AUTOMATIC OBJECT EXTRACTION OVER MULTI-SCALE EDGE FIELD FOR MULTIMEDIA RETRIEVAL

Serkan Kiranyaz, Miguel Ferreira and Moncef Gabbouj

Abstract— **In this work we focus on automatic extraction of object boundaries from Canny edge field for the purpose of content-based indexing and retrieval over image and video databases. A multi-scale approach is adopted where each successive scale provides further simplification of the image by removing more details such as texture and noise, whilst keeping major edges. At each stage of the simplification, edges are extracted from the image and gathered in a *scale-map*, over which a perceptual *sub-segment* analysis is performed in order to extract true object boundaries. The analysis is mainly motivated by *Gestalt* laws and our experimental results suggest a promising performance for main objects extraction even for images with crowded textural edges and objects with color, texture and illumination variations. Finally, integrating the whole process as feature extraction module into MUVIS framework allows us to test the mutual performance of the proposed object extraction method and subsequent shape description in the context of multimedia indexing and retrieval. A promising retrieval performance is achieved, and especially on some particular examples, the experimental results show that the proposed method presents such a retrieval performance that cannot be achieved by using other features such as color or texture.**

Index Terms—**Canny edge detection, multi-scale analysis multimedia indexing and retrieval, sub-segment analysis, object extraction.**

I. INTRODUCTION

The shape of an object in a visual scene, if extracted properly, is a powerful descriptor for the purpose of content-based multimedia indexing and retrieval due to its semantic value. Especially in generic multimedia databases, achieving efficient shape descriptors requires an automatic and highly accurate segmentation operation. Various automatic segmentation efforts have been reported in the literature. Region-based methods [11], [3], [12], [10], [15] mainly depend on the assumption of uniformity and homogeneity of color (or luminance) and texture features within the boundaries of object segments. However, this assumption may obviously fail since a single object may exhibit several color and texture variations. Region merging/splitting techniques such as RSST (Recursive Shortest Spanning Tree) [18], [26], KMCC (K-means Connectivity Constraint) [16] and JSEG [4] also belong to this category. Another alternative is SRG (Seeded Region Growing) [1], which partitions the image into regions where each connected region corresponds to one of the seeds. It can be a quite promising method provided that the initial seeds are sufficiently accurate; however, this represents its main limitation that is how to select the most appropriate initial seeds? Thresholding-based methods [24] are based on the assumption that adjacent pixels, whose properties (color, texture, intensity, etc.) lie within a certain threshold, belong to the same segment. These techniques are usually adequate for binary images or images exhibiting "clean" edges; and their performance degrades drastically for images with blurred object boundaries since they neglect the spatial information. Furthermore, noise and object occlusion further complicate the segmentation task. Alternative approaches include boundary-based approaches [5], [7], [8], [13], [17], [20], [21] which are so far the most promising due to the perceptual evidence indicating that the human visual system (HVS) focuses on edges and tends to ignore uniform regions. This process, known as *Lateral Inhibition*, is similar to the mathematical *Laplacian* operation. One of the most promising boundary-based methods is a graph-based image segmentation (GBIS) proposed in [6].

In this paper we propose a systematic approach which performs automatic object extraction by *sub-segment* analysis over (object) boundaries in order to achieve visual content extraction. From basic visual features through low-level edge processing, we aim to achieve some form of mid-level meaningful description based on the segmentation information extracted from the edge field for the purpose of multimedia retrieval. The main drawback of the aforementioned segmentation algorithms is that they produce over- or under-segmented images, which are not useful for shape-based retrieval. Apart from the biological motivation based on HVS, the primary reason we propose an edge-based approach is that it brings generality, as edges are generally present across any type of visual content, colored or not, textured or not. Moreover, edges mostly carry the boundary information of the object shapes. However, if extracted directly from a natural image, these edges tend to be incomplete, missing, noisy or saturated. Background and texture edges, which are not related to object shape information, significantly degrade the extraction accuracy of the "true" object edges (boundaries). In brief, natural images are usually too "detailed" to achieve an accurate segmentation over the edge field. This is the main reason behind the proposed multi-scale approach where the *scale* basically represents the amount of simplification provided by iteratively applying Bilateral filter [25] on the image. The ultimate goal is the total removal of "details" so as to achieve a "cartoon-like" image on the higher scales where edge detection would eventually yield true object boundaries. In practice there is unfortunately no guaranteed method where all details will vanish and only true object edges prevail. Furthermore, a possible consequence of such a simplification step might as well be the removal of some important parts of the object edges, which makes any further analysis infeasible. Moreover, without a prior knowledge, it is not possible to favor any particular *scale* either. Thus the proposed approach uses the edge-information of all scaled (simplified in certain level) images in a *scale-map* of the edge field where all relevant information (i.e. the complete edge field as well as their relevance information) exists for further (*sub-segment*) analysis. Due to its robustness and better accuracy Canny edge detection [2] is applied to extract the edges; however, the proposed approach is independent from the choice of the edge detection scheme and hence any other edge detector can be conveniently used instead. In short, the main outcome of such a multi-scale approach is the classification of edge pixels with respect to their relevance so that the following *sub-segment* analysis can favor the more relevant edges in order to extract the true object boundaries.

Once the *scale-map* is obtained, certain analytical steps are performed in order to achieve the extraction of object boundaries. The edge pixels are first grouped into *sub-segments* that are then subject to further analysis to extract closed-loop (CL) and non-closed-loop (NCL) segments, through *sub-segment* linkage. The whole *sub-segment* analysis is mainly motivated by certain perceptual rules, known as *Gestalt* Laws, which were originated by the German psychologist Max Wertheimer during the early twentieth century [27]. The guiding principle of this theory is that the larger picture will be seen before its component parts. According to *Gestalt* Laws, the way our mind groups these parts to form a whole was described in a set of rules, which can be expressed by eight visual cues: *proximity, similarity, good continuation, closure, common fate, surroundedness, relative size* and *symmetry*. Henceforth, our *sub-segment* analysis is based on a regularization function formed according to local rules such as *proximity* and *good continuation* of edges and according to the global concern of finding objects, i.e. *closure*. CL segments represent the objects (their boundaries) or the major parts of the objects; whereas, NCL segments are for background scene and incomplete (overlapped or out-of-frame) objects.

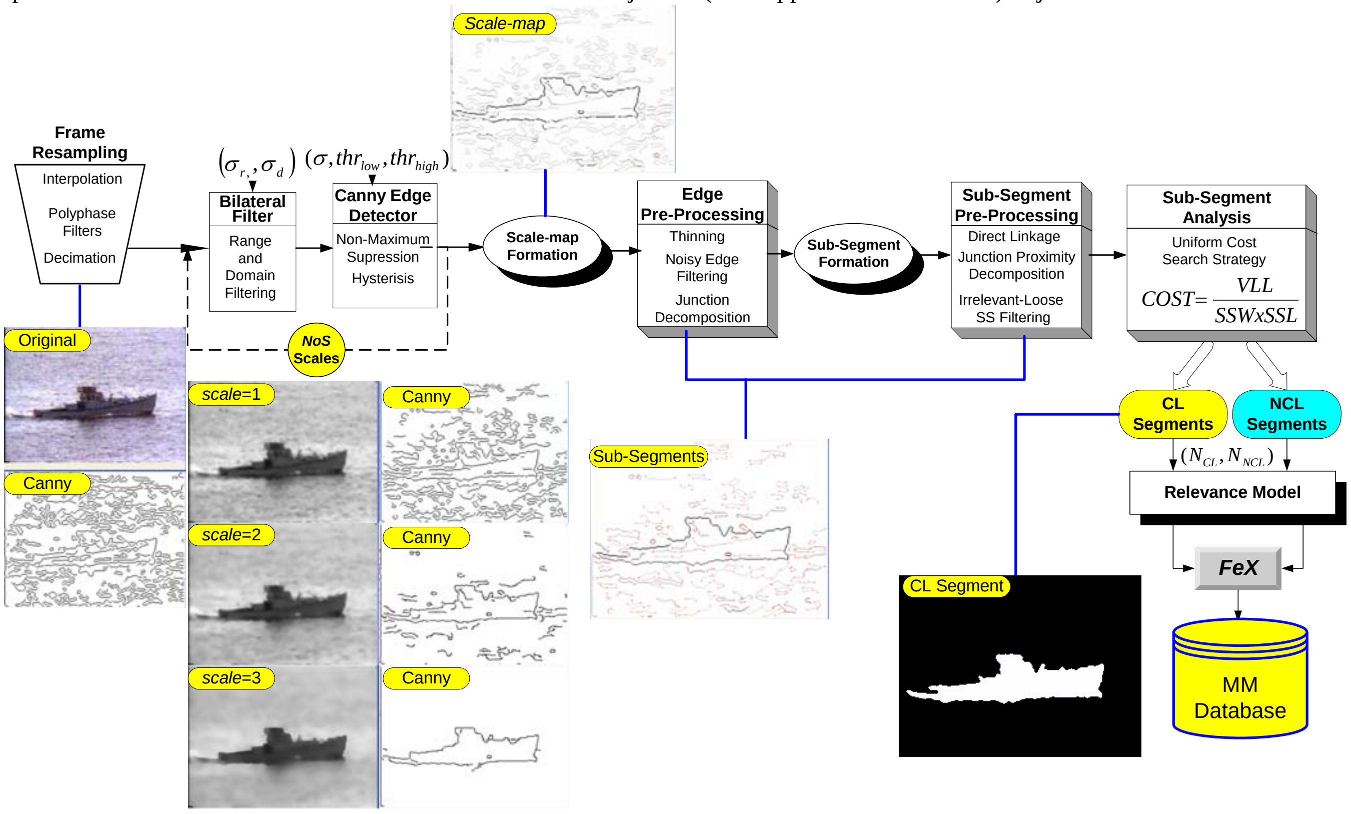


Figure 1: Overview of the proposed approach.

