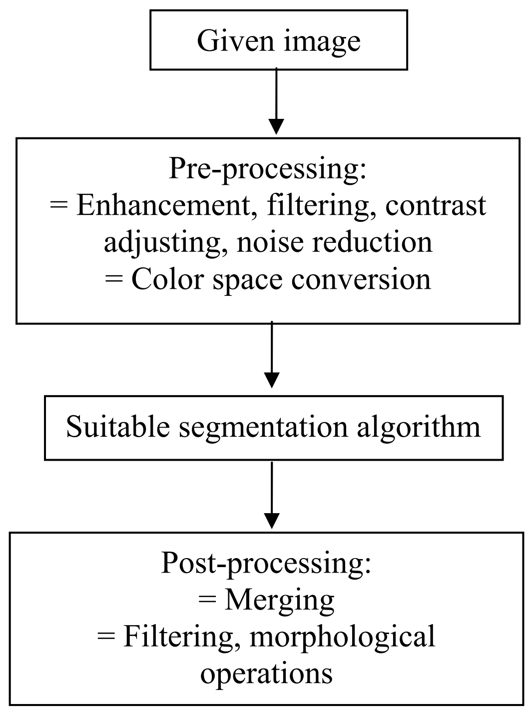


Figure 10: Stages of image segmentation

As it was noted in the very beginning, the purpose of this study is to produce quantitative measure for red color in the fish coloration. As far as data set is not uniform, the algorithms for processing them should be able to cope with such challenges as dark images, presence of additional objects, non-uniform illumination. The aim of pre-processing is to prepare images for applying algorithms of segmentation, while post-processing tries to improve results obtained after main stage of segmentation. In the following sections some useful pre- and post-processing techniques will be considered and illustrated on the given data for both tasks of the segmentation: segmentation of the entire fish (object-oriented) and segmentation of the red color.

3.2 Pre-processing

Applying color to gray scale transformation

The choice of space for representation color images is usually conditioned by the objective of image processing. It has been noticed many times that the performance of image segmentation procedure depends highly on the choice of the working color space [26],[39]. And often the conclusions are very contradictory. While some authors stated that conventional spaces could be rather suitable, others tried to construct artificial or hybrid ones [22],[38],[41].

Trying not to overcomplicate the issue and bearing in mind first part of global task – extraction of the entire fish, we decided to start from histogram thresholding as segmentation technique. And thus the need of gray scale version of color image is obvious. Moreover, for this purpose we need to have image, which histogram will be bimodal or at least close to it.

As it was mentioned above, the very essential way to get the gray scale image from color one is to take its Intensity, although it can be anyone of the components of any color space or combination of them. At first, let us take a look at given *RGB* image in detail. Below (Figure 11) one can see typical image under consideration and corresponding histograms of color components. As we look at each color sheet separately (Figure 12) we see that there are two clearly distinguishable peaks in blue one, which correspond to the dark object and sufficiently smooth background (middle-gray values). Also there are peaks that are responsible for the bright spots in the image – white pieces of paper and the stripe of reflected light from background (leftist side of the image). Nevertheless, it looks promising and suitable for histogram thresholding.

Figure 11: Typical "good" *RGB* image from fish images collection

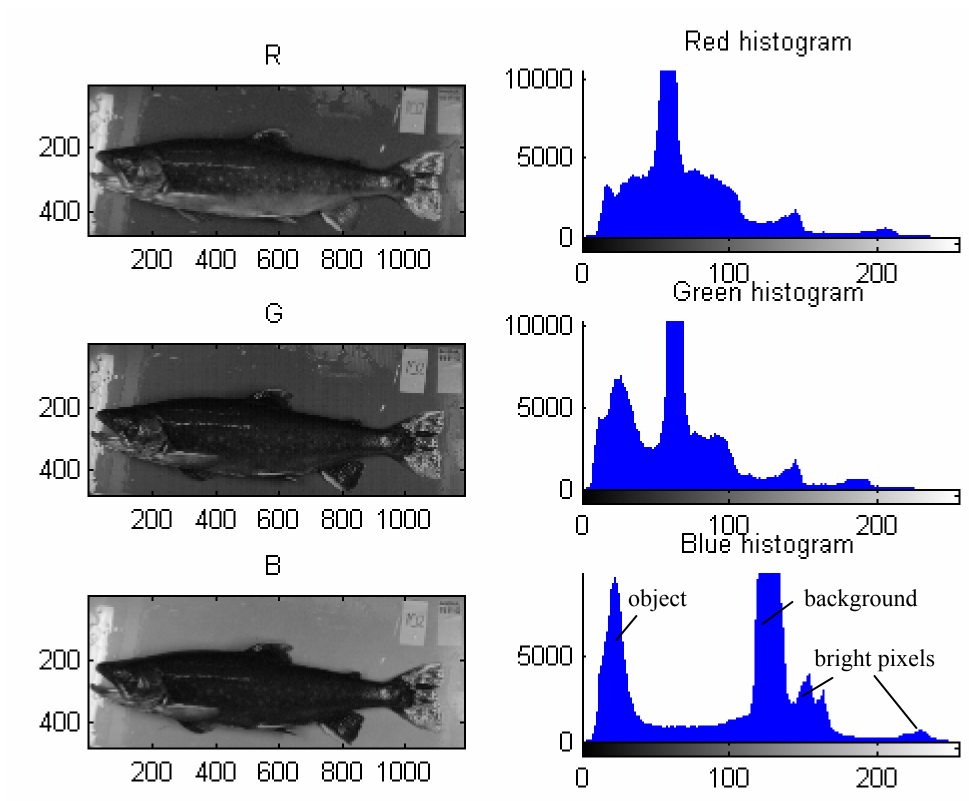


Figure 12: Components of *RGB* image with corresponding histograms

Running a few steps forward, one should say that investigation of other color components of different color spaces mentioned in the section II did not provide more promising representation for histogram thresholding (see Appendix I for some other color space components and their histograms). Thus, among separate components of color spaces the best candidate is blue color component. Indeed, as we know, all images were taken under the same condition – the background on which fishes were shot was of blue color, and consequently there is nothing strange that namely this color component turned out to be able to provide good results for segmentation by this technique – the choice of it is conditioned by a priori knowledge about images in consideration.

In the beginning we have mentioned also other ways for getting gray scale images from color one. We compared histogram of blue channel of *RGB* to widely used Intensity and projection of color image to the principal axis. Results are presented on the figure below (Figure 13). Here again blue component is the most suitable.

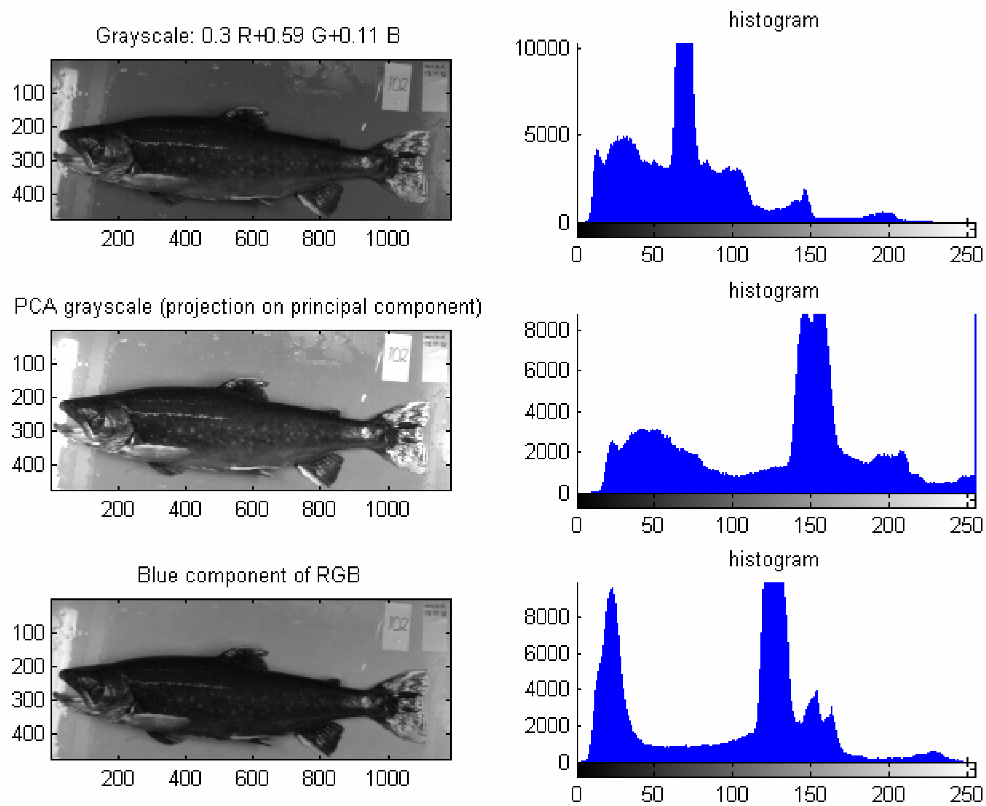


Figure 13: Comparison of *RGB* to gray scale conversions

Color image enhancement

Well-known technique for increasing contrast in a dark image is histogram equalization [13]. It has been developed for gray scale images. Now, when we have deal with *RGB* color images the obvious generalization idea is to equalize each color independently, but actually this will result in somewhat different colors in transformed image. In general it is better to apply the transformation only to the Intensity component of an *HSI* image or to the Luminance component of a *YIQ* image, thus leaving the chromaticity unaltered.

As we can see from the pictures below (Figure 14), equalization in Value component enhanced visual quality of the dark image. Now red color is more distinguishable, while equalization in the *R, G* and *B* separately has changed colors considerably.

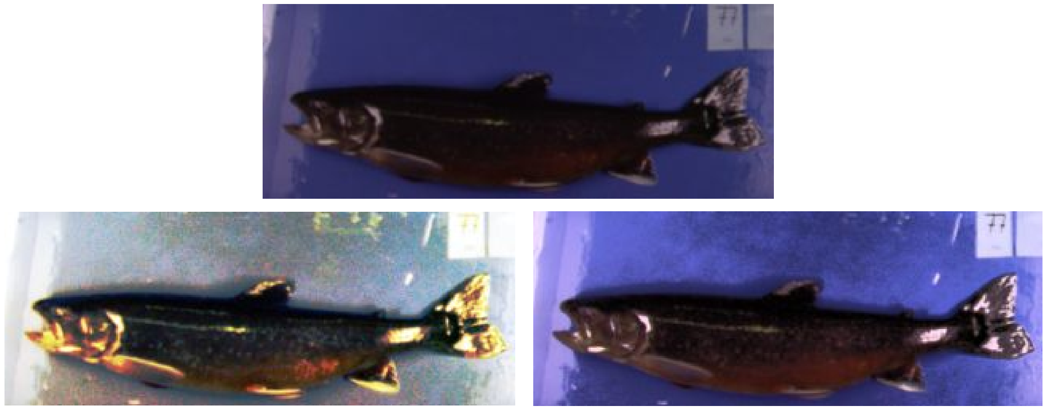


Figure 14: Example of dark image (above) equalized in each of *RGB* color component (left bottom) and in Value of *HSV* only (right bottom)

RGB to CIELAB conversion

In the task of color image segmentation information about chromaticity of the colors can be utilized for clustering. In order to have chromaticity information separated from luminance, conversion from *RGB* to *CIELAB* color space can be performed.

There are some methods for doing this. Normally *CIELAB* (or *L*a*b**) can be determined from *XYZ* color coordinates through well known formulae (see [F. 9]). But then another question arises: how to get those *XYZ*, which are usually defined from spectral reflectance of object and illuminant ([F. 13]), having in disposal only *RGB* coordinates?

$$X = \frac{1}{k}\sum_{\lambda=380}^{780} R(\lambda)E(\lambda)\bar{x}(\lambda) \quad Y = \frac{1}{k}\sum_{\lambda=380}^{780} R(\lambda)E(\lambda)\bar{y}(\lambda) \quad Z = \frac{1}{k}\sum_{\lambda=380}^{780} R(\lambda)E(\lambda)\bar{z}(\lambda)$$

[F. 13]

where *R(λ)* – reflectance spectra, $\bar{x}, \bar{y}, \bar{z}$ – standard observer functions (Figure 16), *E(λ)* – spectra power distribution of the illuminant (Figure 17), and $k = \sum_{\lambda=380}^{780} E(\lambda)\bar{y}(\lambda)$ is normalizing coefficient.

One of the possible solutions can be to use a Macbeth chart (below) with known spectral reflectance, which is shot with the same camera and under the same illumination; power spectra distribution of used illuminant should also be measured. *RGB* image and spectral reflectance of the Macbeth chart are presented below. There are 24 patches in the chart, and there is corresponding number of the curves in the reflectance graph.

