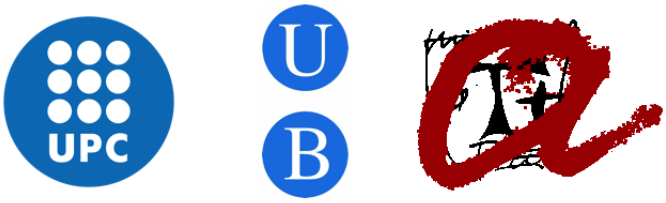

UNIVERSITAT POLITÈCNICA DE CATALUNYA
UNIVERSITAT DE BARCELONA
UNIVERSITAT ROVIRA I VIRGILI

Master in Artificial Intelligence

Master of Science Thesis

Cage active contours: Extension to color spaces and application to image morphing

Jeroni Carandell Saladich

FACULTAT D'INFORMÀTICA DE BARCELONA (FIB)
FACULTAT DE MATEMÀTIQUES (UB)
ESCOLA TÈCNICA SUPERIOR D'ENGINYERIA (URV)

Supervisor:
Laura Igual Muñoz
Department of analysis
and applied Mathematics,
Universitat de Barcelona (UB)

Co-supervisor:
Lluís Garrido Ostermann
Department of analysis
and applied Mathematics,
Universitat de Barcelona (UB)

October 23, 2015

Acknowledgments

I would like to sincerely thank my supervisors Laura Igual and Lluís Garrido for their support. I greatly appreciate their demanding and inquisitive scientific attitude, while keeping always a calm and positive mindset. An extra special acknowledgment must be made to Lluís, who despite his long medical leave, came back with ever more dedication. They are, in my mind, an inspiring example of what university professors should be.

Furthermore I would also like to dedicate this master thesis to my parents Joan and Maria, my brother Blai and my friends in the master: Pablo, Denis, Lorenzo, Hadi, Iosu, Albert, Johan, Anna, Celina, Daniel and many others for the endless studying hours, the unforgettable moments we shared, and the valuable lessons in programming which allowed me to carry out this project. Last but not least, I want to thank Anna for her support and her lessons in Adobe Photoshop which allowed for the creation of the synthetic database.

Abstract

The main purpose of this master thesis is to enhance the performance of Cage Active Contours (CAC) in the context of color image object segmentation as well as provide a theoretical framework on which to justify the potential applications of the segmentation produced.

We present two new energy functions which expand on the previous energies in CAC, by using Mixture Gaussian Models to capture multiple components per region, and extend their applicability to two different color spaces: RGB and the Hue component of HSI/HSV. Furthermore, we provide a mathematical formalization of the components in CAC (*cage* and *contour*) in order to demonstrate and prove the good properties that allow for a segmentation to be used for shape description and for image morphing and warping. In addition, we provide a public implementation in Python for object segmentation of CAC with different energies along with tools for image morphing and warping.

In order to validate our improvements, both quantitative and qualitative tests are used on three different datasets. The results show that the energy in RGB produces a substantial improvement over the previously developed energies for CAC. On the other hand we observe that the energy in the Hue component performs poorly on the given datasets despite its robustness with respect to changes in illumination. In the theoretical part, we prove which initial conditions are needed for a segmentation to provide useful results in shape description. Finally, we provide examples of some possible applications of CAC in morphing supported by our theoretical conclusions.

Contents

List of Figures

1 Introduction

Image segmentation is an extremely complex application of artificial intelligence and, as such, a huge variety of approaches exist. Its complexity derives from the generality of the definition of the object or scene to segment which can have a very context-related or semantically based meaning. A very general definition of image segmentation found in [68] describes it as the process of partitioning an image into disjoint and homogeneous regions. This vagueness motivates us into having a more well defined problem and an initial intuition about the object. In particular, we will focus on segmenting simply connected objects that are visually meaningful. In other words, objects with single regions and no holes and that can be visually differentiable from the background.

Image object segmentation has a variety of applications. In medical image analysis it is key for tasks such as region characterization or tumor monitoring [38, 58, 53]. In multimedia we find applications such as image editing and image and video retrieval from digital databases.

Current methods in object segmentation include hierarchical segmentation, which either agglomerates or divides pixel regions or super-pixels into a desired result such as [6, 45, 26, 3], Artificial Neural Networks (ANN) such as Kohonen's Self-Organizing Maps (SOM) [53], as well as a variety of combination of these and other methods [18, 64, 50].