Finally, a shape-based Feature eXtraction (*FeX*) is performed over both types of segments in order to achieve an edge-based shape description of the whole image. This description will discriminate between both types of segments, by providing contour-based shape description for CL segments, and background information shape description for NCL segments. Forming the whole process as a *FeX* module into MUVIS (Multimedia Video Indexing System) [19], [14] allows us to test the overall performance of the proposed method in the context of multimedia indexing and retrieval. An overview of the proposed approach is illustrated in Figure 1.

The whole process is automatic (i.e. without any supervision, feedback or training involved) and designed with the minimal parameter dependency (i.e. the same set of parameters can be used indiscriminately for an indefinite number of images and we used a default set for all the examples in this paper). The rest of the paper is organized as follows: in Section II an overview about the entire CL/NCL through multi-scale sub-segment analysis is introduced. We discuss the implementation of the proposed method as a *FeX* module producing shape-based descriptors for a MUVIS multimedia database in Section III. Section IV presents the experimental results and Section V concludes the paper and suggests topics for future research.

II. Sub-Segment Analysis Over Edges

The proposed analysis is based upon the luminance component (intensity) of the frames extracted (decoded) either from the images or (key-) frames of the video clips in a host multimedia database. The method is designed to be robust against possible variations in frame size and image/video types. Figure 1 presents the fundamental steps in the CL/NCL segmentation phase, which can basically be divided into four major parts: Frame resampling, Bilateral filtering and *scale-map* formation over the Canny edge field, *sub-segment* formation and analysis and finally CL/NCL segment formation. The default parameters for Canny edge detector, Bilateral filter and the number of CL/NCL (outcome) segments are user-defined. Once the segmentation phase is completed, Feature eXtraction (*FeX*) is performed over the CL/NCL segments as will be detailed in the next section.

The first and natural step is resampling (frame size transformation if needed for too large or small frames) into a practical dimension range, which is sufficiently large for perceivable shapes but small enough not to cause infeasible analysis time. A typical size can be between 200 and 400 pixels for both dimensions. By this way a standard approach can now be applied for the quantitative measures performed within certain analytical steps. The resampled Y-frame can then be used for multi-scale analysis to form the *scale-map* of the image, as presented in the following sub-section. The *sub-segment* formation from the *scale-map* and CL/NCL segmentation will then be explained in Section II.B.

A. Multi-Scale Analysis over Scale-Map

Obviously, the most accurate edge field can be obtained from binary (black and white) or cartoon images where no details (texture, noise, illumination effects, etc.) are present, the edges are clean and therefore, true object boundaries can be extracted simply using an ordinary edge detection algorithm. Complications and degradations in the segmentation accuracy begin to occur when the image gets more and more "detailed" and this is usually the case for natural images. Therefore, in our multi-scale approach, the resampled image is iteratively simplified (or cartoonized) using a Bilateral filter [25] where the "scales" are emerged from the iterations of Bilateral filtering and represent successively simplified versions of the original image. As illustrated on a sample image in Figure 1, at each scale, Canny edge detection is applied over the (simplified) image (Bilateral) filtered by the *scale* number based on a naïve expectation that the majority of the perceptually relevant edges will prevail on the higher scales – yet some of the main object-boundary edges may also vanish along with the details. This is why we need the information across all the scales, since the loss of a few or even just one section of the object edge-boundary would make any further segmentation operation difficult –if not impossible. Once the required number of scales (*NoS* in Figure 1 is the iteration number, which can be set to any value sufficiently big enough, i.e. $NoS \geq 3$) are achieved then by assigning the maximum scale number to each pixel where it still prevails, the *scale-map* can be formed in the lowest scale (i.e. the first iteration, *scale*=1) edge field where all the edges (obtained from object boundaries, noise, textural details, etc.) are present.

1) Bilateral Filter

Bilateral filtering was first introduced in 1998 and it presents certain advantages over the traditional Gaussian (blur) filter such as it can smooth an image whilst preserving major edges. Let I_{in} and I_{out} be the input and output images, respectively. Then the Gaussian filter can be formulated as follows:

$$I_{out}(\vec{x}) = \int I_{in}(\vec{\varepsilon}) . \exp\left\{ -\frac{(\vec{x}-\vec{\varepsilon})^2}{2\sigma^2} \right\} d\vec{\varepsilon} \qquad (1)$$

where σ is the standard deviation. This is a linear filter, which is widely used for noise reduction. On the contrary, a Bilateral filter is a non-linear filter whose filtering coefficients depend on the local image pixel differences and are expressed in the form of two combined Gaussian filters: One in (2D) spatial domain and the other in the (intensity) range of the image. Let σ_d and σ_r be the domain and range standard deviation values, respectively. A Bilateral filter can then be expressed through the following equation:

$$I_{out}(\vec{x}) = \int I_{in}(\vec{\varepsilon}) . \exp\left\{ -\frac{(\vec{x}-\vec{\varepsilon})^2}{2\sigma_d^{\,2}} \right\} \exp\left\{ -\frac{(I_{in}(\vec{x}) - I_{in}(\vec{\varepsilon}))^2}{2\sigma_r^{\,2}} \right\} d\vec{\varepsilon} \qquad (2)$$

As $\sigma_r \to \infty$, the Bilateral filter approaches to a Gaussian. Figure 2 presents a Bilateral filtered sample image with several σ_d and σ_r values.

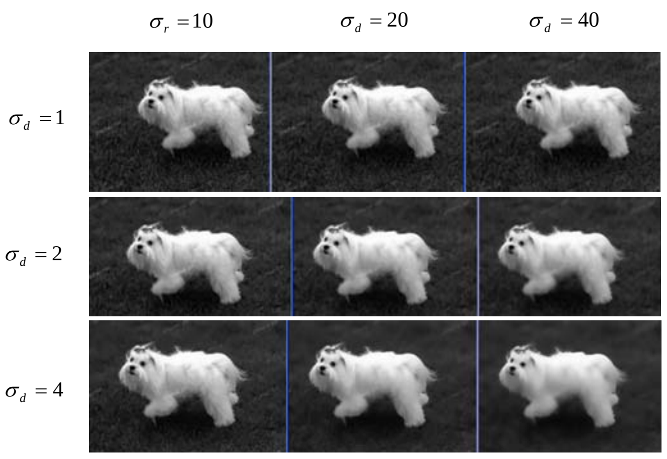


Figure 2: Bilateral filtered image with different σ_d and σ_r

It is clearly visible that textural details are significantly reduced with increasing values of σ_d and σ_r, yet major edges are preserved. However, a *Gaussian* filter while removing textural details smears (blurs) major edges and hence causes degradations in the edge detection accuracy and localization (the spatial position of edges). A comparative illustration in 3D is shown in Figures 3 and 4. As seen in Figure 4, the Bilateral filter is especially efficient for removing details when used iteratively. Each iteration further removes details from the image until eventually reaching a cartoon-like image. Since strong edges remain with good localization precision, this information is easily gathered in an edge-pixel *scale-map* and the *scale* factor for each pixel (until what *scale* is the edge present for each particular location) can then be used in subsequent analysis.

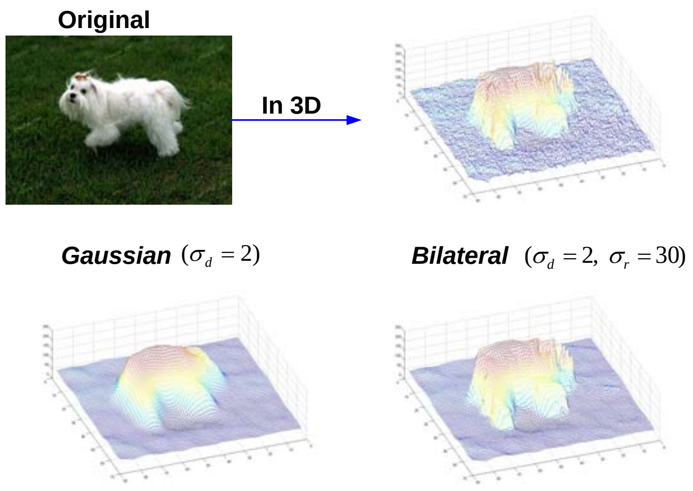

Original
In 3D
Gaussian $(\sigma_d = 2)$
Bilateral $(\sigma_d = 2,\ \sigma_r = 30)$

Figure 3: *Gaussian* vs. Bilateral filtering in 3D.

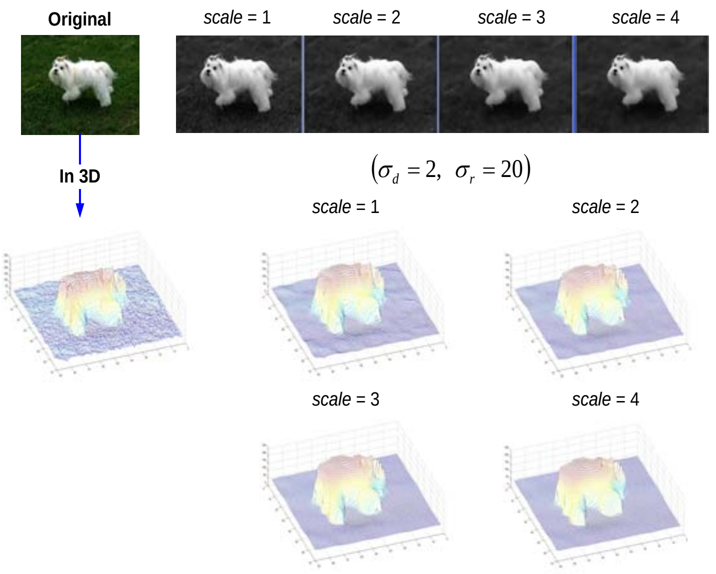


Figure 4: Iterative (multi-scale) Bilateral filtering.