Figure 15: *RGB* image of Macbeth chart taken with the Minolta digital camera and 2 daylight lamps that were used also for making images of Arctic Char in 2002

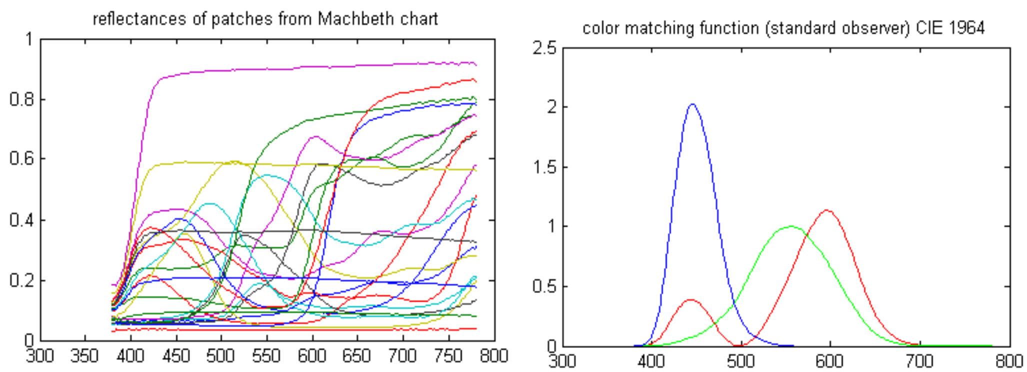


Figure 16: Reflectance spectra of the patches from Macbeth chart (left) and standard observer functions: blue – x, green – y, red – z (right)

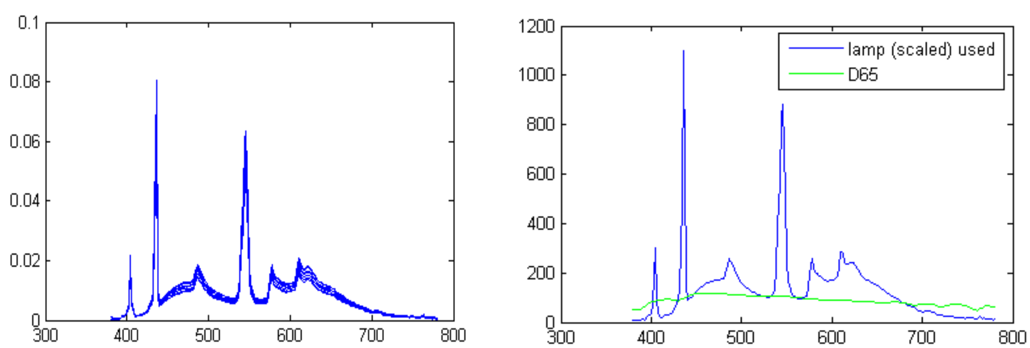


Figure 17: Lamp power spectrum distribution taken in different positions of the shooting area (left); Comparison of scaled spectra (at 560 *nm* having value 100)

Consequently, we can calculate *XYZ* values for each patch (Table 1) using its reflectance, illuminant and standard observer function as it is written in formulas ([F. 13]). On the other hand we have *RGB* values of each patch, as far as chart was also shot with the digital camera (Table 2). Values here are actually averaged – in order to decrease possible noise. In other words we have at the disposal, say "correct" tristimulus *XYZ* values for known *RGB* values. And thus we can try to find transformation between *RGB* and *XYZ*.

Color name	patch#	X	Y	Z
dark skin	**1**	0,1129	0,0991	0,0539
lightskin	**2**	0,387	0,3538	0,2044
bluesky	**3**	0,1786	0,1983	0,2704
foliage	**4**	0,1174	0,1392	0,057
blue flower	**5**	0,2511	0,2476	0,352
bluish green	**6**	0,3431	0,4479	0,3433
orange	**7**	0,3621	0,2753	0,0496
purplish blue	**8**	0,133	0,1329	0,3061
moderate red	**9**	0,2746	0,1917	0,1161
purple	**10**	0,0839	0,0678	0,1304
yellow green	**11**	0,3747	0,4387	0,0816
orange yellow	**12**	0,4702	0,4054	0,0636
blue	**13**	0,0774	0,0707	0,241
green	**14**	0,1721	0,2403	0,0693
red	**15**	0,1846	0,1125	0,0429
yellow	**16**	0,5896	0,579	0,0683
magenta	**17**	0,2746	0,1937	0,2616
cyan	**18**	0,1472	0,2086	0,2859
white	**19**	0,8677	0,8987	0,7343
neutral 8	**20**	0,5612	0,5822	0,492
neutral 6.5	**21**	0,3491	0,3604	0,3034
neutral 5	**22**	0,1973	0,2054	0,1715
neutral 3.5	**23**	0,089	0,0926	0,077
black	**24**	0,034	0,0351	0,0303

Table 1: *XYZ* values for patches from Macbeth chart

115.6755 67.6130 80.6856	217.6667 148.9421 176.1582	116.8918 100.0957 187.3856	88.5684 88.0733 79.0028	157.1885 114.9607 213.5917	152.4699 173.0240 227.0757
228.5352 138.3807 99.8082	99.3796 75.7189 204.7832	215.0909 91.7467 133.5453	93.8405 44.0286 119.7472	172.5376 181.8751 122.0645	235.5507 170.4694 106.2260
72.3992 45.4891 189.5900	98.9107 135.8516 112.8553	203.9787 55.3899 85.8495	249.2122 208.4589 122.4233	211.7871 88.8973 185.9843	95.8005 107.4207 209.0301
250.7005 248.0655 248.7028	236.0045 203.7403 248.6855	179.8271 154.6683 210.9008	131.6993 110.0281 154.6073	81.7104 63.4250 96.5897	40.1299 27.0262 50.3556

Table 2: Averaged *RGB* values for patches form Macbeth chart

In [19] a series of polynomials ranging from tree-term linear combination to a twenty-term cubic equation were used for this purpose. The transfer matrices from *RGB* to CIE space are obtained by employing multiple polynomial regressions.

Equations for calculating transform matrices that employs different number of members are as follows:

1. Linear terms (3x3)

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \quad \textbf{[F. 14]}$$

(3x24) (3x3) (3x24)

2. Linear and cross product terms (3x6)

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} & a_{16} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} & a_{26} \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} & a_{36} \end{pmatrix} \begin{bmatrix} R \\ G \\ B \\ RG \\ RB \\ GB \end{bmatrix} \quad \textbf{[F. 15]}$$

3. Black and white terms added to linear and cross product terms matrix (3x8)

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{pmatrix} a_{10} & a_{11} & a_{12} & a_{13} & a_{14} & a_{15} & a_{16} & a_{110} \\ a_{20} & a_{21} & a_{22} & a_{23} & a_{24} & a_{25} & a_{26} & a_{210} \\ a_{30} & a_{31} & a_{32} & a_{33} & a_{34} & a_{35} & a_{36} & a_{310} \end{pmatrix} \begin{bmatrix} 1 \\ R \\ G \\ B \\ RG \\ RB \\ GB \\ RGB \end{bmatrix} \quad \textbf{[F. 16]}$$

4. Added squared terms to linear and cross product terms matrix (3x9)

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} & a_{16} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} & a_{26} \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} & a_{36} \end{pmatrix} \begin{bmatrix} R \\ G \\ B \\ RG \\ RB \\ GB \end{bmatrix} + \begin{pmatrix} a_{17} & a_{18} & a_{19} \\ a_{27} & a_{28} & a_{29} \\ a_{37} & a_{38} & a_{39} \end{pmatrix} \begin{bmatrix} R^2 \\ G^2 \\ B^2 \end{bmatrix} \quad \textbf{[F. 17]}$$

5. 3x11 matrix: 3x8 U 3x9

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} a_{10} \\ a_{20} \\ a_{30} \end{bmatrix} + \begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} & a_{16} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} & a_{26} \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} & a_{36} \end{pmatrix} \begin{bmatrix} R \\ G \\ B \\ RG \\ RB \\ GB \end{bmatrix} +$$

$$+ \begin{pmatrix} a_{17} & a_{18} & a_{19} \\ a_{27} & a_{28} & a_{29} \\ a_{37} & a_{38} & a_{39} \end{pmatrix} \begin{bmatrix} R^2 \\ G^2 \\ B^2 \end{bmatrix} + \begin{bmatrix} a_{110} \\ a_{210} \\ a_{310} \end{bmatrix} \begin{bmatrix} RGB \\ RGB \\ RGB \end{bmatrix} \quad \textbf{[F. 18]}$$

6. 3x14 matrix: 3x11 and cubic terms

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} a_{10} \\ a_{20} \\ a_{30} \end{bmatrix} + \begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} & a_{16} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} & a_{26} \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} & a_{36} \end{pmatrix} \begin{bmatrix} R \\ G \\ B \\ RG \\ RB \\ GB \end{bmatrix} +$$

$$+ \begin{pmatrix} a_{17} & a_{18} & a_{19} \\ a_{27} & a_{28} & a_{29} \\ a_{37} & a_{38} & a_{39} \end{pmatrix} \begin{bmatrix} R^2 \\ G^2 \\ B^2 \end{bmatrix} + \begin{bmatrix} a_{110} \\ a_{210} \\ a_{310} \end{bmatrix} \begin{bmatrix} RGB \\ RGB \\ RGB \end{bmatrix} + \begin{pmatrix} a_{111} & a_{112} & a_{113} \\ a_{211} & a_{212} & a_{213} \\ a_{311} & a_{312} & a_{313} \end{pmatrix} \begin{bmatrix} R^3 \\ G^3 \\ B^3 \end{bmatrix} \quad \textbf{[F. 19]}$$

In each case we met the problem of over determined system. For example, in the first case there are only 9 variables, and thus we are in need only of 9 equations in order to solve the system. But we actually have 24 equations (1 equation for each color patch). One of the possible methods for solving this problem is least square root method (LSR).

Theorem: Let A be an $m \times n$ matrix ($m \geq n$), and b be a vector in R^m.
Then the system $A^T Ax = A^T b$ is consistent.
Moreover $\|Ax^* - b\| \leq \|Ax - b\|$ for all x in R^n, if and only if x^* is a solution of $A^T Ax = A^T b$.

Let us denote matrix of coefficients in each transforms as *coeff*. Then in these terms for example for the first case (linear terms) left part is *XYZ* and right part is *RGB*. Thus, equation for finding the solution by LSR is:

$$coeff = XYZ \cdot RGB^T \cdot \left(RGB \cdot RGB^T\right)^{-1} \qquad \textbf{[F. 20]}$$

In the same manner matrices of coefficients for other transformations can be calculated. Using these matrices we can convert Macbeth's *RGB* patches into *XYZ* and further to *CIELAB*. For each color patch and each transformation, calculated results (i.e. *LAB* values) can be compared to the "correct" LAB_c values obtained from XYZ_c using formulas [F. 9]. Needed white point tristimulus coordinates is calculated as follows:

$$X_n = \frac{1}{k}\sum_{\lambda=380}^{780} E(\lambda)\bar{x}(\lambda),\; Y_n = \frac{1}{k}\sum_{\lambda=380}^{780} E(\lambda)\bar{y}(\lambda),\; Z_n = \frac{1}{k}\sum_{\lambda=380}^{780} E(\lambda)\bar{z}(\lambda) \qquad \textbf{[F. 21]}$$

In our case coordinates of white point after normalization are: (0.9666, 1.000, 0.8430).

In order to be able to compare colors we have to define measure. For *CIELAB* the color difference is calculated using CIE76 $L^*a^*b^*$ formulae: $\Delta E = \sqrt{\Delta L^{*2} + \Delta a^{*2} + \Delta b^{*2}}$. Results are usually interpreted as following [15]:

Δ***E***	*Effect*
< 3	Not perceptible
> 3,< 6	Perceptible, but acceptable
> 6	Not acceptable

Now, when we are able to calculate all matrix transforms based on the testing color chart we can use them for converting *RGB* images of arctic char into *XYZ* and further to *CIELAB*.

However there are also direct methods for converting from *RGB* to *CIELAB*. The idea is to use testing color checker chart to produce *LAB* values first and only after that find transformation matrices from *RGB* to *LAB*. See schemes below (Figure 18 and Figure 19).

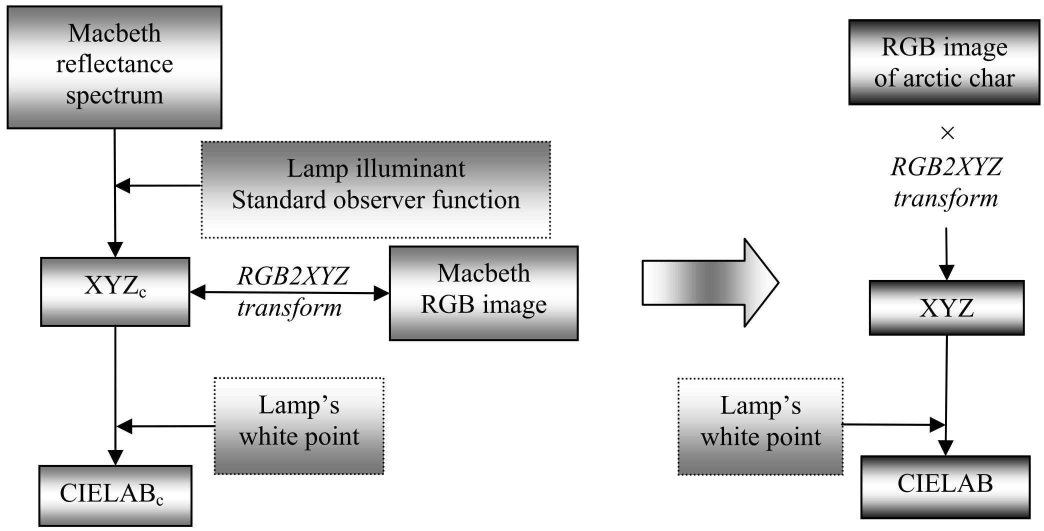


Figure 18: Indirect transformation: *RGB→XYZ* and only after that commonly to *CIELAB*

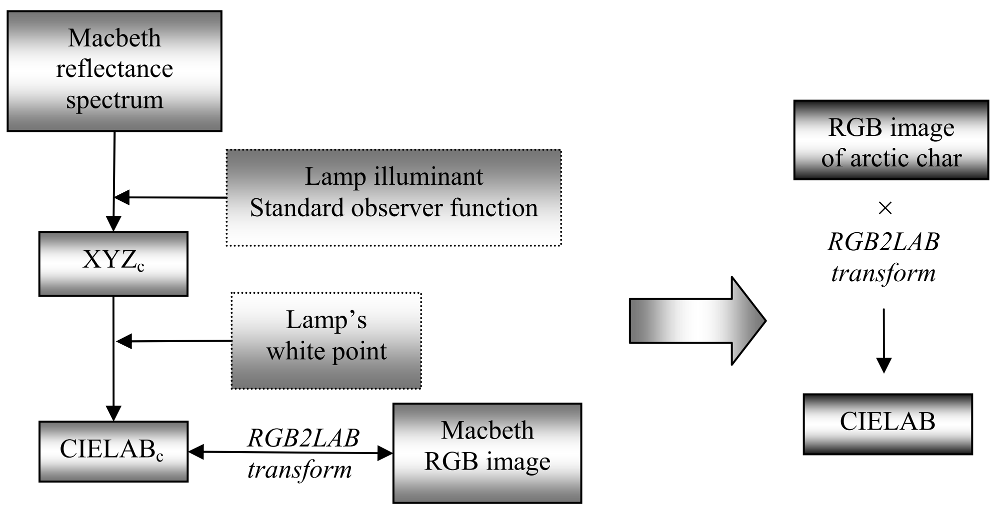


Figure 19: *RGB→CIELAB* directly

Formulas for *RGB2LAB* transform in general form can be written as

$$[L*a*b*] = [coeff] \cdot [RGB] \qquad \textbf{[F. 22]}$$

In [15] three polynomial approximations were defined. We used the same here.