One of the most used state of the art methods in object segmentation is *Graph Cuts* (GC). This technique uses probabilistic graphical models to assign a probability of each pixel of belonging to one of two regions, the foreground or the background. Once the graph is weighted, GC elegantly and optimally partitions the graph by cutting the weakest edges using a theorem in graph theory called Max-flow min-cut theorem. This theorem states that in a network, the maximum flow that can go from the source to a sink, is equal to the minimum capacity that we need to remove from the network in order to have no path between the source and the sink. The current best Graph Cuts technique is the Geodesic Graph Cut [61] which combines this method with active contours by considering the boundary that separates the regions. Graph Cuts depend on user interaction to determine the seeds of each region but does not ensure single connected objects unless the user further interacts to ensure connectivity.

One of the first methods that emerged in object segmentation, however, were *deformable models*. In [41] the authors point out that "the mathematical foundations of deformable models represent the confluence of geometry, physics, and approximation theory. Geometry serves to represent object shape, physics imposes constraints on how the shape may vary over space and time, and optimal approximation theory provides the formal underpinnings of mechanisms for fitting the models to measured data".

Among these types of methods are active contours, also known as snakes. Cage Active Contours (CAC) proposed in [30] are an type of these methods with region-based energies that parametrize the contour with a *cage*, an ordered set of control points, using mean value coor-

dinates (a distinct generalization of barycentric coordinates). This cage is used to define the evolution of the contour through affine transformations and which in turns allows for restrictive criteria to be introduced in the cage, apart from the already intrinsic properties of these coordinates such as smoothness [27]. The properties of this method allow to easily deal with region-based approaches which proves to be hugely advantageous with respect to most previous parametrized approaches, which are only able to deal with edge-based energies. Furthermore, except for [8], which treats 3D Images, there is almost no work in the field of parametric-based approaches, as far as we know, that is able to deal in a unified manner with the mean, Bayesian and histogram model. Like other parametric-based approaches Cage Active Contours deform the contour locally by moving the cage's points. This approach has proven to be quite versatile in the domain of medical image segmentation, where the structure to be segmented has often only one regular connected component.

It is in this method on which we will base our work. We argue that the uniqueness of this method lies not only in the process of the segmentation but also in the properties of the resulting cage which, as we will show further on, are extremely powerful for automatic morphing and warping, and as shape descriptors. On the other hand, the *status quo* of CAC is quite simple and limited to gray-scale images and therefore provides poor results with arbitrary images.

This thesis contains the following contributions:

1. Enhance Cage Active Contours: This is the main goal of the thesis. We propose to improve the method by creating energy functions that can capture more complex properties of regions as well as extending them to the most used color spaces: RGB and the Hue component of HSI or HSV. In particular, we focus on three extensions of CAC:
   - Creation of a Gaussian Mixture Model-based Energy inspired by the *Gaussian Energy* in [30] (section 4.1.1).
   - Extension of the Gaussian Mixture Energy to RGB (section 4.1.1).
   - Creation of the analogous *mean energy* in [30] for the Hue cyclic values (section 4.2).
2. Experimental Validation: We evaluate the proposed CAC energies using three different datasets from a quantitative and a qualitative point of view (sections 5.4 and 5.3 respectively).
3. Formalize CAC Properties: Mathematically formalize the concept of a family of shapes defined by CAC and prove that a categorization of these can be made if some initial conditions are met (section 6).
4. Highlight the properties of the resulting segmentation of CAC with different applications like automatic image morphing and warping, and shape description (sections 6.3 and 6.2).
5. Public Implementation: We provide our Python implementation (with some wrapped functions in C) of Cage Active Contours with a variety of Energies as well as tools for automatic morphing, warping and shape description. The code can be found in `https://github.com/Jeronics/cac-segmenter/`.

The rest of the document goes as follows: In section 2 we describe the related work in deformable models and set the preliminary concepts to explain our contribution. In section 3, we introduce the different components of Cage Active Contours in detail and in a more

formal language. In section 4 we present the improvements we have carried out in order to enhance CAC and extended their applicability to two different color spaces. In section 5 we evaluate the proposed CAC improvements using three different datasets through qualitative and quantitative approaches. In section 6, we prove, in a formal manner, the advantages of the resulting segmentations of CAC in image morphing, warping and in shape description, as well as provide a couple of examples of the former. Finally, in section 7 we discuss our conclusions and future work.