2) Canny Edge Detection

John Canny developed an algorithm for edge detection, which has been widely used since its publication in 1986 [2]. Canny started from postulating three performance criteria for the edge detector:

- The detector should have a low probability of failing to mark real edge points or falsely marking non-edge points (*good detection*).
- The points marked as edges by the detector should be as close as possible to the centre of the true edges (*good localization*).
- There should be only one response to a single edge (*single response*).

In his work, Canny proceeded to search for optimal operators, which would exhibit the above mentioned properties. Considering only the first two criteria, the optimal detector for step edges in a 1D signal was found to be the truncated step, or difference of boxes operator. However, due to the large bandwidth of this operator, the output to a noisy step edge exhibits many maxima which make it difficult to choose one as the wanted response. Considering also the third constraint, Canny tried to minimize the number of response "peaks" in the vicinity of the step edge. By a numerical optimization method the formula of the desired operator was derived. An approximation to this optimal operator is the first derivative of a *Gaussian*, which is used for its computational efficiency. The optimized detector was first derived in 1D but is easily generalized to any dimension, notably for 2D space of image pixel positions. In the 2D case, edges have not only intensity, but also orientation and the detector will be the first derivative of a 2D Gaussian (*DoG*).

After applying *DoG* operator to an image, thinning is performed over the result by the Canny edge algorithm, in the regions of high amplitude response, since the resulting output is usually broader than one pixel. A method known as *non-maximum suppression* is used for that purpose.

Thresholds are set by the algorithm to decide which pixels will be marked as "edges". These thresholds depend on a statistical analysis of the response magnitudes such as the estimated noise energy present in the image. Canny considered that two thresholds should be used to prevent streaking, which occurs when noise causes the output along an edge to fluctuate, above and below a single threshold. This results in the breaking of continuous edges. A high threshold is set according to the noise energy estimate, and a low threshold is set as a fraction of the high threshold (this fraction is in the range of two or three to one according to Canny). Pixels are marked as edges automatically if their response amplitude is above the high threshold, but if their amplitude is between the high and low thresholds, they are only considered as edge pixels if they are in the 8-neighbourhood of another pixel already marked as an edge pixel. This so called *hysteresis* process is used to follow strong edges. Furthermore, the probability of isolated false edge points is reduced, since the resulting *DoG* gradient strength at these points must be above the high threshold.

Canny edge detection is applied for each scale (after Bilateral filtering). The default Canny parameters, $\sigma_d, thr_{low}, thr_{high}$, can be kept fixed and have minimal effect due to the presence of multi-scale information collected in the *scale-map*, the formation of which is explained next.

3) Scale-Map Formation

Recall from the earlier discussion that for a particular *scale*, Bilateral filter is first applied over the image from previous iteration (*scale*-1) and then a new edge field is extracted.

Since iterative application of Bilateral filter has a desired "cartoon" affect, which mainly preserves the object boundary edges whilst removing all the other details, the edge population tends to decrease drastically for higher *scales* (e.g. see Figure 5). The remaining edges in higher *scales*, however, have more relevance and priority for the object extraction purpose –yet lower *scales* may still contain some important edges that are lost in a higher scale and in this aspect the lowest scale (*scale*=1) has the utmost completeness. Therefore, the *scale-map* is formed over the edge field of the lowest *scale* by assigning each edge pixel to the maximum *scale* at which it prevails. Two typical examples are shown in Figure 1 and Figure 5. In this way all the crucial information, particularly the localization and *scale* factor of each constituent edge pixel, can be embedded into the *scale-map*, which becomes the primary input for further analysis. After the formation of *sub-segments*, the proposed object extraction method can rely on the *scale* information during *searching*, *tracing* and *linking* the relevant *sub-segments* formed from the *scale-map*.

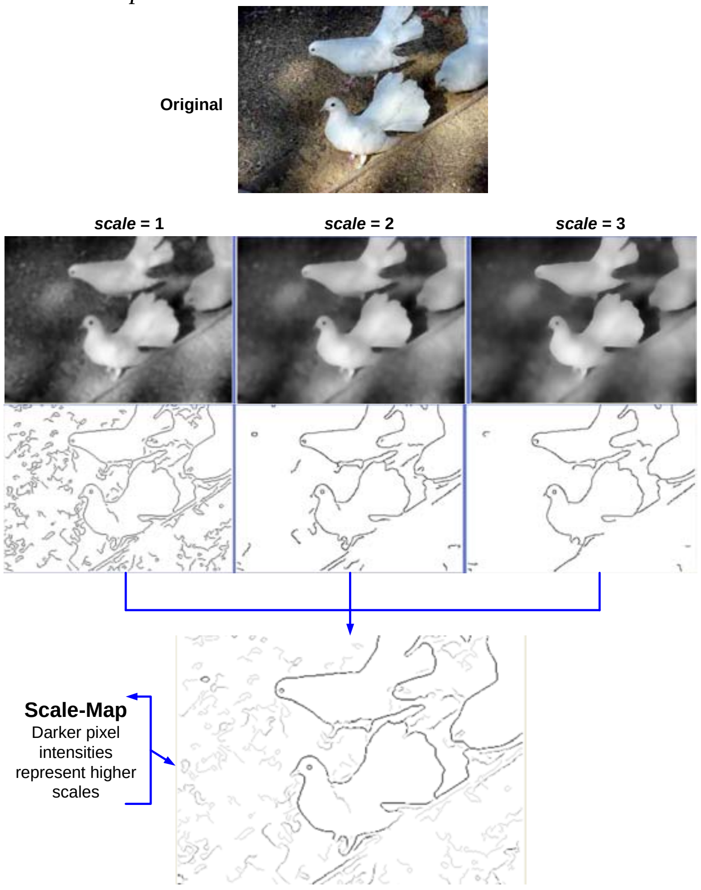


Figure 5: A sample scale-map formation.

4) Sub-Segment Formation

An interesting issue raised is how to choose the starting elements for edge analysis towards grouping. Pixel-level analysis does not lead to a clear identification of image structures, since human visual system (HVS) does not perceive individual pixels separately. When looking at an edge image, how exactly does HVS perception combine edge pixels together, to form natural shape outlines (i.e. object boundaries)? The first low-level structures that HVS can clearly perceive are continuous and connected series of pixels. However, abrupt direction changes in a connected series, can lead to the impression of two different edge sections, which are considered separately in mental grouping. We use these perceptual facts to build such basic elements that will be the basis for further analysis. These elements are so-called *sub-segments*, through which both forms (CL and NCL) of segmentation are achieved. A *sub-segment* can basically be defined as an ordered series of connected edge pixels with two *end-points* and they are formed using a tracing algorithm, which performs the following steps over the *scale-map*:

- *Thinning*: *Non-maximum suppression* in Canny's algorithm, although providing usually thin edges, does not guarantee that the resulting edges are one pixel thick as desired. Therefore, thinning is performed beforehand, using simple sliding window operations. Afterwards the tracing operation re-commences for the rest of the steps.
- *"Noisy Edge" Removal*: If the number of connected edge pixels is below a certain threshold (e.g. 5 pixels), they are assumed to be edge detection noise for the sake of accuracy and speed; therefore, such edges are removed from the *scale-map*.
- *Junction Decomposition*: In order to achieve a better segmentation from the *sub-segment* analysis, junctions need to be decomposed into their branches (separated *sub-segments*). For that, an algorithm breaks the tracing procedure at a certain edge pixel location where there is more than one valid branch to follow. This pixel location is considered as an *end-point* for the traced *sub-segment*.
- *CL Segment Detection*: During *sub-segment* tracing, whenever it returns back to the initial *end-point*, thus creating a loop-back, the respective *sub-segment* is assigned as a CL segment or simply as a potential "ready" *object* where no further analysis is needed. Otherwise, the tracing eventually breaks when no more connected pixels exist, and a second *end-point* is assigned to complete the formation of the *sub-segment*.
- A sample *sub-segment* formation is shown in Figure 6.

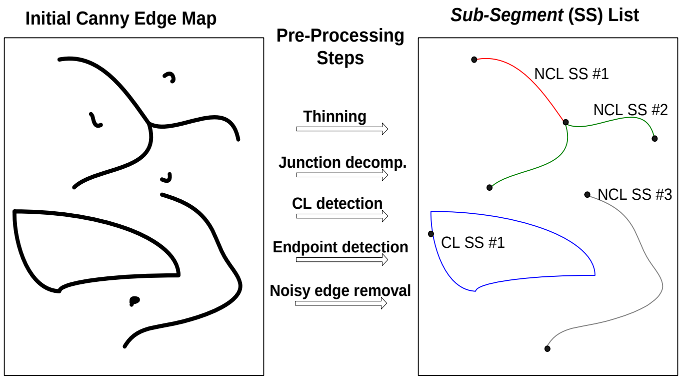


Figure 6: Sub-segment formation from an initial Canny edge field.

Once all the *sub-segments* are formed from the edge pixels of the *scale-map*, a *scale* weight, *W*(*SS*), of each *sub-segment*, *SS*, is then calculated as in (3).

$$W(SS) = \frac{\sum_{e \in SS} s(e)}{L(SS)} \tag{3}$$

where *s(e)* is the scale factor of an edge pixel *e*, in *SS* and *L(SS)* is the total length of *SS*. The weight and length of a particular *sub-segment* both signify its relevance and thus, the list of sub-segments formed can now be used conveniently in the following *sub-segment* analysis to form CL segments.