1. First order polynomial approximation

$$\begin{bmatrix} L* \\ a* \\ b* \end{bmatrix} = [coeff_1] \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$ **[F. 23]**

2. Second order polynomial approximation

$$\begin{bmatrix} L* \\ a* \\ b* \end{bmatrix} = [coeff_2][R \quad G \quad B \quad R^2 \quad RG \quad RB \quad G^2 \quad GB \quad B^2]^T$$ **[F. 24]**

3. Third order polynomial approximation

$$\begin{bmatrix} L* \\ a* \\ b* \end{bmatrix} = [coeff_3][R\,G\,B\,R^2\;RG\,RB\,G^2\;GB\,B^2\;R^3\;R^2G\,R^2B\,RG^2\;RGB\,RB^2\;G^3\;G^2B\,GB^2\;B^3]^T$$

[F. 25]

Here again in order to calculate matrices of coefficients one needs to apply regression method. The same LSR method was applied here for solving these equations.

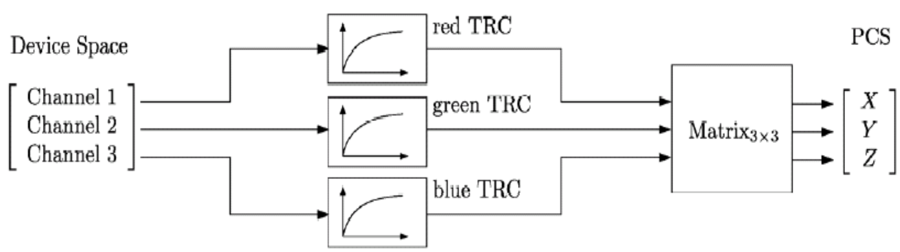


Figure 20: Transformation from device color space to profile color space

Two techniques for converting data from *RGB* to *L*a*b** based on test data and measured power spectra distribution of illuminant used for shooting fish images were described. We have to mention also that there is a method for the same purpose for which we do not have to measure illuminant. It is called three-component matrix based profile model [6] (Figure 20). This model describes transformation from device color space to PCS (profile connection space). The transformation is based on three non-interdependent per-channel tone reproduction curves to convert between non-linear and linear *RGB* values and a 3x3 matrix to convert between linear *RGB* values and relative *XYZ* values. Namely this transformation is used in Matlab 7.0 for conversion between CIE color spaces and the *sRGB* color space, which was defined by an industry group to describe the characteristics of a typical PC monitor.

3.3 Adopted segmentation techniques

Segmentation of the object

For finding the fish from the image histogram thresholding combined together with edge detection was adopted as segmentation technique. Otsu [28] method for calculating threshold value was used. It is rather simple and fast algorithm and we fortunately have at disposal rather suitable, almost bimodal gray scale version of given color images, where dark pixels belong to the object, while light ones represent blue background. The whole procedure is as following:

Pre-processing:

1. Take gray scale version of image (blue channel component) and calculate its histogram.

Segmentation:

2. Perform binarization of the image according to optimal threshold value calculated by Otsu method.
3. Find edges in the thresholded image using Sobel mask for gradient determination.

Post-processing:

4. Dilate the edges in order to have closed boundary, eliminate noise and small objects that are out of interest.
5. Fill interior gaps in order to have mask of the object.
6. Smooth the boundary by means of erosion.

Evaluation:

7. Create outline of the mask and superimpose it onto original image in order to estimate the result.
8. Calculate pixels within contour in order to have quantitative measure of the object.

In spite of more or less satisfactory performance of this algorithm for the most part of images, there were some on which it failed due to the presence of yellow strap. If we consider *RGB* color cube - these colors are represented by opposite vertices: Blue: (0,0,1), Yellow: (1,1,0); and it means that on the blue component objects of yellow color will appear as dark ones and thus might be segmented together with desired object, which is supposed to be darker then background. The only way to circumvent this problem is to get rid of yellow strap before thresholding. This can be done with the help of clustering procedure - pixels of yellow color obviously should form separate cluster. We can use it for producing a mask for extraction yellow from the original *RGB* image (for details see [8]). Further stages are the same as in ordinary case.

Segmentation of the color

The same approach – thresholding – can be used for determination pixels of red color directly from *RGB* image. If we look at intensities of red, green and blue components along the line that crosses (bottom-up in the Figure 21 below) regions of different colors: blue, red, dark gray and then again blue, it can be readily observed that pixels in the red area have rather big difference between color components and thus we only need a threshold value between red and green and red and blue for accomplishing this task.

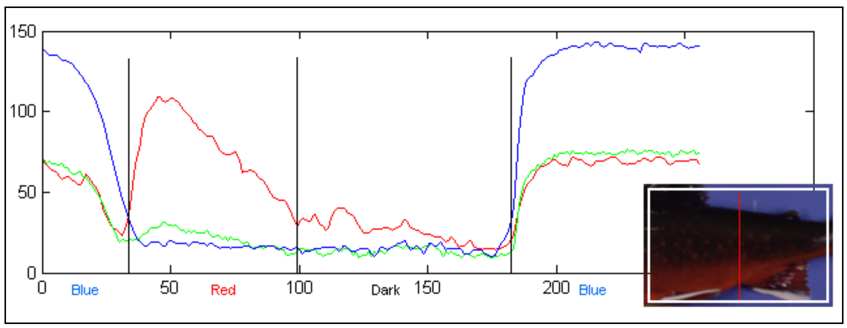


Figure 21: *RGB* intensities along the line

Pixels are supposed to be red if they satisfy following rule:

$$\begin{cases} R - B > th \\ R - G > th \end{cases}$$

Unfortunately, this threshold cannot be calculated automatically and should be determine manually for every image. It depends greatly on the illumination and saturation of the colors in particular image. On the other hand it is very fast and simple approach that can be done interactively. It was noted that the results of thresholding based on the concept of pixel redness with the appropriate value is very close to those gained by clustering in *CIELAB* color space.

Clustering in *CIELAB* actually implies taking into account only chromaticities $a*$ and $b*$. Experiments showed that use of third component – Luminance, $L*$ - does not improve results much, but increases the time of algorithm's performance considerably. Pictures of 3d and 2d clustering results can be found in the Appendix III. In the Figure 22 the calculated distribution of clusters in $a*b*$ is shown for the particular fish image.

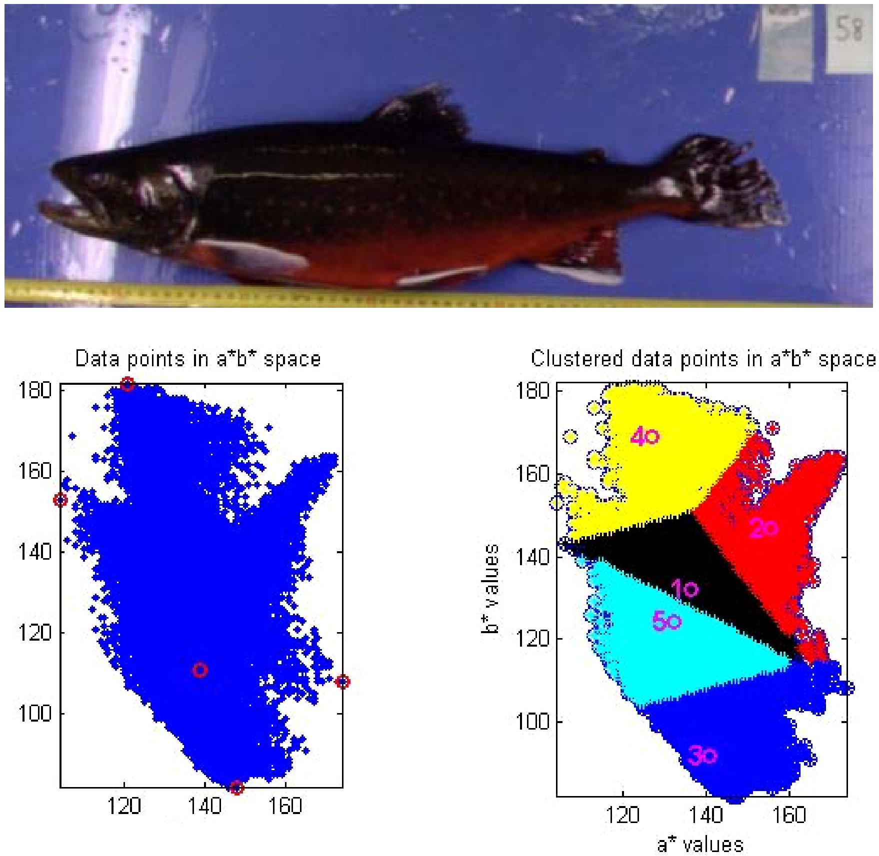


Figure 22: *RGB* image and distribution of its points in chromaticity (*a*b**) plane, after conversion to *CIELAB* color space; bottom left picture - initial positions of cluster centers; bottom right - final partition and positions of cluster centers

K-means batch mode algorithm was used for clustering. At first unsupervised k-means algorithm with random initialization stage of 5 clusters (according to the number of the colors that can occur in images: blue, dark gray, white, red and yellow sometimes) were chosen. But later initialization step were changed. If we look at the typical distribution of color points in the *a* b** plane (recall that *a** ranges from green to red, while *b** from blue to yellow), we can easily see salient regions that correspond to the constellation of points of red, yellow and blue colors. The idea of positioning initial cluster centers was to put them into the end points of both directions. This gives us 4 centorids; fifth one was assigned to the center of mass. Such ad hoc initialization, based on the analysis of distribution of point in the chromaticity plane provides better results if compare to random initialization, starting from which k-means sometimes gave strange results: yellow points were classified together with red. With described initialization this will never occur.

Here pre-processing stage includes conversion of *RGB* image to the *L*a*b** color space and preparing data for k-means algorithm. Post-processing stage involves interpretation of the clustering results, by constructing masks for separate clusters and backward conversion to *RGB* (Figure 23).

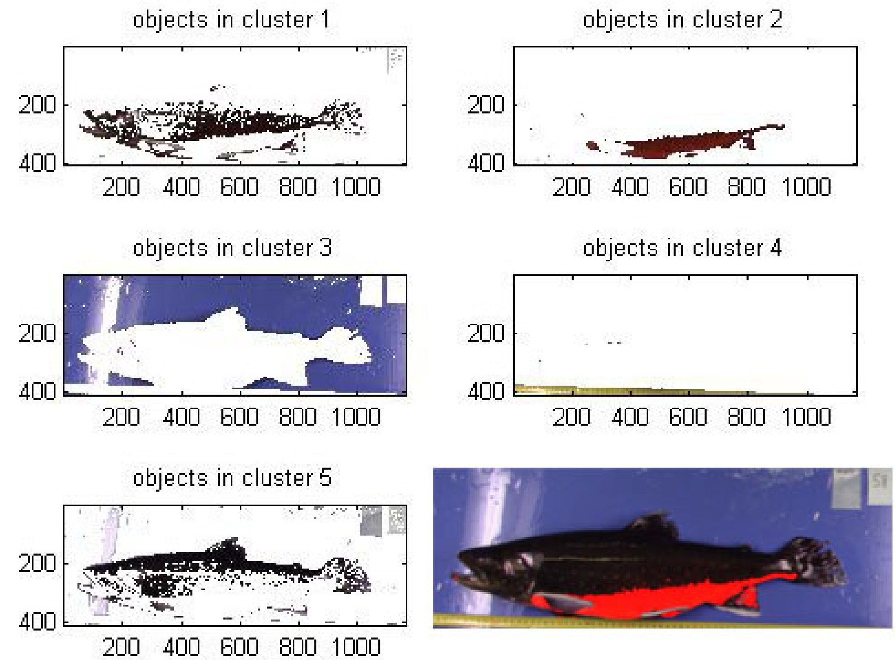


Figure 23: Interpretation of clustering results and original image with highlighted red area.

As it was more then once said, some images were bad focused and so dark, that clustering could not cope with the task of distinguishing red in them. In this situation equalization in the brightness component helped much. In the picture below there are clustering results for very dark image before and after such equalization. It is easy to see that in the first image region that corresponds to the concentration of red color pixels is divided into two clusters, while on the second picture clustering results are more expected.

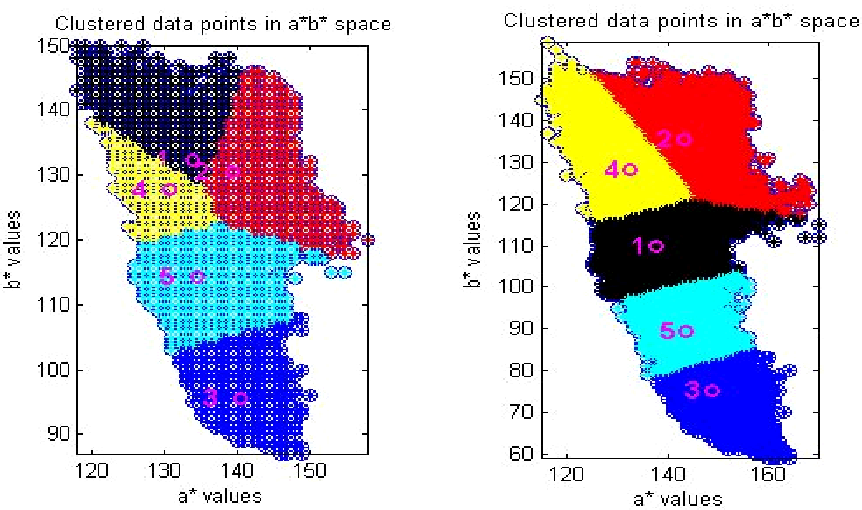


Figure 24: Distribution of clusters in *a*b** chromaticity plane for originally dark (left) and equalized image (right)

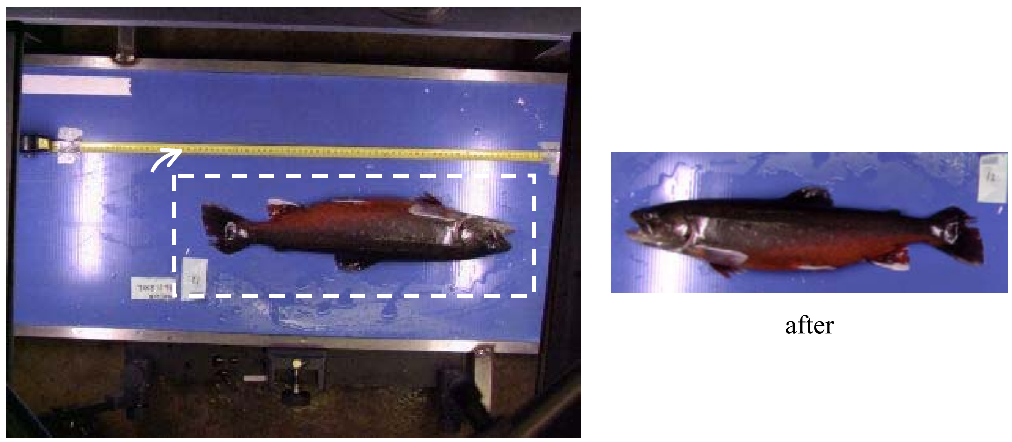
3.4 Post-processing

Post-processing techniques depend on results of segmentation algorithm used. We used morphological filtration for the results of object-oriented segmentation, such as dilation and erosion [13]. Results for segmentation of red colored regions are supposed to be compact due to physical properties of them and thus do not demand any special post-processing, except may be filtering of noise pixels that may be occasionally included as red ones. Post-processing here also includes interpretation of the clustering results and mapping them back to the spatial domain (*RGB*) to form separate clusters.