2 Related Work

In this section, we go over the preliminary concepts that help understand the contributions of this thesis. We will start by looking at the family of methods to which CAC belongs to, followed by a more detailed look at the method in question and explain in which points we will make our contribution. From now on we will only consider the case of 2D images for the sake of simplicity and considering that our work is developed on this domain.

2.1 Deformable Models

Deformable models are curves or surfaces that can move under the influence of physically or pseudo-physically based energies [31]. These can be partitioned into a combination of external energies, which take into account the position of the contour on the image with respect to the features of the latter such as contours, and internal energies, which take into consideration the shape of the curve usually to insure smoothness, continuity, elasticity among other [54, 33, 38]. In this way, by combining these two energies we impose constraints on the resulting segmentation thus ensuring robustness to image noise or poorly defined boundaries. Obviously the more specific the domain is, such as medical image, the more information we can define in these energies. In analogy with Physics, these forces are usually described in terms of associated energies that have to be minimized in order to fit the boundary. This way the mathematical formulation of the problem is simplified and general functional iterative optimization methods like gradient descent can be used. For example, using a basic gradient descent method, the evolution of the boundary is determined by:

$$r_{t+1} = r_t - \alpha \nabla E(r_t) \tag{2.1}$$

where α is the evolution rate parameter, r is a vector describing the contour and $E(r_t)$ is the energy functional to be minimized. Furthermore, in the case of object segmentation, the internal energy can be decomposed into the energies in the interior and exterior parts of an interface:

$$E(r_t) = E_{int}(r_t) + E_{ext}(r_t) + E_c(r_t) \tag{2.2}$$

where E_{ext}, E_{ext} and E_c correspond to the interior, exterior and constraint energies. In the rest of this thesis we will use internal and interior internal energies interchangeably to mean the latter, and the term *region energy* to mean the former.

2.2 Active contours

Active contours invented by [41] are a general method for delineating an object outline that can be fit to tackle the problem of single connected object and have indeed proven to be a very powerful tool in doing so. Also known as snakes, they are deformable models that consist on evolving an interface which is propagated in order to recover the shape of the object of interest.

The description of the interface sub-categorizes these method into *parametric* and *geometric* approaches. The first approach requires, as the name implies a set of discrete parameters, such as points as seen in [41] or basis functions (a basis for a function space) such as B-splines [37, 60]. The advantage of basis functions is their linear combinations have inherent regularity.

Conversely, geometric active contours defined as the zero level set of a higher dimensional function have more topological flexibility, because contours can break apart or join without the need of re-parametrization. However, this property can prove to be a double edge-sword when the desired shape has to have a specific topology. Level sets are the most representative technique in this category.

The evolution of these interfaces is driven by the minimization of an energy function defined so as to express the properties of the object to be segmented in mathematical terms. In this context, we have to differentiate two types of image features in which these properties are expressed: edge-based, such as the image gradient on the contour as in [14], or region-based terms, as introduced by Chan and Vese in [16]. The advantage of region-based terms are known to be more robust to noise than edge-based contours and therefore do not require the initial boundary to be so close to the solution [68]. The work of Chan and Vese is based on evolving the interface according to the variance of the gray-level values of both interior and exterior regions allowing for segmentation of objects with boundaries not defined by gradient to be detected. This approach has been extended, since then, to other features such as the Bayesian model [63] and histogram model [51]. These approaches define the whole inner region of the evolving contour as the interior region and its complement as the exterior. Thus, they may fail if these features are not spatially invariant. In [52] a solution is proposed by considering the features in a band around the evolving contour. Another solution proposed by [43] is to consider the inner and outer regions as those points that are in the intersection of their respective regions and in the ball centered on the contour. In [44] a more context-aware solution is introduced where a kernel function is applied to each point to define a region-scalable fitting term. Finally, two fast algorithms are presented in [10]) and [66], where a B-Spline parametrization and a discrete approximation-based representation are presented, respectively.