B. CL and NCL Segmentation from Sub-Segments

Both types of segmentations (CL and NCL) are performed over the *sub-segment* list extracted in the previous step. Due to space limitations NCL segmentation is not described in this paper and focus is drawn particularly on CL segmentation that is basically the extraction of the primary object boundaries in the image. The purpose of the aforementioned *sub-segment* formation is twofold: First, a generic analysis can now be performed over thinned, decomposed and well-formed *sub-segments* instead of individual edge pixels. In this way a better modeling of Gestalt laws can be adopted (i.e. *proximity*, *relative size* and *good continuation*) for linking sub-segments to form CL segments (i.e. *closure*). Second, by means of the sub-segment formation with two *end-points*, CL segmentation problem can be converted into a search process in a state-space where the *end-points* are assigned as "states" and the aim is to visit (link) certain number of states to loop back to the initial state with the minimum Gestalt cost. Therefore, the search process links one or more individual *sub-segment(s)* using an abstract linkage element, so-called *virtual link*, by minimizing a *cost* function formed according to Gestalt laws. A *virtual link* is defined as the virtual connection between two states (*end-points*) and it is performed "virtually" during the search and "artificially" for the *sub-segments* belonging to a CL segment. Note that in any CL segment, there can be one or more *virtual links*, which connects the "gaps" where the edge detection algorithm fails to link in the first place. According to *proximity* law, a *virtual link* threshold (thr_{VLL}) can be defined as the maximum possible linkage that can be established between two *end-points*, i.e. e_1 and e_2. Therefore, a *virtual link*, $VL(e_1, e_2)$, is only considered if its length $VLL(e_1, e_2)$ is below thr_{VLL}, i.e. $VLL(e_1, e_2) < thr_{VLL}$. Its purpose is twofold: it serves as a quantitative level of tolerance for *proximity* and it is also used to reduce the number of state transitions, so as to make the search process more tractable. It is however, fairly difficult to propose a precise measure for thr_{VLL} since *proximity* law is based on human psychology and HVS (perception). So it can further depend on several factors such as the content in the image (i.e. the size of the objects in the scenery, their relative distances, etc.), image (frame) size, etc. thr_{VLL} should be big enough to link the gaps due to the deficiency of the edge detector (e.g. see Figure 7) but not too big to cause infeasible search time. Since there is usually a close relation with object sizes and their basic elements, i.e. *sub-segments*, we set thr_{VLL} according to first order statistics obtained from the *sub-segment* list, as in (4):

$$thr_{VLL} = k_0 \frac{\sum_{SS} L(SS)}{N} = k_0 \mu_L^{SS} \tag{4}$$

where k_0 is an empirical coefficient, *N* is the total number of sub-segments formed. According to an extended set of experimental results, $0.3 < k_0 < 0.6$ is a reasonable range for k_0 and note that thr_{VLL} is thus set to a fraction of the average *sub-segment* length, μ_L^{SS}. In this way, only small (virtual) links will be possible in an image with comparatively smaller, texture-like or noisy edges (and *sub-segments*), as intended.

In order to improve the accuracy and speed of the search process for CL segmentation over *sub-segments*, two consecutive processes are first performed: pre-processing over *sub-segments* and then filtering irrelevant *sub-segments*.

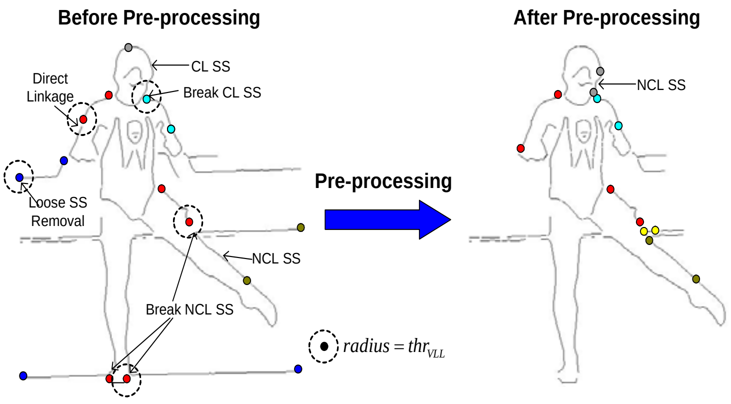


Figure 7: Pre-processing steps applied on a sample sub-segment field.

1) Pre-Processing over Sub-Segments

The pre-processing process can be split in three separate phases:

- *Direct Linkage*: two *sub-segments* with particular *end-points* (e_1, e_2), which can only be linked with each other (satisfying $VLL(e_1, e_2) < thr_{VLL}$) can be immediately linked. A *virtual link*, $VL(e_1, e_2)$, between both *end-points* is created as a straight line with one pixel width and both *sub-segments* are merged into one.
- *Junction Proximity Decomposition*: Canny edge detector has a particular flaw in dealing with "T-junctions", where three different homogeneous regions meet. What the detector frequently does is to output the "T-junction" as a single branch for two of its branches, leaving the third branch of the junction separated by a small gap. Once the *sub-segments* are formed from edge field (*scale-map*), the branches become *sub-segments* where one of them has an *end-point* in the close vicinity of the other two merged. The search process needs to trace each branch separately; therefore, we decompose the merged *sub-segment* into two from the closest point to the *end-point* of the third one.
- *Loose Sub-Segment Removal*: In case a particular *sub-segment* has an *end-point* that cannot be linked to any other *end-point* within the *virtual link* threshold, that *sub-*

segment cannot be a part of a CL segment. In order to reduce the number of states that the search algorithm will have to process, such "loose" sub-segments are removed beforehand.

Figure 7 illustrates typical examples of the each pre-processing step over a sample *sub-segment* field.

2) Filtering the Least Relevant Sub-Segments

Upon completion of the pre-processing steps, the least relevant *sub-segments* are filtered out in order to reduce their noisy disturbance and to further speed up the search operation leading to the final CL segmentation. As mentioned earlier the most relevant *sub-segments*, which bear the major object edges are usually longer with higher scale weight values and the opposite is true for the least relevant ones (i.e. from textural details, noise, etc.). Therefore, the relevance, *R,* of a *sub-segment SS*, can then be expressed as follows:

$$R(SS) = W(SS) \times L(SS) \tag{5}$$

where $W(SS)$ is the scale weight and $L(SS)$ is the length (total number of edge pixels) of *SS*. Sorting all the *sub-segments* formed over the *scale-map* and removing the least relevant ones -or equivalently choosing an overestimated number (i.e. $N \geq 100$) of the most relevant *sub-segments* completes this filtering stage. The effect of this step and due segmentation result can be clearly seen in the example image shown in Figure 8.

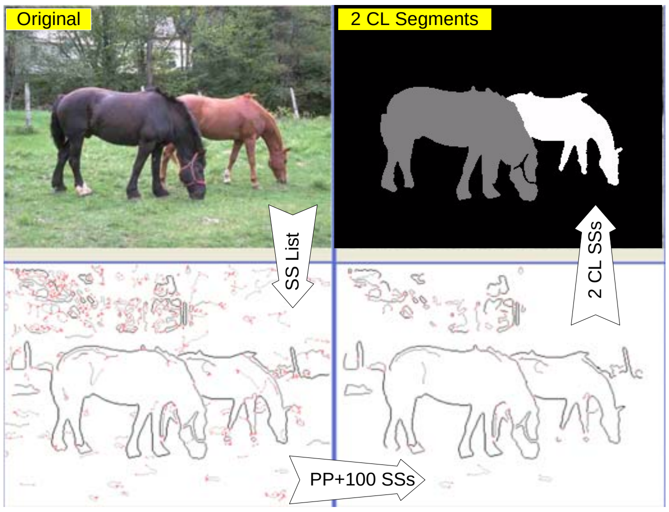


Figure 8: Pre-processing and irrelevant *sub-segment* filtering applied (*N*=100) on a sample image.

3) CL Segmentation via Sub-Segment Analysis

Having formed, pre-processed and filtered the initial *sub-segments*, the core of our method lies in the perceptual linking of the final *sub-segments*, trying to achieve a natural shape closure. Recall that the final *sub-segment* list may contain both types of *sub-segments*: CL and NCL. Such CL *sub-segments*, which are extracted directly from the edge *scale-map* and survive the *sub-segment* pre-processing and filtering stages, or that are formed by (direct) linkage during the same stage, represent stable and "ready" objects. Thus, among NCL *sub-segments*, the aim is to search for objects (CL segments) by successive linking operations (via *virtual links*) whenever possible. As mentioned earlier, four of Gestalt laws, *proximity, good continuation, relative size* and *closure*, are used in the model in order to achieve perceptually meaningful CL segments.

Consider *N sub-segments*, with two *end-points* each, the objective is to form N_{CL} objects where N_{CL} is the desired number of CL segments (objects). A CL segment, S_{CL}, consisting of l *sub-segments* (SS_i) can be defined as:

$$S_{CL} = \sum_{i=0}^{l} SS_i \;\middle|\; SS_0 = SS_l \tag{6}$$

We define a state space, which consists of *2N* end-points. Each state is represented by its respective *end-point* coordinates and the segment identification number. By choosing one of the *end-points*, of a particular *sub-segment* as being the *start state*, we automatically set the *end state* as its other *end-point*. A search algorithm, from one state (e_1) to another (e_2), algorithmically traces all possible state transitions by performing *virtual links* (if $VLL(e_1, e_2) < thr_{VLL}$) to end up in the *end-state*. In addition to *virtual link* length, the length and *scale* weight of the traced *sub-segment* are used to calculate the transition cost, which is modelled based on the aforementioned Gestalt laws. One such virtual connections between two *end-points* (states) is shown in Figure 9.