CHAPTER 4: EXPERIMENTAL RESULTS

4.1 Description of test images

Color images of fishes were acquired with the help of Minolta digital camera under illumination of 2 fluorescent lamps that modeled daylight and were saved in uncompressed tiff format.



after

before

Figure 25: Image from original set and view of the same image: cropped and clipped

Except fish, typical image contains yellow strap and pieces of white paper with identification number and date of shoot. All images have the same feature – blue background. The background has the property to reflect light, especially when it is wet; it is not entirely smooth and can contain white zones. In the Figure 25 the original condition of typical image is presented. Rightmost picture presents the image after cropping and clipping of original one. These operations have been done for all images in collection because of two reasons. First, originally each image includes a lot of additional, but not useful information about the environment. Hence, it was deleted as not wanted. Sometimes the task of getting rid of yellow trap required rotation on the arbitrary angle before cropping. Secondly, as it was already mentioned, images were stored in uncompressed format in order to provide as much information as it possible. Each of images took 5 Mb, which could not but influence the speed of processing them. After pre-processing the average size of image became 2-2.5 Mb.

It should be noted also that, despite of more or less equal conditions of shooting procedure, images vary greatly. About 10 percents of them are not focused well. There are also images, which are very dark or with low level of color saturation. See examples below (Figure 26). Consequently we have met some challenges that were solved particularly in the pre-processing step, while others – by post-processing. As a pre-processing we performed reduction of the intensity in the image to some average value, which was calculated through a number of blue patches taken from sufficiently big amount of well lightened images. The idea was that background should look the same on every image. When we got mentioned average value we then calculated coefficient for every image which was used then to tune current image intensity to the desired one.

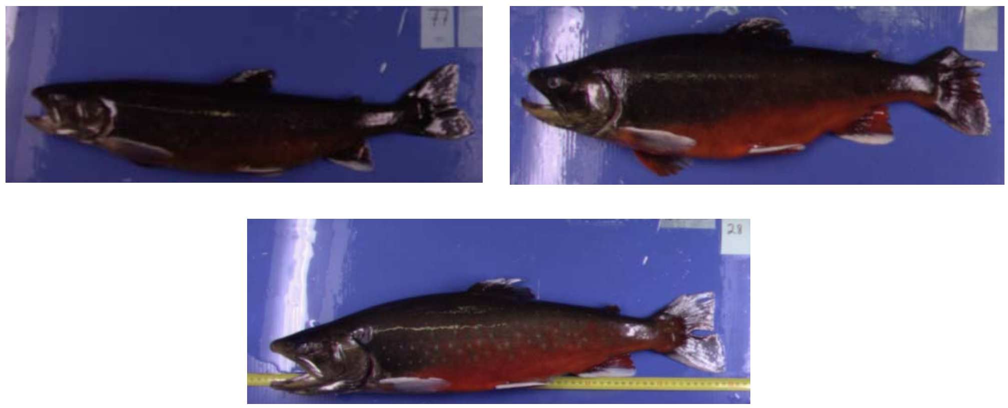
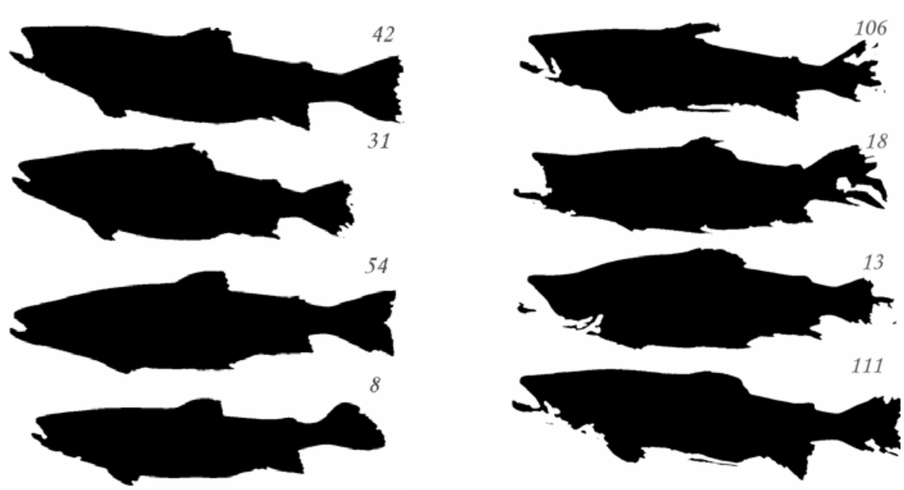


Figure 26: Examples of images: 1) dark, bad focused (blurred) image where colors are almost undistinguishable; 2) normal condition; 3) image with tape and fish overlapped

4.2 Segmentation results

Figure 27 and Figure 28 on the next page give step by step illustration of the object-oriented segmentation procedure. Results of the processing were saved as black and white binary masks with corresponding index of the fish. Though in whole algorithm gave satisfactory results (examples on the left picture below), on some images we have met problems with tail and head regions (right picture below), which were caused by the illumination conditions. Results of this phase are presented in Appendix II.



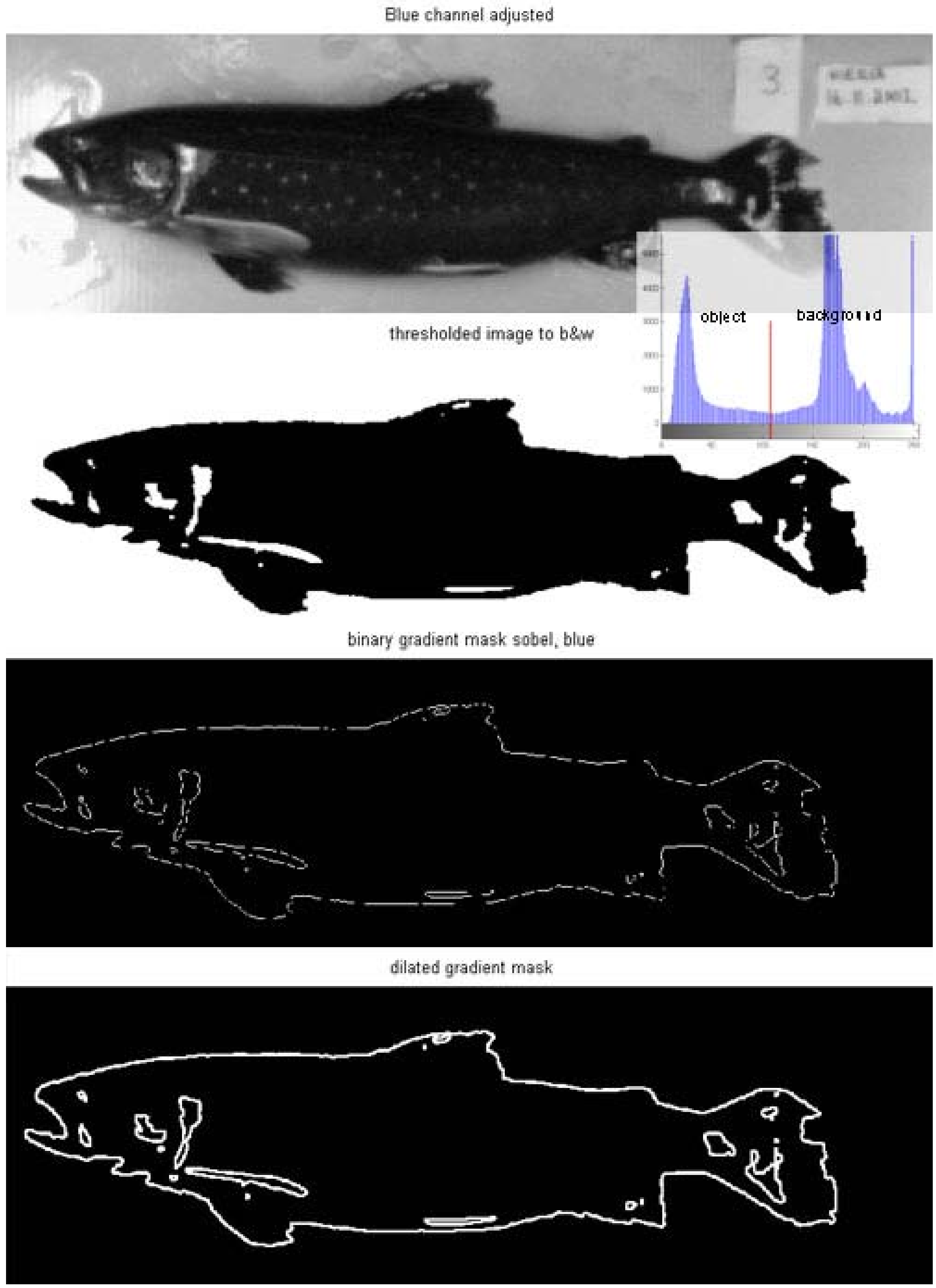


Figure 27: Normal image. First steps of segmentation of the fish algorithm: binarization according to threshold, determination of gradient and dilation of gradient

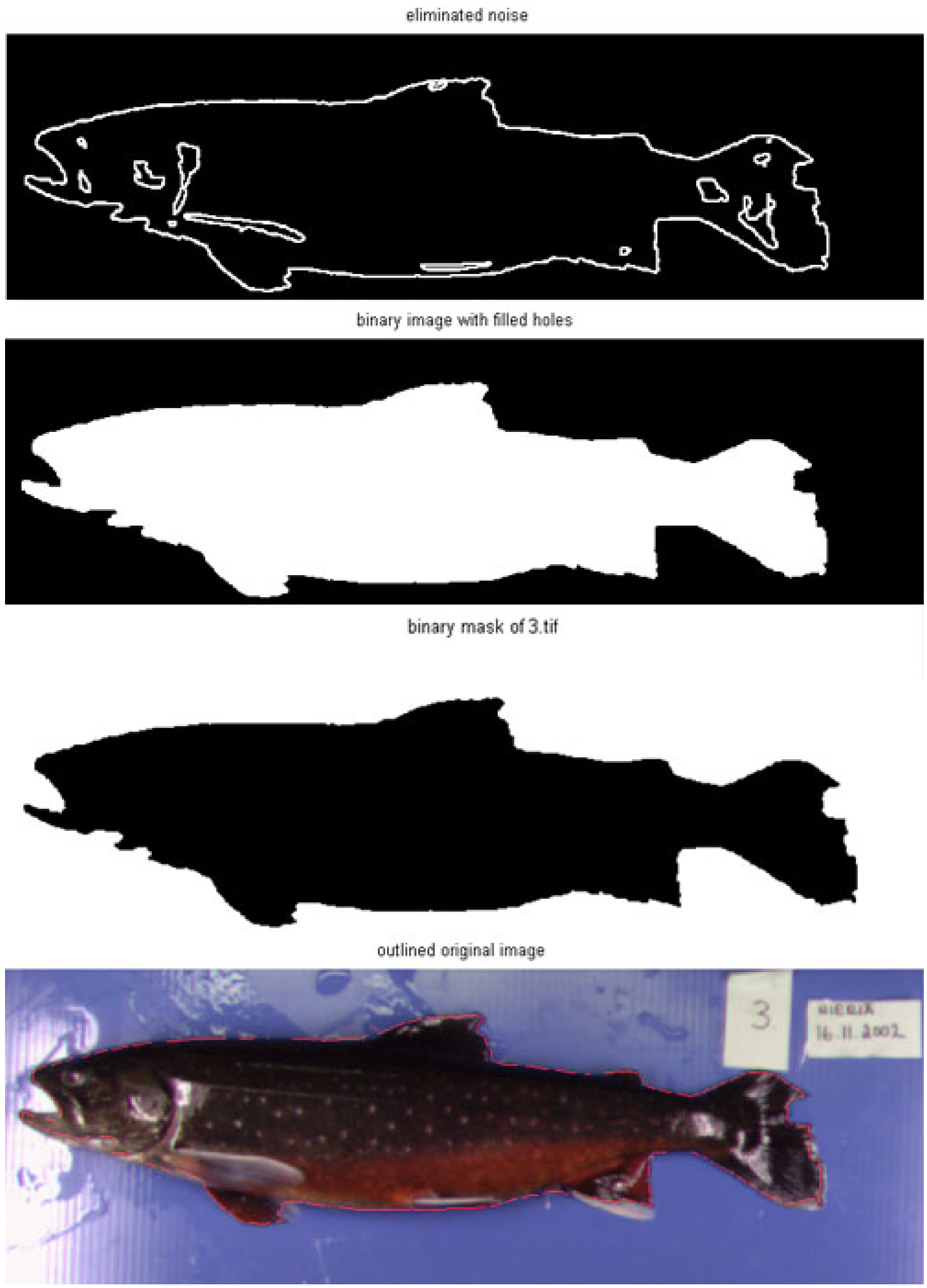


Figure 28: Second part of the algorithm: elimination of the noise, creation of the mask, smoothing the boundary, evaluation of the results

Results of red color segmentation also can be found in Appendix II. Here also a big part of images that had been clustered in a good form and provided homogeneous red colored area in the belly of fishes. Nevertheless on some of them we got unsatisfactory results (Figure 29). Together with needed red region there are also pixels in the head and tail that are not red, but were clustered as red.