2.3 Level Sets

The level set method, introduced in [56] consists on embedding an image in a higher dimensional space through a function ϕ so that the curve C is defined as the intersection at 0 of its image. Therefore we have that $C = \{(x, y) : \phi(x, y) = 0\}$. If the curve is defined as $C(t) : [0, 1] \to \mathbb{R}^2$, we get a Euclidean parametrization where if t is the evolution parameter then C(t) is the curve of level zero of $\phi(t, x, y)$. Through rigorous mathematical reasoning it can be demonstrated that evolving a curve in the normal direction is equivalent to solving for the differential equation:

$$\phi_t = \frac{\partial \phi}{\partial t} = \gamma |\nabla \phi| \tag{2.3}$$

where t is the evolution parameter and the term ϕ_t is known as the *gradient flow*. This makes way for defining different energy functions with respect to the curve. In [16], the authors present a method to evolve a curve by minimizing the variance in the interior, Ω_1, and exterior region, Ω_2, defined by the curve. Formalized as:

$$E(C) = \frac{1}{2}\iint_{\Omega_1}(I-\mu_1)^2\,dx\,dy + \frac{1}{2}\iint_{\Omega_2}(I-\mu_2)^2\,dx\,dy, \tag{2.4}$$

where the terms based on the contour length and area enclosed by Ω_1 have been dropped for simplicity. In (2.4), the image I corresponds to the observed data. The μ_1 and μ_2 refer to mean intensity values in the interior and exterior region, respectively.

In [63], an approach that assumes a Gaussian model for Ω_1 and Ω_2 was presented. For that issue the pixel intensities inside and outside the contour $\mathcal{C}$ are assumed to follow a Gaussian probability distribution. The energy to be minimized (considering only the region based terms) is given by

$$E(\mathcal{C}) = \iint_{\Omega_1} e_1\,dx\,dy + \iint_{\Omega_2} e_2\,dx\,dy. \tag{2.5}$$

Here e_1 and e_2 correspond to the log-likelyhood function

$$e_h = \log\sigma_h + \frac{(I(p)-\mu_h)^2}{2\sigma_h^2} \tag{2.6}$$

where $h = \{1, 2\}$, μ_h is the internal ($h = 1$) or external ($h = 2$) mean and σ_h is the internal or external variance.

In [51], a level set method to segment images using histograms is presented. In this case, the objective is to segment an image by maximizing the distance between the two probability distributions. The authors propose to use the *Bhattacharyya* distance to maximize the difference between probability distributions p_i. Particularly, the distance is $-\log B$ where B is the *Bhattacharyya* coefficient defined as

$$B = E(\mathcal{C}) = \int_R \sqrt{p_1(z)p_2(z)}dz, \tag{2.7}$$

and $z \in R$. The functions $p_1(z)$ and $p_2(z)$ correspond to the gray-level histogram of regions Ω_1 and Ω_2 and are computed by means of Parzen windowing. By minimizing the energy function B, we maximize the distance between the two probability distributions.

2.4 Discussion

In this thesis, we are going to work on Cage Active Contours (CAC), a type of parametric active contour which are fit to work with region energies similar to the ones defined in Level set methods[30].

Because of the theoretical framework upon which level sets are built, very arduous steps are required in order to evolve the curve, including the application of Euler-Lagrange (EL) to solve for a stationary point [13]. As it will be seen in section 3, Cage active contours allow for discretization of the energy and the calculation of the gradient through partial derivatives as opposed to using EL. In Figure 2.1, a diagram of the process of obtaining the gradient contrasts the steps taken by both CAC and level sets.

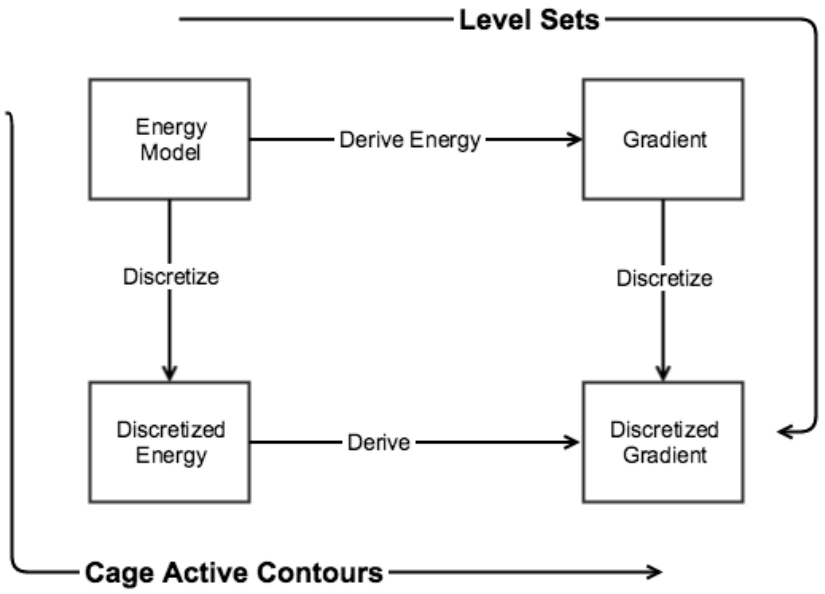

Figure 2.1: Diagram of paths for calculating an energy's gradient for CAC and Level Sets.