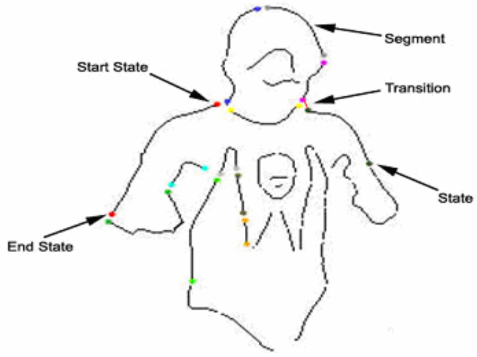


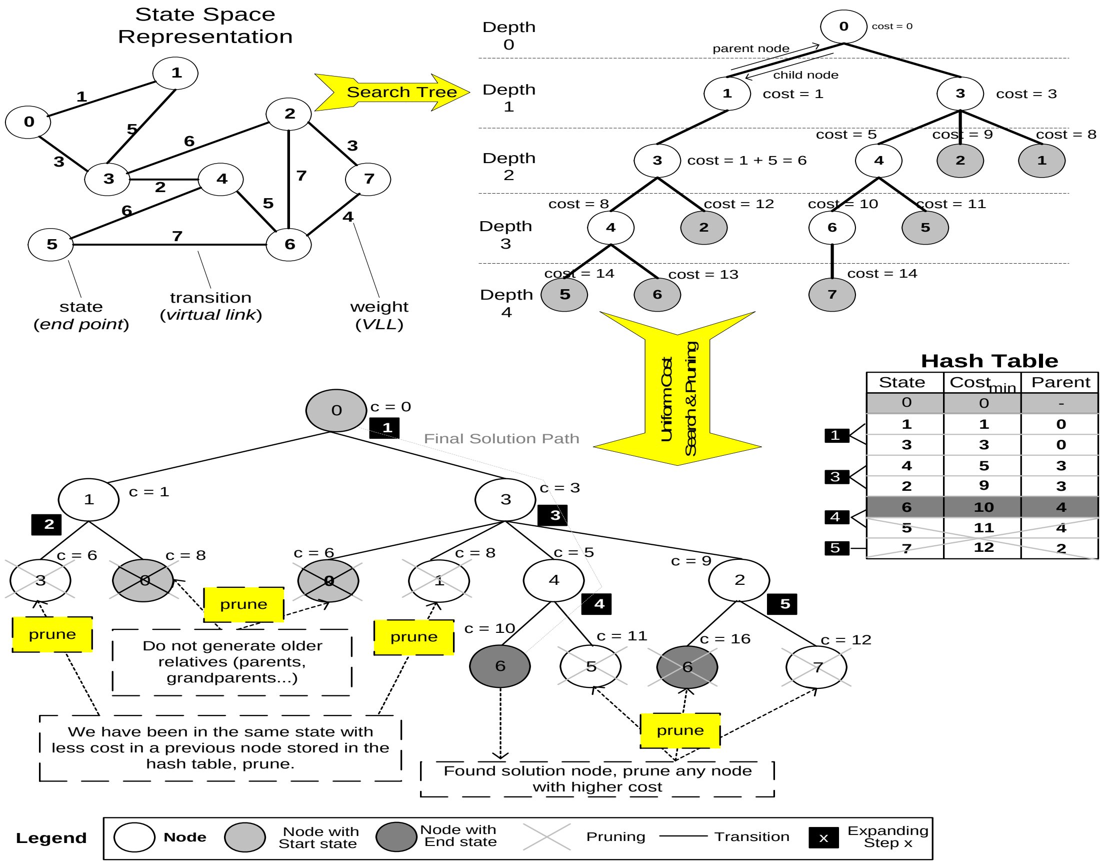
Figure 9: State space for a given sub-segment layout.

It becomes clear that this particular problem fits well in a graph-based search approach. In the general case, there can be multiple paths, which lead to the solution (*end state*). Note that the links are reversible, i.e. valid for both directions. In the search algorithm, all data regarding a particular state of the problem is collected in a structure called a node. The *virtual links* are then considered as the operators for node expansion as they supply the transition network of the state space. The search is repeated for each *sub-segment* in the list and the (selected) final path closing the initial *sub-segment*, in case multiple paths exist, will be the one with the least cumulative cost. The transition cost function ($C(e_1, e_2)$), to jump from an end-point e_1 belonging to *sub-segment* s_1 to an end-point e_2 in *sub-segment* s_2 can be expressed by:

$$\boxed{C(e1, e2) = \frac{VLL(e_1, e_2)}{SSW(s_2) \times SSL(s_2)}} \tag{7}$$

where *VLL* is the *virtual link* length, *SSL* is the *sub-segment* length and *SSW* is the *sub-segment scale* weight operators. Obviously, this *cost* function is formed to favour shorter transitions (*proximity*) to a neighbouring *end-point* belonging to longer *sub-segments* (*relative size*) with higher *scale* weight (*good continuation*). Arriving to an *end-point* of a *sub-segment* and leaving from the other *end-point* with a new transition, this cost cumulates from one transition to the next, and the trace (and search) terminates when the *end-state* is reached, providing eventually a CL segment (*closure*).

As for the search method, uniform cost search [23] is used for its completeness and optimality but also for its low time complexity, since it generally converges faster to the optimal path, by expanding least cost nodes first. Once the method has found a solution node, it uses the solution path cost to prune any node with a cost exceeding it. Additional techniques were implemented to further reduce time and space complexities, taking into consideration that this problem belongs to the "route-finding" class of problems, which have reversible operators and repetition of states. By using a hash table, storing all generated least-cost nodes for each state, node pruning can start before reaching a solution state, which results in a significant reduction in search time. Nodes in the hash table will be either "closed" or "open", whether they have been, or have yet to be expanded, with "open" nodes forming the fringe. By consulting the hash table, with all visited states stored in the form of their least-cost node, entire path sub-trees can be avoided by not going through the same state again, which has been visited with lower cost along a different path. However, when a new node is generated, which contains the same state information and exhibits a lower-cost than a previously stored node then the corresponding entry in the hash table is updated to the new node. Since the new node has its status still set to "open", its subsequent descendant nodes sub-tree is regenerated with lower cost values which will eventually update more nodes in the hash-table.



State	$Cost_{min}$	Parent
0	0	-
1	1	0
3	3	0
4	5	3
2	9	3
6	10	4
5	11	4
7	12	2

Figure 10: Uniform cost search applied on a sample state space, as state 0 as the initial state and state 6 as the target state.

Such node expansions, applied to an example state space are highlighted by black boxes in Figure 10, which also shows the overall pruning operations performed. In this figure, states 0 and 6 are the *start* and *end states*, respectively. Note that in the expanding step 2, the node containing state 3 with cost 6 is pruned, along with its sub-tree, since a node with the same state but lower cost ($c = 3$) had been found and kept in the hash table in step 1. Another immediate pruning applied is to avoid repeated states along a path, thus not generating any older relatives as child nodes, as can be seen in steps 2 and 3. Once the solution state is found for a given node (in the case of the figure, after the expanding step 4) its cost value ($c = 10$) can also be used for pruning any node with a higher cost, i.e. after step 4 and 5 where both generated child nodes are above the solution cost that is found earlier. The optimal solution is found when all the nodes in the fringe have a higher cost value than the current solution node, which happens in this case after only 5 steps. After the completion of the search algorithm, one can trace the solution path from the hash table by starting in the node containing the *end-state* and tracing back each successive parent node, which lies in the path with minimum (cumulative) cost, until the initial node with *start-state* is reached. For each *sub-segment*, a search tree in the form of a hash table, similar to the one in Figure 10, is built each time, and the optimal (the least cost) path is extracted after the aforementioned pruning operations are applied.

4) The Relevance Model: Choosing the Right Objects

Upon completion of the uniform *cost* search, the extracted CL segments are inserted in a list, which might contain some "ready" CL segments extracted directly from the *scale-map* (i.e. the "ready" *objects*). A relevance model is now needed to choose N_{CL} segments among all (ready and extracted via uniform cost search) CL segments in the list. The proposed model depends on the maximization of the following $R(S)$ (relevance) function:

$$\boxed{R(S)=\frac{SSW_{\Sigma}^{S}\times SSL_{\Sigma}^{S}}{1+\sqrt{C_{\Sigma}^{S}}}} \tag{8}$$

where SSW_{Σ}^{S} is the total weight, SSL_{Σ}^{S} is the total length and C_{Σ}^{S} is the total *cost* of a CL segment *S*, respectively. This balanced model obviously favours the CL segments exhibiting a low cumulative (Gestalt) cost C_{Σ}^{S}, with long and highly relevant (having higher *scale* weights) *sub-segments*. As a result, N_{CL} segments with the highest $R(S)$ values are the CL objects extracted and used for the feature extraction, as explained in the following section.

III. Indexing and Retrieval

For a MUVIS multimedia database [19], the indexing process involves Feature eXtraction (*FeX*) operations applied onto visual items such as key-frames of the video clips or frame buffer of the images. A particular *FeX* module [9] formed from the proposed algorithm needs to provide both feature extraction of visual frames, and a similarity distance function for retrieval. The default *FeX* module parameters such as Canny edge detection and Bilateral filter parameters, number of CL/NCL segments (N_{CL}, N_{NCL}) to be extracted, the feature vector dimension, etc., can be set by the user depending on the database content and type. In the next sub-section we will present the feature extraction process from CL segments and due to space limitations feature extraction from NCL segments is not described in this paper.

A. Feature Extraction from CL Segments

The histogram of normalised angular first moment calculated from the centre of mass (*CoM*) of a CL segment is used as the main feature vector. From *CoM* the contour points are divided according to K angular partitions each of which has an angular resolution $\Delta\alpha$ as follows:

$$\Delta\alpha=\frac{2\pi}{K} \tag{9}$$

K dimensional, unit length feature vector, *FV*, can then be formed from the discrete angular moments as follows:

$$FV(k)=\frac{\int_{k\Delta\alpha}^{(k+1)\Delta\alpha} d(\alpha).l(\alpha).d\alpha}{\int_{0}^{2\pi} d(\alpha).l(\alpha).d\alpha}\cong\frac{\sum_{\alpha=k\Delta\alpha}^{(k+1)\Delta\alpha} d(\alpha)}{\sum_{\alpha=0}^{2\pi} d(\alpha)},\quad k=0..K \tag{10}$$

where $d(\alpha)$ represents the distance from the *CoM* to the boundary at angle α. The formation of *FV* for a sample CL segment can be visualised as in Figure 11.