In the real application we have solved this problem by constructing mask, only within which we were considered pixels for the classification. For further information see [8].

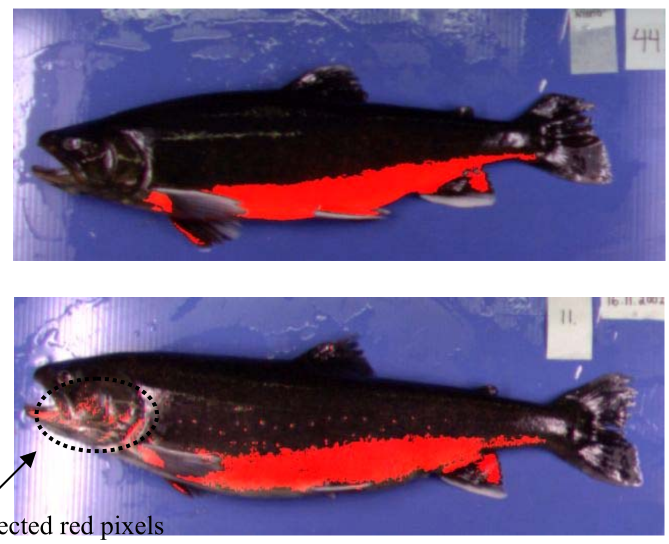


Figure 29: Examples of good (top) and unsatisfactory (bottom) clustering results for different images

CHAPTER 5: DISCUSSION

In this study we tried to suit known algorithms for image segmentation for the particular problem originated from real life. During this research several methods for pre-processing and post-processing given collection of color images were investigated to improve results of actual segmentation methods which were: thresholding for object localization and clustering in chromaticity plane for red color distinction.

Though these tasks were done independently, they have been combined in order to cope with some challenges met in the process of finding fish in the image. The case in point is overlapped objects such as yellow strap and fish body. Information about color of the tape was utilized to separate it from the fish before starting the thresholding algorithm for gray scale version of color image.

Though much effort were done to improve visual quality of the images before applying any technique, a part of them (about 10 %) anyway gave not very good results.

In the task of red color segmentation it was very interesting to see how particular way of transformation between color spaces (*RGB* to *CIELAB*) affect the results of segmentation, performed after conversion and investigate which one provide more reliable results. Several images from initial collection were subjected to this checking. At first manually selected mask for red was constructed for each of them. This played the reference point for algorithm of segmentation. Then different approximation to direct and indirect *RGB2LAB* transformations were done and followed by clustering. Clustering results were compared to the reference one using true/false positives analysis.

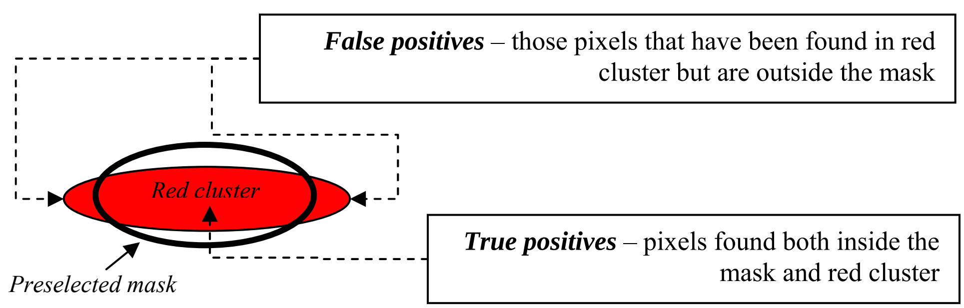
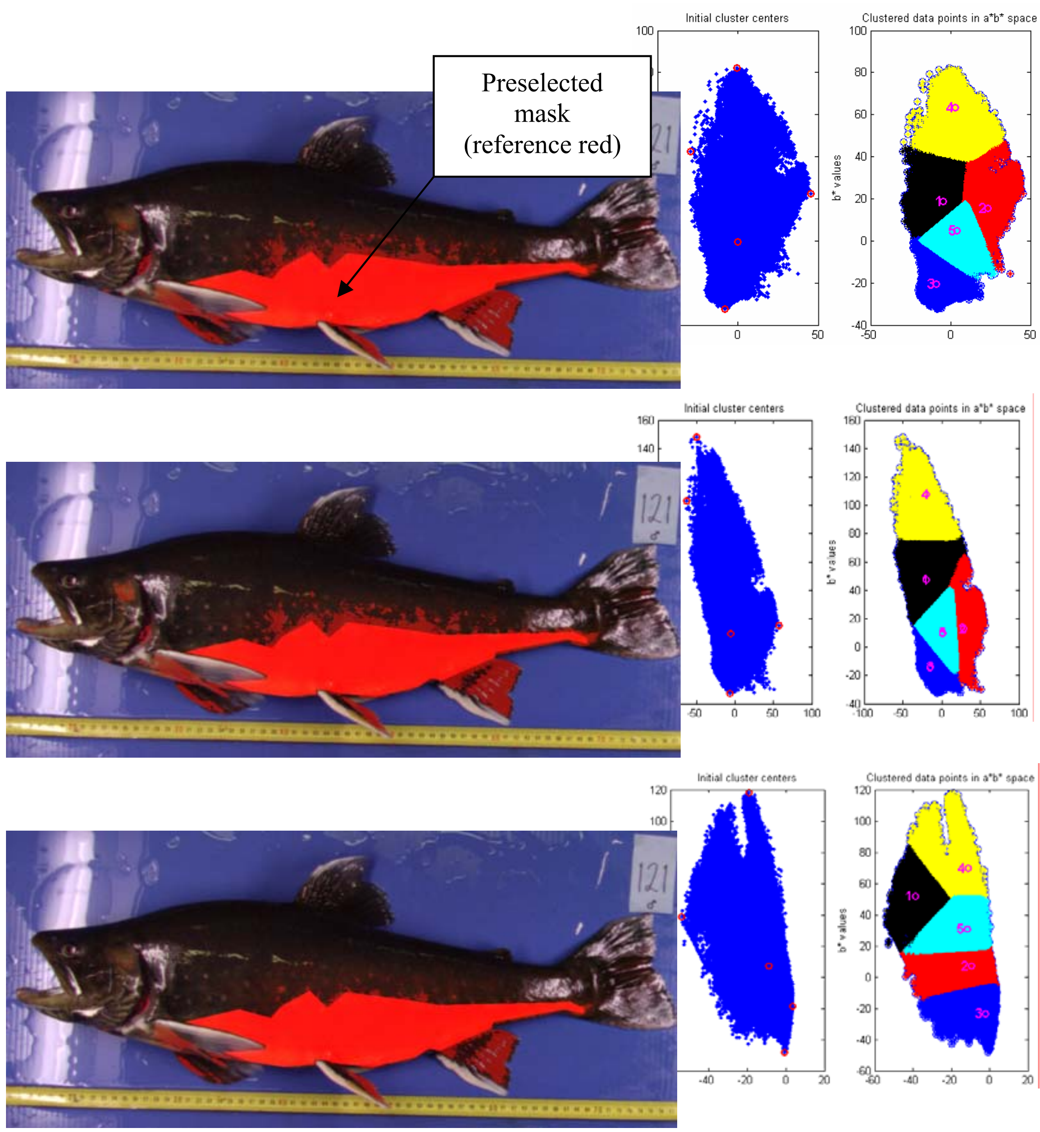


Figure 30: True/false positives analysis scheme

It has been noted, that direct transformation in average is better then indirect. The best results are achieved through 3d-order polynomial approximation of it. Example of these comparisons for particular image can be seen below.



fish#	approx	mask	red cluster	true positives	false positives	Percents, %	
121	1	75143	117606	74111	43495	63,01634	36,98366
	2		114189	73829	40360	64,65509	35,34491
	3		89105	68487	20618	76,86101	23,13899

On the other hand we also applied three-component matrix based profile model, mentioned in the end of section about *RGB* to *CIELAB* conversion. It turned out that this conversion provides distribution of the colors in *a*b** which gives more promising clustering results, despite the fact that it doesn't take into account information about the light source used during process of making pictures of fishes.

List of figures

REFERENCES

[1] Bazi Y., “A comparative study of histogram based thresholding algorithms”. URL: http://science.unitn.it/~tomasi/think/pdf/sbazi.pdf. (Dec 2004)

[2] Brook A., Kimmel R., Sochen N.A., “Variational Segmentation for Color images”, 2003, URL: http://citeseer.nj.nec.com/506053.html (Aug 2004)

[3] Carminati L., “Image segmentation overview”, URL: http://micasoft.free.fr/Rapports/Cours de segmentation d'image/ (Nov 2004)

[4] Carron T. and Lambert P.,”Color Edge Detector Using Jointly Hue, Saturation and Intensity”, *Proc. of ICIP’94* (13-16 Nov 1994), vol. III, pp. 977-981, Austin, TX.

[5] Cheng H. D., Jiang X. H., Sun Y. and Jing Li Wang, “Color Image Segmentation: Advances and Prospects”, *Pattern Recognition*, vol. 34, pp. 2259-2281, 2001.

[6] Color Consortium specification ICC.1:2001-04 URL: http://www.color.org/icc_specs2.html (Apr 2005)

[7] CVonline: The Evolving, Distributed, Non-Proprietary, On-Line Compendium of Computer Vision, URL: http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/OWENS/LECT14/lecture12.html (Jan 2005)

[8] Doronina E., “GUI for processing images of Arctic Char”, documentation for IT project at the University of Joensuu, department of Computer Science, Jan 2005.

[9] Foley J. D., van Dam A., Feiner S. K., Hughes J.F., “Computer Graphics Principles and Practice”, 2d edition, 1996, Addison-Wesley New York.

[10] Frigui H., Krishnapuram R., ”A robust competitive clustering algorithm with the applications in computer vision”, *IEEE Transactions on Pattern Analysis and Machine Vision*, vol.21, no.5, pp. 450-465, 1999.

[11] Fu K. S. and Mui J. K. “A Survey on Image Segmentation”, *Pattern Recognition*, vol. 13, pp. 3-16, 1981.

[12] Game and Fishery Research Center in Enonkoski, Finland, Internet page, URL: http://www.rktl.fi/english/fish/ (Nov 2004)

[13] Gonzalez R. C., Woods R. E. “Digital image processing”, Addison-Wesley publishing, 1992.

[14] Haralick, R. M., Shapiro L. G., “Image segmentation techniques”, *Computer Vision, Graphics and Image Processing*, vol. 29, no.1, pp. 100-132, Jan 1985.

[15] Hardeberg J. Y., “Transformations and Colour Consistency for the Colour Facsimile”. A dissertation submitted in partial fulfillment of the requirements for the degree of Sivilingeniør (M.Sc equivalent) at the Norwegian Institute of Technology (NTH), Trondheim, Norway. The work was effectuated at the ´Ecole Nationale Sup´erieure des T´el´ecommunications, Paris, France, 1995.

[16] Icy Waters Ltd web page. URL: http://www.icywaters.com/charr/charr.htm (Aug 2004)

[17] Irish Charr Conservation Group, URL: http://www.charr.org/ICCG/why.htm (Aug 2004)

[18] Internet FAQ archives, “How many kinds of Kohonen network exist and what is k-means”. URL:http://www.faqs.org/faqs/ai-faq/neural-nets/part1/section-11.html (Apr 2005)

[19] Kang H.R., “Color Scanner Calibration of reflected samples”, SPIE vol. 1670, *Color Hard copy and Graphic Arts*, 1992.

[20] Kass M., Witkin A. and Terzopoulos D., “Snakes: Active Contours Models”, *Int’l Journal of Computer Vision,* vol.1, pp.321-331,1988

[21] Lindbloom, Bruce. Official web-site, URL: http://brucelindbloom.com/Eqn_RGB_XYZ_Matrix.html (Jan 2005)

[22] Lucchese L. and Mitra S.K.,"Color Image Segmentation: A State-of-the-Art Survey" (invited paper) *Image Processing, Vision, and Pattern Recognition*, Proceedings of the Indian National Science Academy (INSA-A), New Delhi, India, vol. 67, A, No. 2, pp. 207-221, March 2001.

[23] Lucchese L. and Mitra S.K., "Unsupervised Segmentation of Color Images Based on k-means Clustering in the Chromaticity Plane," Proc. of IEEE Workshop on Content-based Access of Images and Video Libraries (CBAIVL'99), Fort Collins, CO, pp. 74-78, 22 June 1999.

[24] Mahfoudth Ould Ahmed Taleb, *"*Combined algorithms for color image segmentation*",* PhD Thesis, Institute for Technical Cybernetics, Belarus 2002. (in Russian)
URL: http://handysolution.com/science/phd/mahfoudh_diss.rar (Jul 2004)

[25] Martinkauppi B., "Face colour under varying illumination- analysis and applications", academic dissertation, Oulu 2002; URL: http://herkules.oulu.fi/isbn9514267885 (Dec 2004)

[26] Mukherjee J.,"MRF clustering for segmentation of color images*",* *Pattern Recognition Letters*, 23, pp. 917-929, 2002.

[27] Ohta Y.I., Kanade T., Sakai T., "Color information for region segmentation", *Computer Graphics and Image processing*, vol. 13, pp. 222-241, 1980.

[28] Otsu N., "A threshold selection method from gray level histograms", *IEEE Trans. Syst, Man Cybern,* SMC-9, pp. 62-66 (1979)

[29] Pages on Color Vision, department of Paper Engineering, Chemical Engineering and Imaging in Western Michigan University,
URL: http://www.wmich.edu/ppse/color (Jan 2005)

[30] Pal N.R. and Pal S.K, "A Review on Image Segmentation Techniques", *Pattern Recognition*, vol. 26, no. 9, pp. 1277-1294, 1993.

[31] Pan-European Intranet funded by the EU, URL: http://www.charrnet.org/charrnet?template=help%2CAbout.vm (Nov 2004)

[32] Poynton Ch. A., "FAQ about Color",
URL: http://www.poynton.com/ColorFAQ.html (Dec 2004)

[33] Pratt W.K, "Digital image processing", Wiley, New-York, 1978

[34] Pratt W.K, "Digital image processing", A Wiley-Interscience Publication, 3rd edition, 2001.