3 Cage Active Contours

Before defining Cage Active Contours, we have to introduce the coordinates which will be used to parametrize these active contours. Furthermore, we will derive some of the properties of these coordinates and formally introduce some concepts which will serve as an introduction to the shape description in section 6.

3.1 Mean Value Coordinates

Mean value coordinates are used in computer graphics to linearly interpolate values given at a set of vertices of a closed contour. This allows for applications such as parametrization [42, 22, 28], shading [32, 59] or deformation to be carried out with extremely good results. Although these coordinates were initially introduced for mesh parametrization [27], Cage Active Contours uses them form deformation.

These applications require that a point p be expressed as an *affine combination* of the vertices $v_1, v_2, \ldots, v_N$ of an enclosing closed contour. That is,

$$p = \sum_{i=1}^{N} \varphi_i(p) v_i \tag{3.1}$$

where $\varphi_i(p)$ is the corresponding *affine coordinate* of the point p with respect to the vertex v_i and N is the number of vertices.

A variety of approaches have been presented for the computation of $\varphi_i(p)$. In deformation applications we have harmonic coordinates [39], Green coordinates [46], or mean value coordinates [27]. The advantage of the latter over the rest include a simple computation and the convenience of being able to parametrize any point of the space, be it inside or outside the polygon demonstrated in [34].

Given a set of ordered[1] vertices $V = (v_1, v_2, \ldots, v_N)$ of a polygon of N points disposed in an anticlockwise order, the *mean value coordinates* of a point p with respect to[2] V are $\varphi^V(p) = (\varphi_i^V(p) | i \in (1, \ldots, N))$.

$$\varphi_i(p) = \begin{cases} \delta_{i,j} & \text{if } p = v_j \\ (1-\mu)\delta_{i,j} + \mu\delta_{i,j+1} & \text{if } p = v_j(1-\mu) + v_{j+1}\mu \\ \frac{w_i}{\sum_{j=1}^{N} w_j} & \text{otherwise} \end{cases} \tag{3.2}$$

[1] An ordered set will be defined with (,) while unordered set will be denoted by {,}

[2] In order to simplify the notation, we will use $\varphi(p)$ instead of $\varphi^V(p)$ unless there is a possible ambiguity in the context.

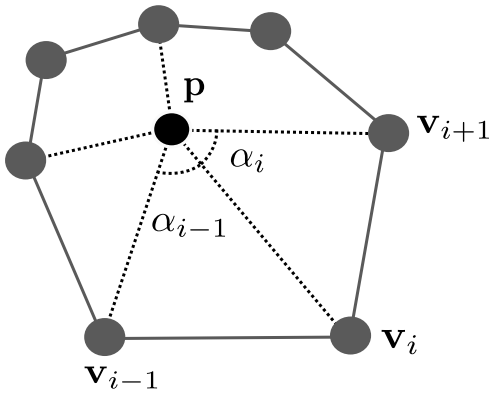

Figure 3.1: Mean Value coordinate of vertex points (image from [30])

where

$$\delta_{i,j} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases} \tag{3.3}$$

and w_i is calculated

$$w_i = \frac{\tan(\frac{\alpha_{i-1}}{2}) + \tan(\frac{\alpha_i}{2})}{||v_i - p||} \tag{3.4}$$

where $\|v_i - p\|$ is the distance between the vertex v_i and the considered point p and α_i is the *signed* angle of $[v_i, p, v_{i+1}]$.

Now we have that given the affine coordinates $\varphi(p)$ of a point p, the point p can be recovered with (3.1). If the vertex v_i of the cage moves to position v'_j, the 'deformed' point p' can be recovered as

$$p' = \sum_{i=1}^{N} \varphi_i(p) v'_i, \tag{3.5}$$

where note that the point p' is recovered from the affine coordinates $\varphi_i(p)$, see figure 3.2.