Apart from unit normalization, this feature vector is further designed to be translation, rotation and size invariant. Translation invariance comes naturally as a result of using the *CoM* as the reference point. Size (area) invariance and unit normalization are achieved by dividing each bin value in the histogram by the area of the CL segment. Rotation invariance is achieved during the similarity distance calculation of the retrieval phase and will be described in the next sub-section.

Once all feature vectors for each CL segment are calculated, they are indexed into a descriptor file for that particular frame (image or key-frame) and appended into the MUVIS database structure.

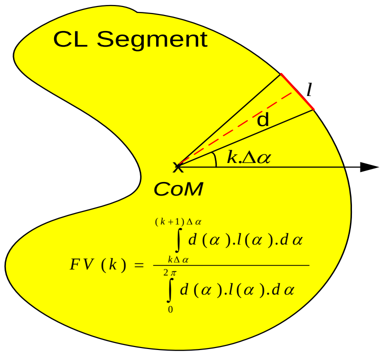


Figure 11: Feature Vector formation from 1st moment inserted in kth bin of the N-bins angle histogram.

B. Retrieval for CL Segments

The retrieval process in MUVIS is based on the traditional query by example (QBE) operation. The features of the query item are used for (dis-) similarity measurement among all the features of the visual items in the database. Ranking the database items according to their similarity distances yields the retrieval result.

Between the query frame and a particular frame in the database, the feature vectors of all (N_{CL}) segments are used for similarity distance calculation with the following matching criteria: for each CL segment in the query frame, a "matching" CL segment in the compared frame is found. The *Euclidean* metric based minimum similarity distance is used as the matching criterion. Afterwards the total similarity distance will be the sum of matching similarity distances of all segments present in the query frame.

In order to achieve rotation invariance, the feature vector of a CL segment (angular histogram bins) is shifted one bin at a time with the other vector kept fixed and the similarity distance is calculated per step. The particular shift, which gives minimum similarity distance, is chosen for that particular CL segment of the query frame. Since this sliding operation on the histogram bins basically represents the feature vector of the rotated CL segment, rotation invariance can therefore be achieved.

IV. Experimental Results

The experiments are performed to test the object extraction efficiency with respect to HVS perceptive criteria (subjective test) and the retrieval performance via query-by-example (QBE) within MUVIS multimedia databases indexed by the proposed *FeX* module. All experiments are carried out on a Pentium-5 1.8 GHz computer with 1024 MB memory. Images used in this section are gathered from several sources including Corel database and personal image archives. The following Canny and Bilateral filter parameters are used for all the experiments in this section: $\sigma = 1.5, thr_{low} = 0.4, thr_{high} = 0.8$ for Canny edge detector and $\sigma_d = 2, \sigma_r = 20$ for Bilateral filter. The rest of the parameters are as follows: $k_0 = 0.4$ for thr_{VLL}, $NoS = 5$ for scale number (number of iterations) and we used *N*=100, as the number of relevant *sub-segments*.

The section is organized as follows: Subsection *A* presents a comparative performance evaluation of the proposed object extraction algorithm with respect to three segmentation methods that were proposed recently; while, Subsection 0 evaluates the shape-based retrieval performance via *ground truth* method whilst providing both visual and numerical results.

A. Visual Evaluation of Object Extraction Results

In Figure 1 the key role of Bilateral filter within the proposed multi-scale approach is clearly visible on an image where strong textural details are present in the background. Such a “cartoon” effect can also be seen in Figure 12 where in this case the object has a significant textural overlay. In this figure, although Bilateral filter is applied once in the first scale, the Canny edge field at this scale is still too detailed. The second and particularly the final scales exhibit a clear enough edge field; however, some parts of the object edges (e.g. the mouth of the fish) are also lost. This particular example justifies the need of the multi-scale approach, where the analysis should be performed in the lowest scale due to its completeness, but should also use higher scales to favor particular *sub-segments* showing higher relevance.

For a comparative evaluation, we implemented three segmentation algorithms recently proposed, K-means Connectivity Constraint (KMCC) [16], Graph-Based Image Segmentation (as we refer to GBIS) [6], and JSEG [4], each of which shows promising results in terms of segmentation accuracy. Note that the proposed method is not a segmentation algorithm, rather an object extraction method and there are major differences between the two; segmentation is a process of pixel classification and clustering into sub-regions, i.e. *segments*, whereas object extraction, as its name implies, is the act of finding boundaries of object(s) in an image. Therefore, it can be used as a segmentation algorithm by defining each object as a *segment* and the background as another, but the reverse is obviously not true, i.e. any segmentation algorithm cannot distinguish between the objects or background –instead they form some segments showing similarities in terms of color, texture, etc. For this reason we tuned the parameters of these segmentation algorithms in order to obtain the highest segmentation accuracy possible for the set of images used in this section. For example, we set the crucial parameter, maximum segment number, as 4 for all three algorithms, in order to reduce over-segmentation since all images contain 3 or less objects. Figure 13 and Figure 14 show typical natural images in the left column where the proposed method’s outcome, the boundaries of N_{CL} object(s) are used to create the object segments as given in the 2nd column using a flooding method and the rest of the three columns belong to the segmentation masks obtained from three algorithms, KMCC, GBIS and JSEG, respectively. Since significant textural components are present as well as significant variations or non-uniformity properties in color, intensity and texture within the example images, it is not surprising to see that the segmentation methods usually fail to extract true object boundaries and over- and under-segmentation occur as a result of this. Only in some examples where there is a certain degree of uniformity and homogeneity in color/texture distribution (e.g. (c, f, i) in Figure 13 and (b, i, j, k) in Figure 14) reasonable segmentation results can only be obtained from some of them. Among these particular examples, since the proposed method performs analysis over *sub-segments* with one-pixel thick, note also that a higher segmentation resolution and accuracy is thus achieved on the extraction of the “true” object boundaries compared to the existing segmentation algorithms.

As some typical examples shown in Figure 15, slight segmentation errors can also be encountered, such as one or few tiny object parts being missed or slight background disturbance over object boundaries, yet the majority of the object shape is still intact.

The basic assumption the proposed algorithm relies on is that the lowest-scale (Canny) edge detector captures most edges in the image, including the boundary edges of the objects of interest. As explained earlier, the tolerance here is modeled by *virtual link* length (*VLL*), which virtually connects the “gaps” (that Canny fails to detect), within a maximum connection threshold (thr_{VLL}). Henceforth, we mainly encounter erroneous results when this assumption fails. Some typical examples are shown in Figure 16. Furthermore, there might be several other reasons for such erroneous segmentations. One that often occurs is the textural predominance within the image causing Canny edge detector to eventually miss some significant sections of the true object boundary (e.g. (a), (c) and (e) in Figure 16). Moreover, the illumination effects such as strong shades (e.g. (b)) or insignificant intensity difference between the object and the background (e.g. (d)) are cases where only the human visual perception can exhibit the usage of "knowledge" or "intelligence" to interpolate missing edges.

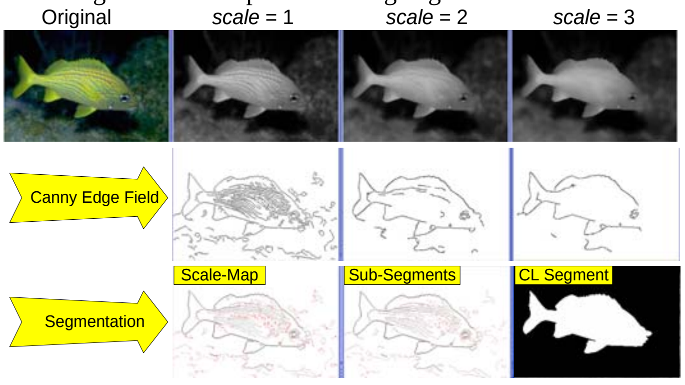


Figure 12: 3-scale simplification process over a natural image and the final CL segment extracted.

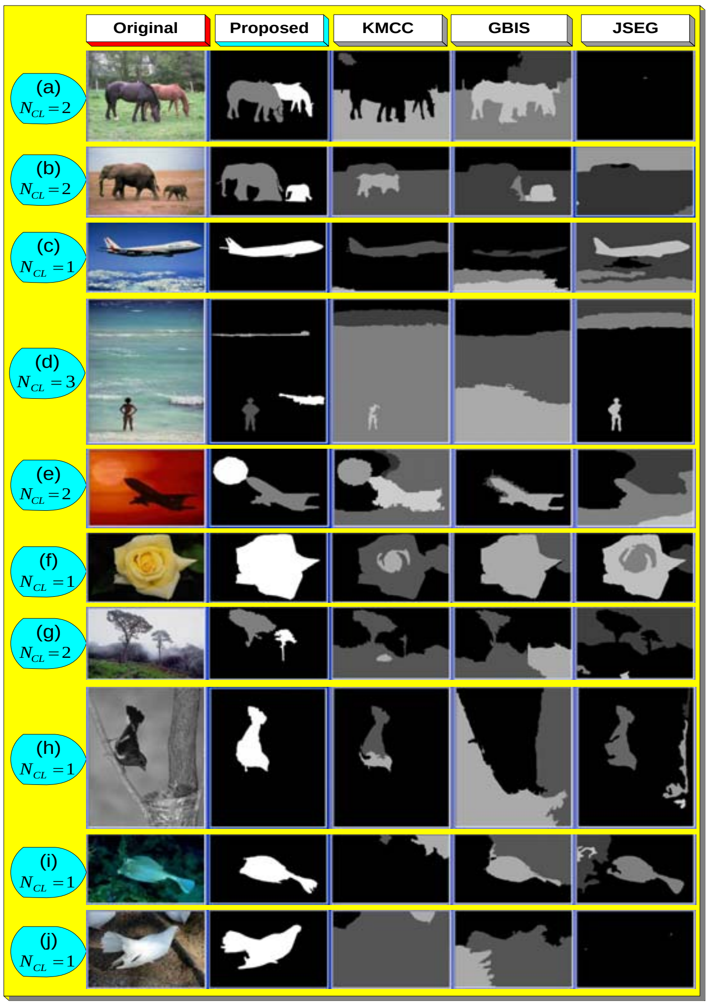


Figure 13: Typical segmentation results. The leftmost column shows the original image, the second and other three columns show the segmentation masks of the proposed method and other three algorithms.