[35] Prosec J., official web site, URL: http://www.troutsite.com/index.html (Oct 2004)

[36] Russ J.C., "Optimal greyscale images", *Journal for Computer Assisted Microscopy,* vol. 7(4), pp. 221-234.

[37] Sezgin M., Sankur B., "Survey over image thresholding techniques and quantitative performance evaluation", *Journal of Electronic Imaging*, 13(1), pp. 146-165, Jan 2004.

[38] Skarbek W., Koschan A., "Colour Image Segmentation - A Survey". Technishe Universitat Berlin, Berlin, Germany, 1994.

[39] Sonka M., Hlavac V., Boyle R., "Image Processing, Analysis and Machine Vision", 2d edition, Pacific Grove (CA): PWS Publishing, 1999.

[40] Tremeau A. and Borel N.,"A Region Growing and Merging Algorithm to Color Segmentation", *Pattern Recognition*, vol.30, no.7, pp.1191-1203, 1997

[41] Vandenbroucke N., Macaire L. and Postaire J. G. "Color image segmentation by pixel classification in an adapted hybrid color space. Application to soccer image analysis", *Computer Vision and Image Understanding*, Vol. 90 (2003), pp. 190 - 216.

[42] Wikipedia, online free encyclopedia. URL: http://en.wikipedia.org (Feb 2005)

[43] Young, I. T., Gerbrands J. J., van Vliet L. J, "Image Processing Fundamentals" (interactive web-course in Delft University, Netherlands), URL: http://www.ph.tn.tudelft.nl/Courses/FIP/noframes/fip-Segmenta.html (Dec 2004)

APPENDIX I

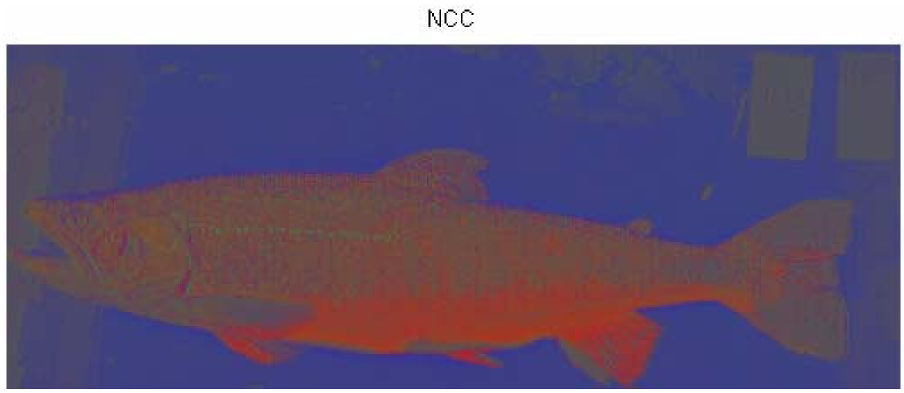


Figure 31: *RGB* representation of *NCCrgb* color space

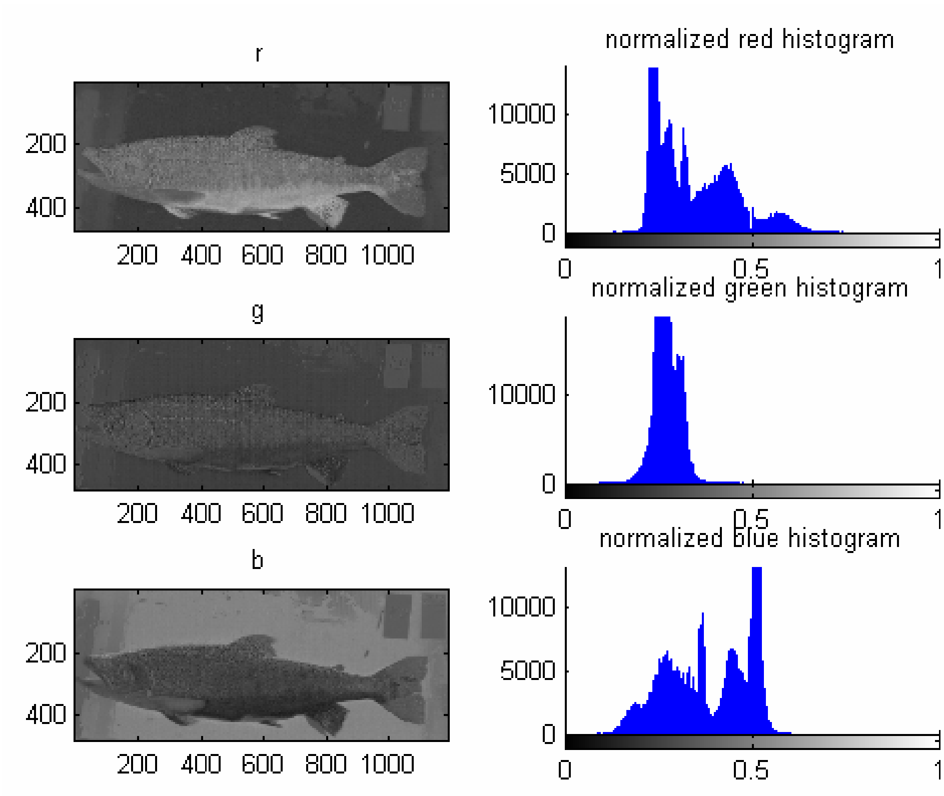


Figure 32: Components of *NCCrgb* and corresponding histograms

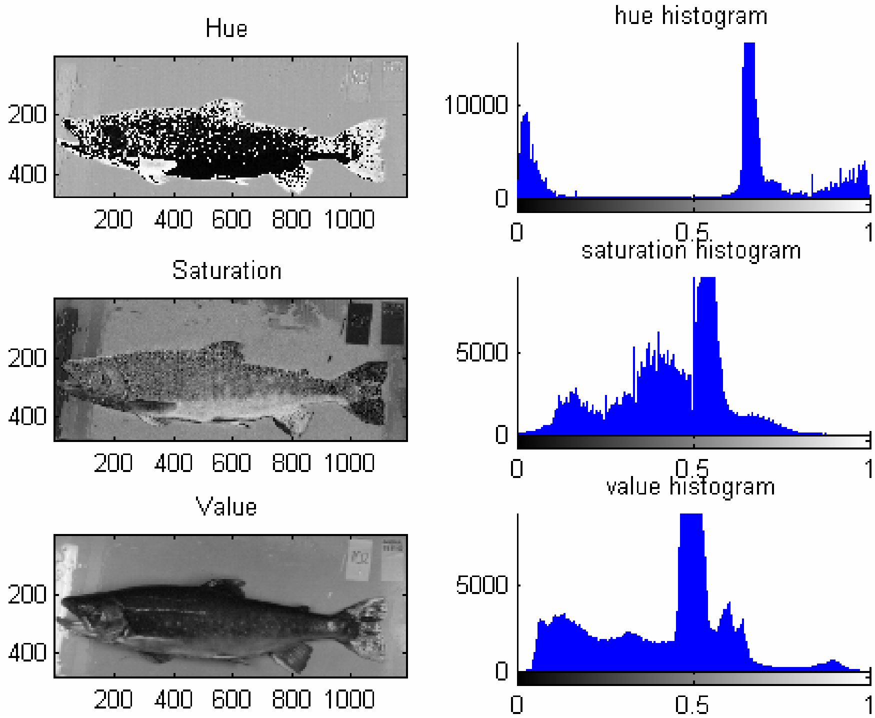


Figure 33: *HSV* components with corresponding histograms

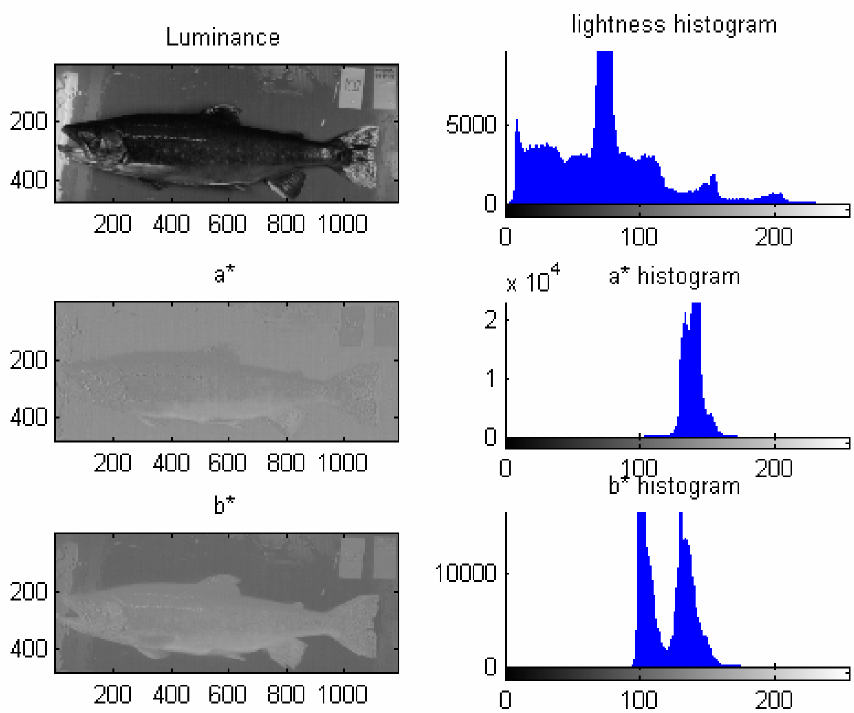
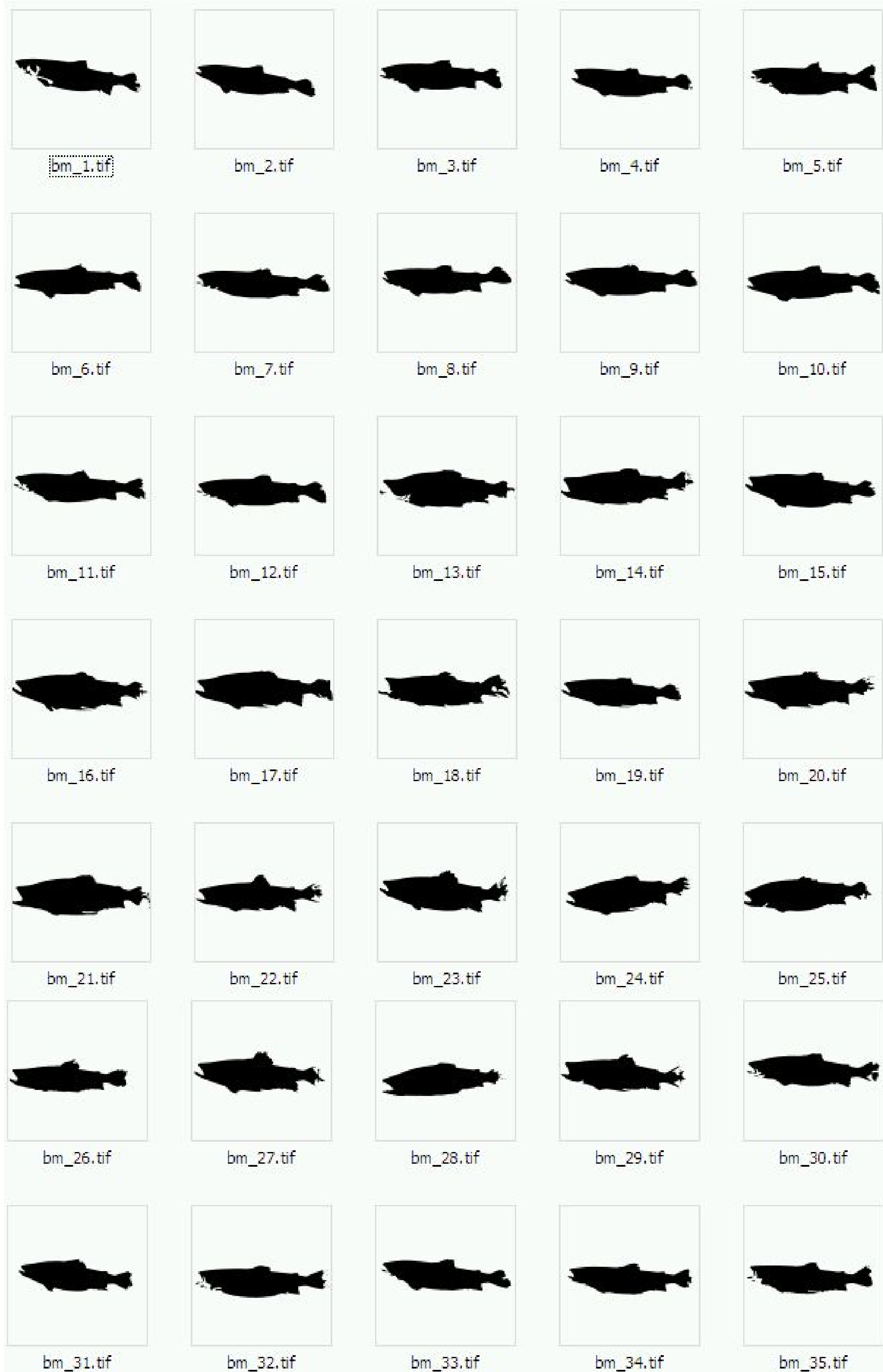
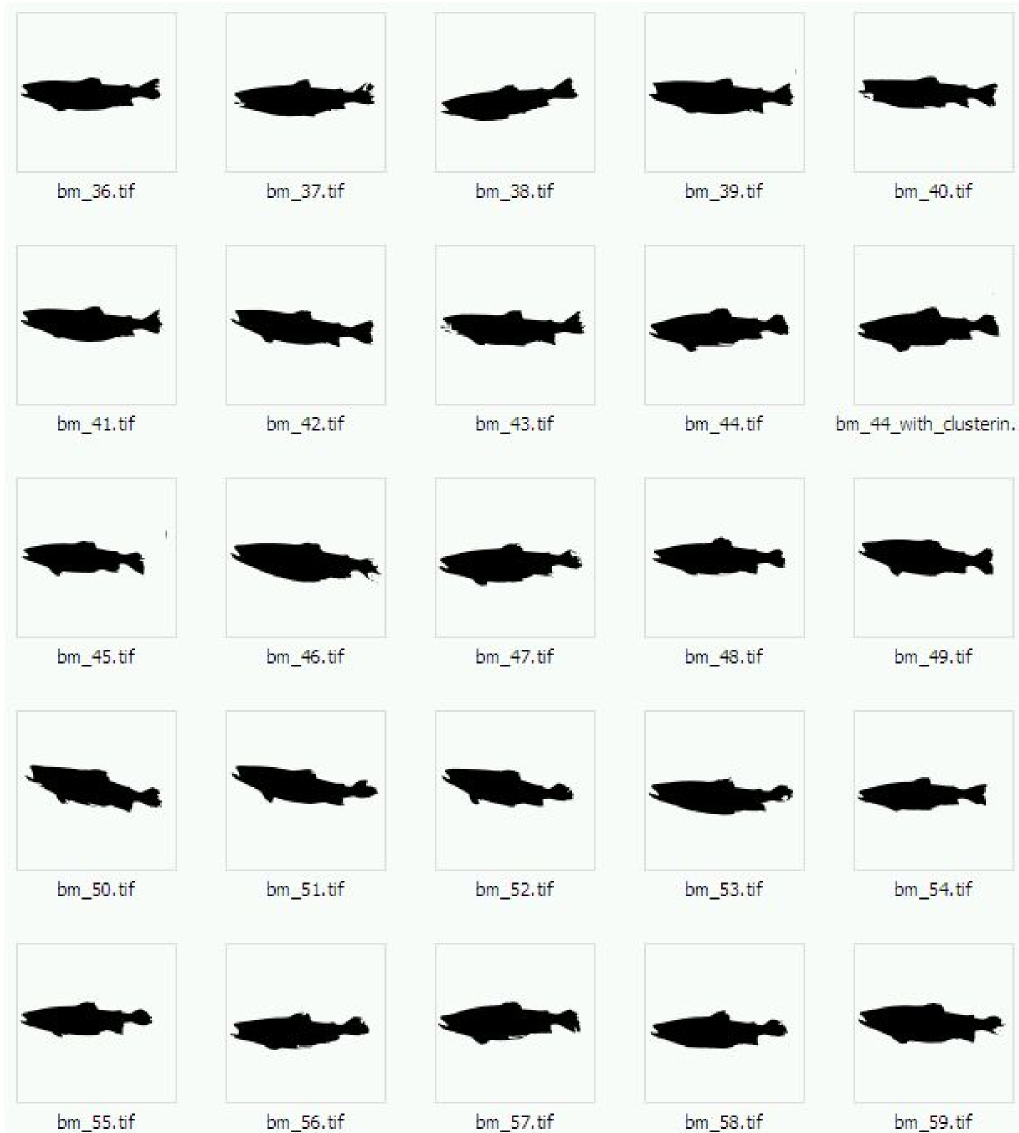
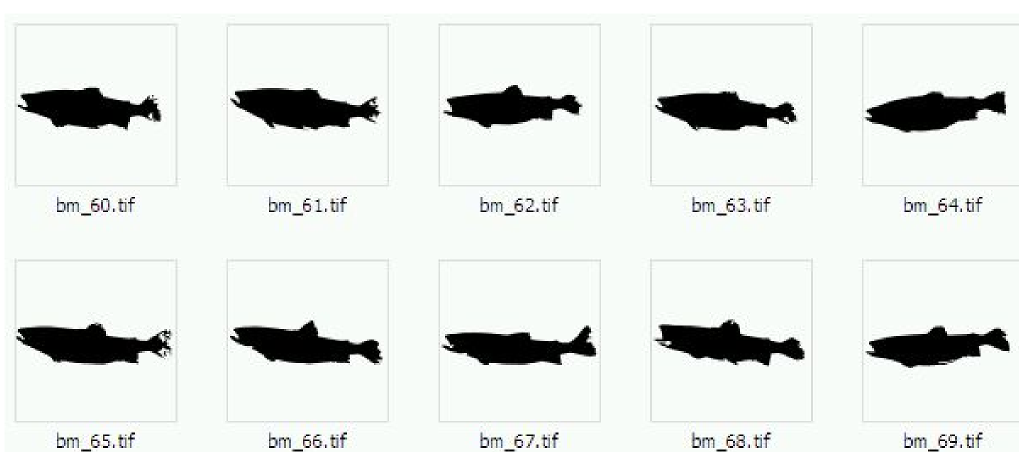