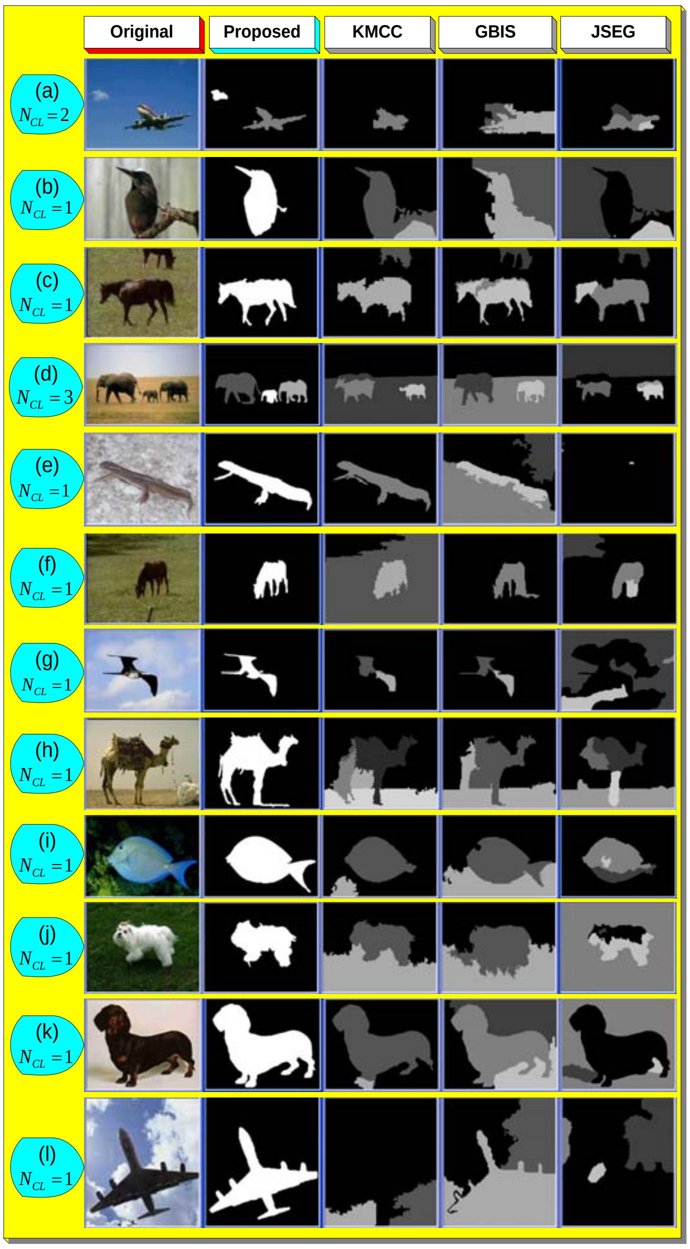


Figure 14: Typical segmentation results. The leftmost column shows the original image, the second and other three columns show the segmentation masks of the proposed method and other three algorithms.

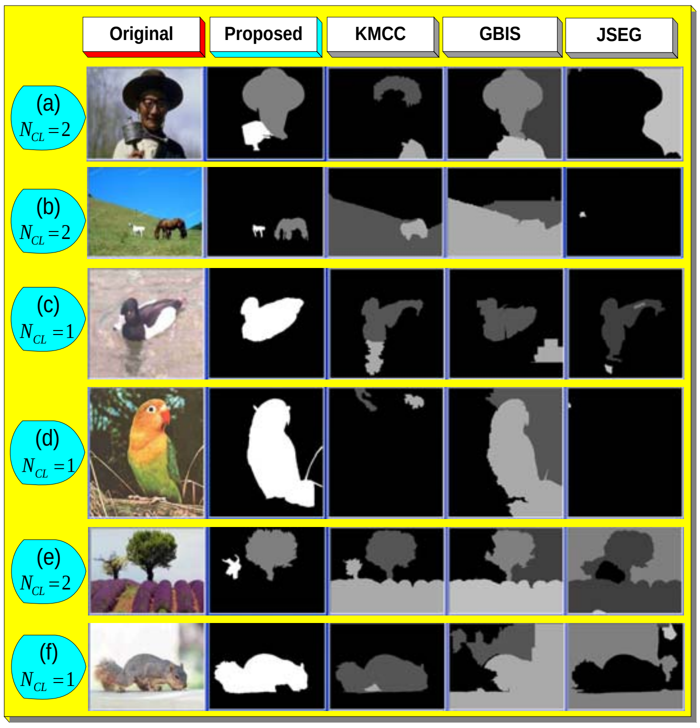


Figure 15: Slightly erroneous segmentation results. The leftmost shows the original image, the second and other three columns show the segmentation masks of the proposed method and other three algorithms.

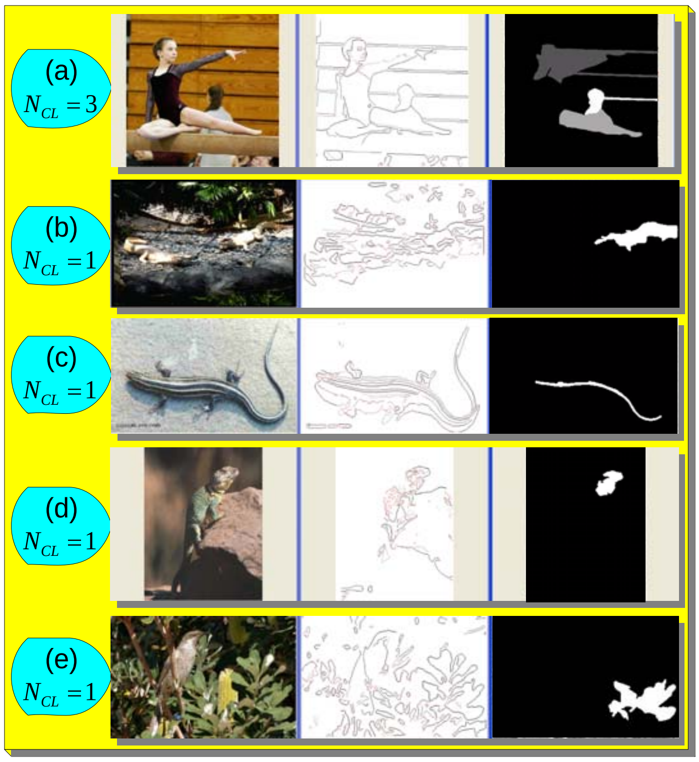


Figure 16: Typical erroneous object extraction results. The leftmost shows the original image, the middle is the *scale-map* with red dots showing the *end-points*. The rightmost shows the segmentation mask.

B. Retrieval Performance

The evaluation of the retrieval performance is carried out subjectively by selecting only the images containing straightforward (obvious) content. In other words, if there is any subjective ambiguity on what the retrieval results should be for a particular image, that sample is simply discarded from the database. Therefore, the experimental results presented in this section depend only on the decisive subjective evaluation via *ground truth* method.

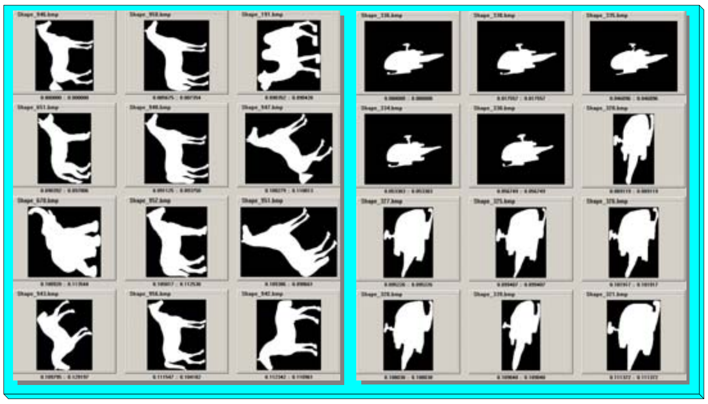


Figure 17: 2 examples showing 12-best retrievals from *binary shape* database. The top-left is the query image.

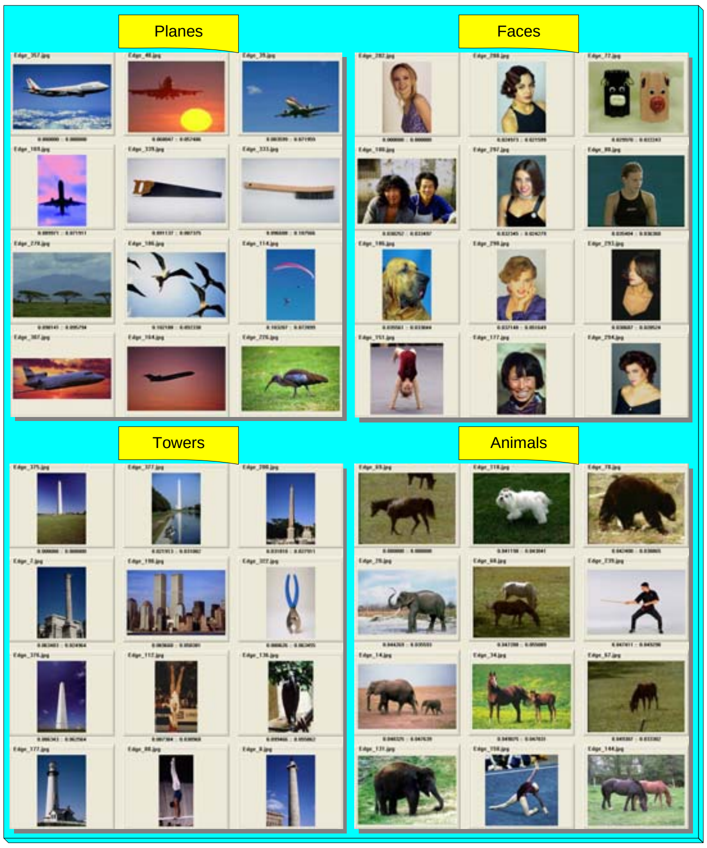


Figure 18: 4 examples showing 12-best retrievals from *natural image* database. The top-left is the query image.