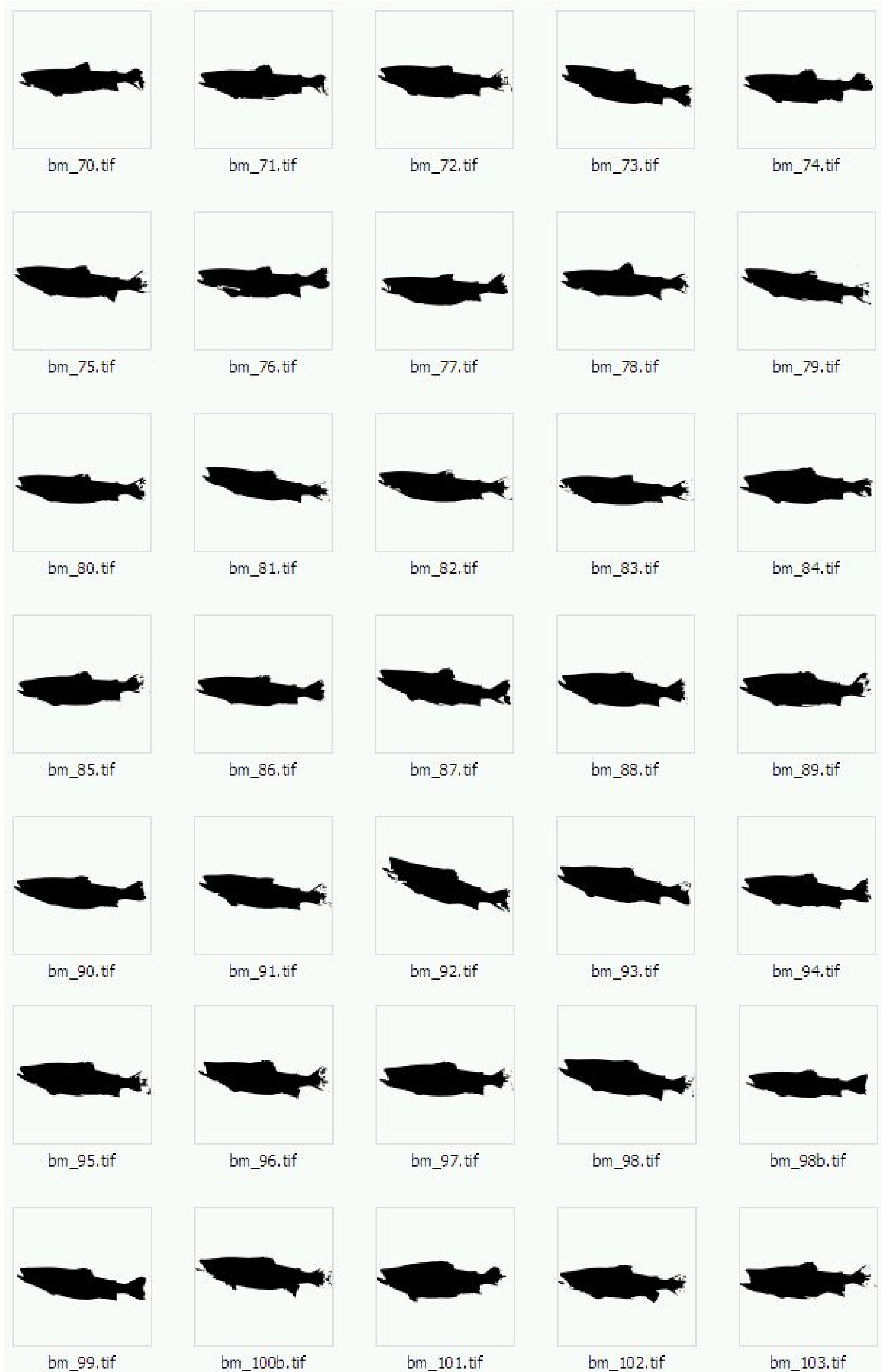
Figure 34: *L*a*b** components and corresponding histograms

APPENDIX II


bm_1.tif
bm_2.tif
bm_3.tif
bm_4.tif
bm_5.tif
bm_6.tif
bm_7.tif
bm_8.tif
bm_9.tif
bm_10.tif
bm_11.tif
bm_12.tif
bm_13.tif
bm_14.tif
bm_15.tif
bm_16.tif
bm_17.tif
bm_18.tif
bm_19.tif
bm_20.tif
bm_21.tif
bm_22.tif
bm_23.tif
bm_24.tif
bm_25.tif
bm_26.tif
bm_27.tif
bm_28.tif
bm_29.tif
bm_30.tif
bm_31.tif
bm_32.tif
bm_33.tif
bm_34.tif
bm_35.tif


bm_36.tif
bm_37.tif
bm_38.tif
bm_39.tif
bm_40.tif
bm_41.tif
bm_42.tif
bm_43.tif
bm_44.tif
bm_44_with_clusterin...
bm_45.tif
bm_46.tif
bm_47.tif
bm_48.tif
bm_49.tif
bm_50.tif
bm_51.tif
bm_52.tif
bm_53.tif
bm_54.tif
bm_55.tif
bm_56.tif
bm_57.tif
bm_58.tif
bm_59.tif


bm_60.tif
bm_61.tif
bm_62.tif
bm_63.tif
bm_64.tif
bm_65.tif
bm_66.tif
bm_67.tif
bm_68.tif
bm_69.tif


bm_70.tif
bm_71.tif
bm_72.tif
bm_73.tif
bm_74.tif
bm_75.tif
bm_76.tif
bm_77.tif
bm_78.tif
bm_79.tif
bm_80.tif
bm_81.tif
bm_82.tif
bm_83.tif
bm_84.tif
bm_85.tif
bm_86.tif
bm_87.tif
bm_88.tif
bm_89.tif
bm_90.tif
bm_91.tif
bm_92.tif
bm_93.tif
bm_94.tif
bm_95.tif
bm_96.tif
bm_97.tif
bm_98.tif
bm_98b.tif
bm_99.tif
bm_100b.tif
bm_101.tif
bm_102.tif
bm_103.tif

The following table presents numerical results collected during the processing procedure applied on each image. First column stands for the index of the image and refers also to the certain individual in the collection. Second column gives the numerical measure of the binary mask (in pixels). Binary mask supposed to cover the area of the fish body. If the quality of the binary mask found for the particular fish is not very good, it is marked in the Comments column. This column also contains comments on the results of the k-means algorithm, which are also reflected in the third column. Fourth column presents the relation between data collected in second and third columns multiplied by 100 in order to have a measure of fish‘s redness. Abbreviation “cropmask” is used where manually selected for cropping fins and head before clustering procedure can improve the results much. For details see [8]. If the results can be considered as good ones only with cropping mask the cells are highlighted with dark grey color (like for image #4). If there are general problems with the image like both binary mask and clustering are not very good then the whole row is highlighted with light gray color.

Image #	Binary mask pixels	Number of red pixels in red cluster	Red %	Comments
1	74034	17399	23,29	in binary mask: gaps in head region; bottom hind fin included in red area: cropmask were used to cut it out
2	93882	19814	21,70	some red pixels found in the head: better to use cropmask
3	107341	17799	17,00	red found in bottom fins, both hind and fore
4	113973	26277	20,75	clustering provides bad results: only with cropmask
5	103750	34233	26,83	only with cropmask, clustering produces bad results
6	94325	20503	21,10	red in hind fin: use cropmask

7	83555	16933	20,19	red in fins: use cropmask
8	104417	23297	21,03	red fins
9	109727	24072	21,40	red fins
10	99963	19252	18,65	red fins
11	120991	29635	24,24	many wrong red pixels in head: use cropmask
12	106036	23892	22,09	
13	194074	52739	22,02	binary mask: gaps in head and tail, clustering provides many wrong red pixels (also red digits are included) : use cropmask
14	213819	58697	25,09	binary mask: bad in tail; clustering gives too much additional red (also digits are included there): use cropmask
15	181201	36368	19,85	
16	176021	31490	17,54	
17	221226	43802	18,52	red fins:use cropmask
18	188578	32775	15,94	binary mask is not very good: problems in tail and head
19	158539	28361	17,98	red fins
20	185939	41478	18,75	tail in binary mask is poor; red fins: use cropmask
21	240583	54876	20,46	binary mask: problems in tail and bottom
22	160137	25345	15,72	clustering provides quite many wrong red pixels in head and fins
23	162629	33024	19,50	image should be rotated 3 deg ccw before processing (23.rot)
24	176336	40964	13,49	
25	198167	30827	15,60	tail is poor in binary mask
26	146614	45854	19,28	bad red cluster: only with cropmask
27	149215	31200	21,03	red fins, some red in head and tail: use cropmask
28	157125	42280	24,36	binary mask: problem in tail; digits and fin's red pixels are included in red region:use cropmask to avoid them
29	153359	39682	26,20	red fins: use cropmask
30	124577	19330	16,31	red hind and front fins: use cropmask
31	119570	29075	22,77	red hind and front fins: use cropmask

32	139139	28759	22,36	
33	109795	26254	22,91	wrong pixels in head and fins: use cropmask
34	140666	32454	23,18	red in head: use cropmask
35	143452	27260	19,21	red hind: use cropmask; gaps in head in binary mask
36	128275	30031	23,27	redundancy in head and fins: use cropmask
37	117872	31455	26,50	red fins and some red in tail: use cropmask
38	135470	31487	15,81	contains strap; redundancy in red cluster: in head, in tail and in body; red digits: only with cropmask
39	150047	44370	25,17	redundancy in head: only with cropmask
40	124467	33030	24,90	many wrong red pixels in head and in fins: only with cropmask
41	142865	37512	21,24	dark image, low saturated, redundancy in head: only with cropmask
42	141485	35798	26,49	dark image, bad performance both in clustering and in thresholding: only with cropmask
43	123800	33068	24,09	red cluster is not good: red in head, red fins: only with cropmask
44	130553	21465	16,96	some wrong red in fins: use cropmask
45	110785	19994	16,51	some redundant red in head and fins: use cropmask
46	154412	27606	17,58	poor tail in binary mask, some wrong red in fins: use cropmask to get rid of them
47	127196	23312	18,16	dark image, red cluster is noisy: in fins and in head
48	147632	28286	17,27	noise in red cluster: head, fins: use cropmask
49	135320	22731	16,88	red fins: use cropmask
50	147319	30631	19,77	red cluster is noisy: in head and in fins: use cropmask
51	153693	29218	14,74	very dark image, low saturated, even clustering produces poor resullts for red: (only with cropmask)
52	138752	27638	19,19	red fins:use cropmask
53	125982	21395	15,58	mistakes in head: use cropmask
54	138051	23326	15,99	dark image, redundancy in head: only with cropmask
55	121356	19100	15,45	dark image, low saturated, redundancy in head: only with cropmask
56	114854	21113	17,70	dark image, redundancy in head, tail and fins: only with cropmask

57	173091	30535	17,90	red fins: use cropmask
58	159456	31964	18,63	contains yellow strap: red digits, red fins: only with cropmask
59	162364	35386	20,14	dark image, redundancy in head, fins and body, only with cropmask
60	148276	21986	13,75	red fins: use cropmask
61	171364	43106	22,28	redundancy in head and red fins: only with crop mask
62	144129	23860	15,36	small amount of red in head, red fins: use cropmask
63	164677	354180	20,78	red fins: use cropmask
64	149869	28894	16,93	dark image, small amount of red in head fins and tail: use cropmask
65	271687	69167	21,36	dark image (low hue) clustering in LAB also give poor results for red: redundancy in head and body: only with cropmask
66	271175	38797	13,01	dark image, bad performance: in head and tail and fin: only with cropmask
67	143859	24252	17,08	very dark image, low saturated, even clustering sometimes can't distinguish red color: only with cropmask
68	206081	50498	23,36	dark image, many wrong red pixels in head and fins: only with cropmask
69	208825	60281	24,00	dark image, many wrong red pixels in head and fins: only with cropmask
70	186168	35228	18,34	dark image, wrong red pixels in head region and fins: use cropmak
71	177442	31791	17,69	very dark, out of focus, bad performance,even clustering gives poor results for red: only with crop mask
72	243776	6543815	26,84	very dark, low hue, out of focus, only with cropmask, clustering gives poor red
73	229640	72707	26,34	very dark, low hue, out of focus, only with cropmask, clustering gives poor red
74	211841	53097	24,66	dark image, redundancy in head and fins: only with cropmask
75	287906	58521	18,93	dark image, bad performance in clustering, problem in head region, should be rotated 3deg ccw before processing 75rot.tif
76	250045	50090	17,65	dark image, red fins: use cropmask
77	223078	64685	-	very dark image, bad performance of clustering: many redundant red in head and tail: only with cropmask
78	202597	46032	-	very dark image, bad performance of clustering: many redundant red in head and tail: only with cropmask
79	235177	44118	-	very dark image, bad performance of clustering: many redundant red in head and tail: only with cropmask
80	214660	58182	-	very dark image, bad performance of clustering: many redundant red in head and tail: only with cropmask

81	227252	50042	20,35	red in fins and head region, better with cropmask
82	249102	70484	21,76	dark image, low hue, bad performance :only with cropmask
83	221002	61803	22,47	dark image, low hue, bad performance: only with cropmask
84	271735	59328	20,74	red fins: use cropmask
85	241296	36082	16,20	dark image: only with cropmask
86	240218	43231	16,36	red fins, some red dots in body: use cropmask
87	214372	36555	14,60	dark image, redundancy in head and fins, some red dots in body: only with cropmask, should be rotated before use
88	277334	51709	15,01	dark image,low hue, bad performance of clustering: only with cropmask
89	222758	36018	15,78	red in head and fins: use cropmask
90	240936	31132	13,13	some wrong pixels in head, better with cropmask
91	222696	41970	14,44	dark image, some wrong pixels in head and body: only with cropmask
92	239599	50139	14,69	dark image, bad performance of clustering :head and fins found to contain many redundant red: only with cropmask
93	300020	67174	20,27	contains tape, should be rotated before use and cropped, some wrong pixels on body and fins: use cropmask
94	230877	42561	16,15	small amount of wrong red in head, red fins: use cropmask
95	268813	63900	20,20	wrong pixels in the lip and tail, red fins: with cropmask only
96	203545	40017	17,56	dark image,red in head and hfins: only with cropmask
97	273358	29527	15,95	dark image, wrong pixels in the body and on head:: only with cropmask
98	274362	55504	17,54	dark image, some wrong pixels in body and head: only with cropmask
98b	195284	36608	20,91	dark image, some wrong pixels on the head, only with cropmask
99	216360	30949	13,15	dark image: clustering is not stable and might not find red: red fins, some red in head
100	255309	61895	20,17	dark image, unable to find binary mask for image №100, image 100b should be rotated before use: only with cropmask
101	251203	47637	19,64	red fins:use cropmask
102	193975	41726	19,72	small amount of wrong pixels in head,red fins: only with cropmask
103	244191	41712	16,57	wrong red pixels in the lips, red fins: use cropmask
104	190852	41710	20,98	red fins, small amount of redundant re don head: use cropmask

105	200558	35535	19,01	dark image, bad performance, wrong pixels in head and tail, only with cropmask
106	166021	33676	17,33	red fins, some red in head and tail: use cropmask
107	209674	35031	16,67	red fins: use cropmask
108	171894	38435	17,72	dark image, bad performance, lots wrong pixels in head an tail, only with cropmask
109	203660	34589	16,77	red fins
110	160020	34676	20,95	some redundant red in body, red hind fin: only with cropmask
111	197906	42281	21,01	poor binary mask, bad performance of clustering: red found in head and tail and fins: only with crop mask
112	137272	29163	17,49	dark image, bad performance: redundant pixels in head ant tail and fins: only with cropmask
113	156244	26912	17,42	red fins:use cropmask
114	149380	26211	18,27	red fins, some red in head: use cropmask
115	174364	39959	22,54	dark image, bad performance(head tail and fins are 'red"): only with cropmask
116				unable to find binary mask, dark image, bad performance of clustering: many wrong pixels in head: only with cropmask
117	236207	35950	15,58	dark image, bad performance of clustering, sometimes it is unable to recognize red: only with cropmask
118	257231	51393	17,75	dark image, bad performance of clustering, sometimes it is unable to recognize red: only with cropmask
119	155787	57335	-	dark image, bad performance of clustering, sometimes it is unable to recognize red: only with cropmask
120	162533	44152	-	dark image, bad performance of clustering, sometimes it is unable to recognize red: only with cropmask
121	394735	111577	24,47	redundancy in red cluster: fins, head, digits: use cropmask; in binary mask some wrong pixels
122	190723	32820	20,64	red fins:use cropmask
123				unable to find binary mask: fish is connected to the border
124	199276	40123	20,62	dark image, red in head and tail and fins: only with cropmask
125	160511	36810	23,40	dark imagered cluster conatins too much red: in the head and tail and body and fins: only with cropmask
126	216972	58271	25,42	some wrong red in the head, red fins: only cropmask
127	245225	62667	24,39	poor tail in binary mask, some wrong red head and in fins: use cropmask to get rid of them
128	176348	34004	16,49	dark image, red cluster contains redundant red pixels in head and tail, and also in fins: only with cropmask

129	151697	29843	18,39	dark image, small amount of redundant red in head and tail, red fins: use cropmask to get rid of them
130	205557	42421	20,02	small amount of redundant red pixels in head and fins: use cropmask
131	168788	32476	17,84	small amount of redundant red pixels in head and fins: use cropmask
132	212692	47547	22,26	some body red pixels, red fins: use cropmask
133	209641	42982	18,09	some red pixels in head and body, red fins: use cropmask
134	193705	38449	18,59	red fins: use cropmask
135	151275	29894	18,50	red hind fin: use cropmask to get rid of it
			19,29	←*gray cells are not included here*

APPENDIX III

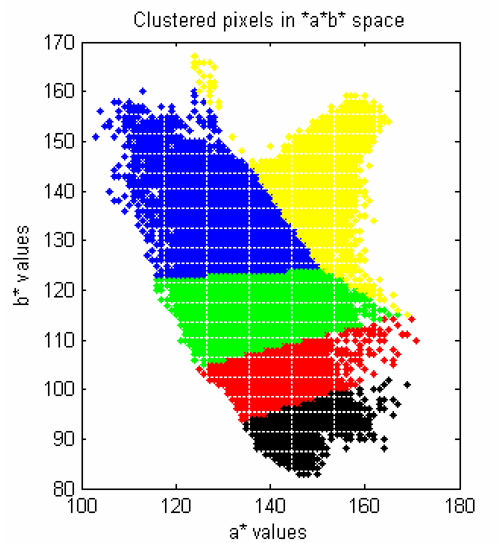


Figure 35: Clustering results in 2d (*a*b**) for Figure 11

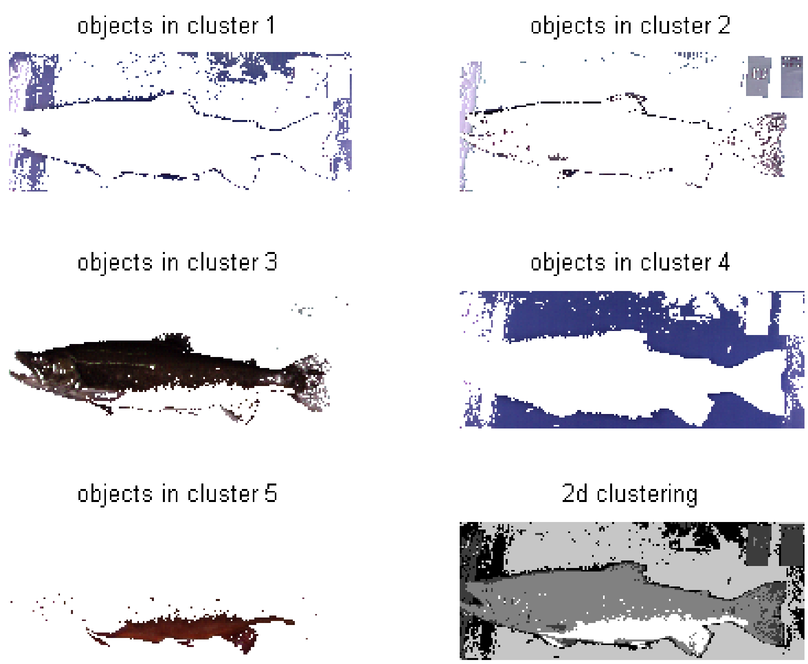


Figure 36: Interpreted results of 2d clustering

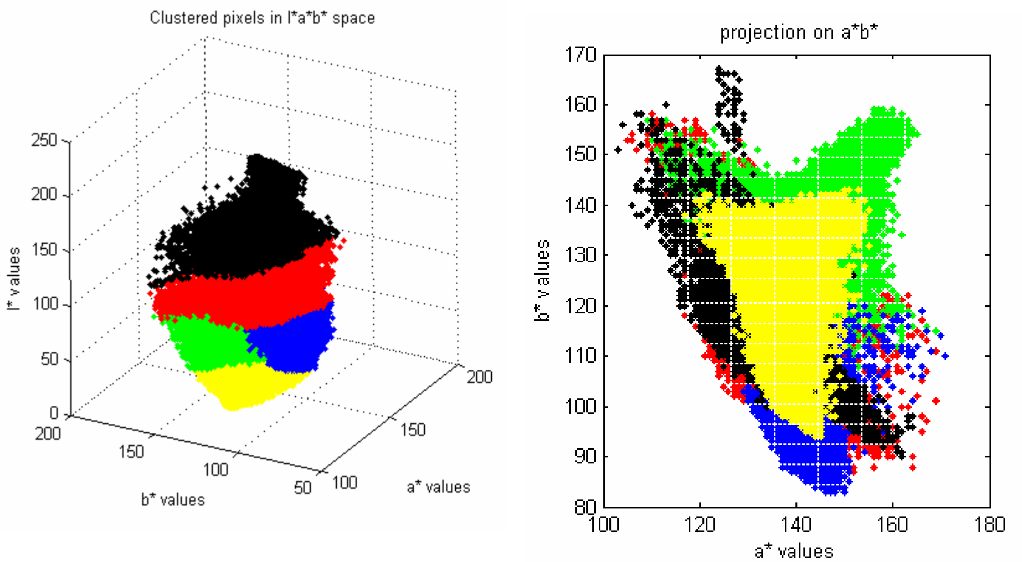


Figure 37: Clustering in 3d space. (to the left): clusters in 3d ($L^*a^*b^*$), (to the right): projection of them onto 2d (a^*b^*)

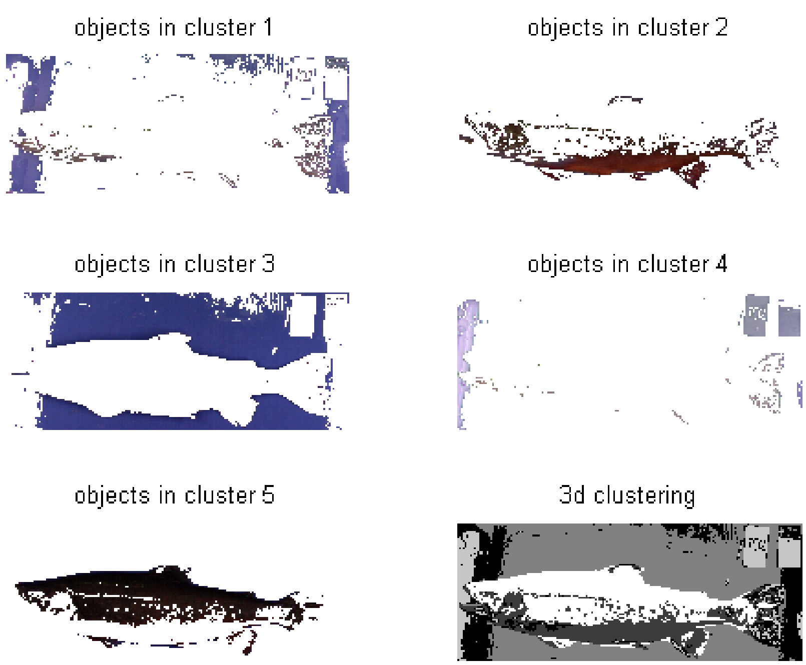


Figure 38: Interpreted results of 3d clustering