For the analytical notion of performance along with the subjective evaluation, we used the traditional PR (Precision-

Recall) performance metric measured under relevant (and unbiased) conditions, notably using the aforementioned ground truth methodology. Note that recall, *R*, and precision, *P*, are defined as:

$$R = \frac{RR}{TR} \text{ and } P = \frac{RR}{N} \tag{11}$$

where *RR* is the number of relevant items retrieved (i.e. correct matches) among total number of relevant items, *TR*. *N* is the total number of items (relevant + irrelevant) retrieved. For practical considerations, we fixed *N* as 12.

In order to test the retrieval performance we have indexed two image databases using the proposed *FeX* module. The first database is the *binary shape* database, which contains 1400 binary images. This database is basically used to examine the effectiveness of the proposed feature extraction method, since the correct segmentation for this particular kind of images is guaranteed by setting $N_{CL}=1$. The correct retrieval results, as well as the achieved scale and rotation invariance are illustrated in the two typical examples shown in Figure 17. For this database, we observed a significant retrieval performance, i.e. 85% or higher *Recall* for 78% average *Precision* level. The second database is composed of 400 natural images carrying various content (i.e. sports, cars, objects, wild-life scenery, animals, outdoor city-scape, etc.). Among several retrieval experiments, we observed a range of 21-78% *Recall* versus 8-100% of *Precision* level. We have seen that some of the retrieval results can hardly be otherwise achieved using color or texture features, e.g. see Figure 18. During the indexing of the natural image database bearing 400 images, 5 iterations (scales) are used and the average indexing time per image is calculated as 9.82 seconds. No iterations (0 scales) are used for the binary image database and the average indexing time per image is 1.21 seconds.

V. CONCLUSION

The proposed shape based indexing scheme based on automatic and multi-scale object extraction method via *sub-segment* analysis over Canny edge field achieves the following major innovative properties:

- The overall algorithm is unsupervised (no training), with minimal parameter dependency. To our knowledge, it is one of the first attempts to index a natural image database according to object shape description in a fully automatic way.
- It is based on *Gestalt* laws to yield perceptually meaningful segmentation as the final product.
- A multi-scale approach is employed to simplify the image iteratively, removing "details" at each scale and thus creating a priority measure for edges, improving the knowledge about the image constitution (major object boundaries).
- The search space is reduced by such a multi-scale approach. Moreover, a uniform *cost* search algorithm modified with intensive pruning schemes, related to the scale information, increases the search speed significantly.
- The cost function used in the relevance model increases the likelihood of finding true object boundaries by favoring the longest and the most relevant CL segments with the minimal usage of *virtual links.*
- The experimental results approve that the proposed method can be a good alternative to the traditional segmentation algorithms particularly for natural image databases where their primary assumptions usually fail and furthermore a high (one-pixel) resolution (accuracy) is achieved for the extracted object boundaries.
- Finally, the features extracted from the overall segmentation scheme provide an effective indexing and retrieval performance, as assessed via subjective query results.

Current and planned research work include designing of a better and more semantically meaningful feature vector from CL/NCL segments, applying an adaptive merging operation to obtain global object(s) from the extracted CL parts and adapt more Gestalt laws such as *symmetry and surroundedness* in the cost function, to achieve a better object extraction scheme. By using the proposed *FeX* module, integrating the "query by sketch" capability into MUVIS framework is also considered.

REFERENCES

[1] R. Adams and L. Bischof, "Seeded Region Growing", IEEE Trans. on Pattern Analysis and Machine Intelligence , vol. 16, pp. 641-647, 1994.

[2] J. Canny, "A Computational Approach to Edge Detection", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, no. 6, pp. 679-698, November 1986.

[3] Y. –L. Chang and X. Li, "Adaptive image region growing", IEEE Trans. On Image Proc., vol. 3, pp. 868-872, 1994.

[4] Y. Deng, and B. S. Manjunath, "Unsupervised segmentation of color-texture regions in images and video", IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI '01), Vol. 23, No. 8, pp. 800-810, Aug. 2001.

[5] J. P. Fan, D. K. Y. Yau, A. K. Elmagarmid, W. G. Aref, "Automatic image segmentation by integrating color-edge extraction and seeded region growing", IEEE Trans. on Image Processing, Vol. 10, No. 10, pages 1454-1466, Oct. 2001.

[6] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient Graph-Based Image Segmentation", Int. Journal of Computer Vision, Vol. 59, No. 2, Sep. 2004.

[7] Q. Gao, "Extracting Object Silhouettes by Perceptual Edge Tracking", In Proc. of IEEE Int. Conference on Systems, Man and Cybernetics, Vol. 3, pp. 2450-2454, Orlando, USA, Oct. 11-15 1997.

[8] Q. Gao, D. Qing, S. Lu, "Fuzzy Classification of Generic Edge Features", ICPR 2000, vol. 3, pp. 668-671, 2000.

[9] O. Guldogan, E. Guldogan, S. Kiranyaz, K. Caglar, and M. Gabbouj, "Dynamic Integration of Explicit Feature Extraction Algorithms into MUVIS Framework", In Proceedings of the 2003 Finnish Signal Processing Symposium (FINSIG'03), Tampere, Finland, pp. 120-123, 19-20 May 2003.

[10] D. Geiger and A. Yuille, "A Common Framework for Image Segmentation", Int. Journal of Computer Vision, vol. 6, pp. 227-243, 1991.

[11] R. M. Haralick and L. G. Shapiro, "Survey: Image Segmentation Techniques", Comput. Vis. Graph. Image Proc., vol 29, pp. 100-132, 1985.

[12] S. A. Hijjatoleslami and J. Kitter, "Region growing: A new approach", IEEE Trans. On Image Proc., vol. 7, pp. 1079-1084, 1998.

[13] M. Kass, A. Witkin and D. Terzopulos, "Snakes: Active Contour Models", Int. Journal of Computer Vision, vol. 1, pp. 312-331, 1988.

[14] S. Kiranyaz, K. Caglar, O. Guldogan, and E. Karaoglu, “MUVIS: A Multimedia Browsing, Indexing and Retrieval Framework”, in *Proc. of Third International Workshop on Content Based Multimedia Indexing*, CBMI 2003, Rennes, France, 22-24 September 2003.
[15] I. Kompatsiaris and M. G. Strintzis, “Content-based Representation of Colour Image Sequences”, IEEE Int. Conf. on Acoustics, Speech and Signal Proc. 2001 (ICASSP), Salt Lake City, USA, May 2001.
[16] V. Mezaris, I. Kompatsiaris and M. G. Strintzis, "Still Image Segmentation Tools for Object-based Multimedia Applications", Int. Journal of Pattern Recognition and Artificial Intelligence, vol. 18, no. 4, pp. 701-725, June 2004.
[17] S. Mahamud, L. R. Williams, K. K. Thornber, and K. Xu, “Segmentation of multiple salient closed contours from real images”, IEEE Trans. on Pattern Analysis and Machine Intelligence, 25(4):433-444, 2003.
[18] O. J. Morris, M. J. Lee and A. G. Constantinides, “Graph Theory for Image Analysis: An Approach based on the Shortest Spanning Tree”, IEE Proceedings, vol. 133, o. 2, pp. 146-152, 1986.
[19] MUVIS. http://muvis.cs.tut.fi
[20] L. H Staib and J. S. Duncan, “Boundary finding with Parametric Deformable Models”, IEEE Trans. On Pattern Analysis And Machine Intelligence , vol. 14, pp. 161-175, 1992.
[21] P. L. Palmer, H. Dabis and J. Kittler, “A performance measure for boundary detection algorithms”, Comput. Vis. Image Understand., vol. 63, pp. 476-494, 1996.
[22] P. Perona and J. Malik "Scale-Space and Edge Detection Using Anisotropic Diffusion," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.12, No. 7, pp. 629639, 1990.
[23] S. Russell, and P. Norvig, “Artificial Intelligence: A Modern Approach”, Prentice Hall, Englewood Cliffs, NJ, 1995.
[24] P. K. Sahoo, S. Soltani and A. K. C. Wong, “A survey of thresholding techniques”, Computer Vis. Graph. Image Proc., vol. 41, pp. 233-260, 1988.
[25] C. Tomasi and R. Manduchi, “Bilateral filtering for gray and color images”, In Proc. of the Sixth International Conference on Computer Vision, Bombay, India, January 1998.
[26] E. Tuncel and L. Onural, “Utilization of the recursive shortest spanning tree algorithm for video-object segmentation by 2-D affine motion modeling”, IEEE Trans. On Circuits and Systems for Video Tech., vol. 10, no. 5, pp. 776-781, Aug. 2000.
[27] M. Wertheimer, “Laws of organization in perceptual forms”, Partial translation in W.B. Ellis, editor, “A Sourcebook of Gestalt Psychology”, pages 71-88, NY, Harcourt, Brace and Company, 1938.