Given a set of points, the affine coordinates for each point are computed in an independent way using (3.2). If a point v_i of the polygon is stretched in a particular direction, all the points follow the same direction with an associated weight given by $\varphi_i(p)$ which is inversely proportional to the distance from p to v_i, since it is the denominator of (3.4). In figure 3.2, this effect is depicted when point v_i in the left image is translated to v'_i. the internal point p suffers a greater deformation than the points which are farther where the weight are smaller and hence, they are barely affected by this deformation.

Figure 3.3 illustrates the influence of the cage distance to the curve. The polygon is shown with a solid line whereas the evolution curve with a dashed line. The first column shows the initial configuration. The second and third column shows how the curve is deformed when a control point of the polygon is moved. As can be seen, the movement of the polygon produces a smooth deformation of the curve. The closer the polygon to the curve the higher the deformation that is applied to the curve.

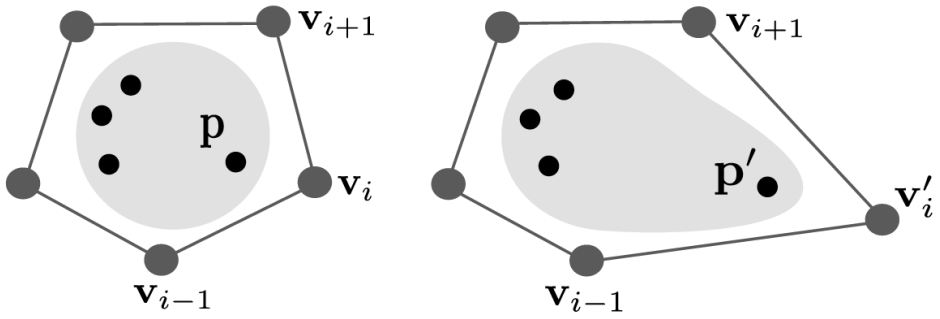

Figure 3.2: Influence of a vertex over the points on the plain (image from [30])

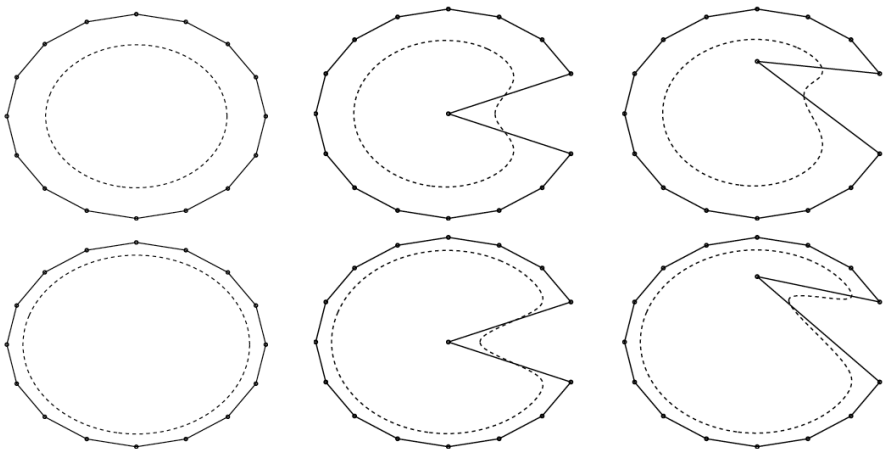

Figure 3.3: The curve deformation for two polygon configurations. The polygon is shown as solid line and the evolving curve as dashed line.

3.1.1 Properties of Mean value coordinates

Other properties that will be necessary for the development of the applications in section 6 are that mean value coordinates have the following properties [34]:

C.1 *Affine precision*: , for any affine function $f : \mathbb{R}^2 \to \mathbb{R}^D$, $f = \sum_{i=1}^{N} f(v_i)\varphi_i^V$ for $v_i \in V$

C.2 *Similarity invariance*: If $f : \mathbb{R}^2 \to \mathbb{R}^2$ is a *similarity* and for a cage $V' = f(V)$ we have that $\varphi^V(p) = \varphi^{V'}(f(p))$

C.3 *Smoothness*: φ_i is C^∞ everywhere except at the vertices v_j where it is only C^0.

C.4 *Edge Linearity*: φ_i^V is linear along the edges of the cage V.

C.5 *Refinability*: if we redefine V to V' by splitting an edge between vertices v_j and v_{j+1} at $v = (1-\mu) * v_j + \mu * v_{j+1}$, then $\varphi_j^V = \varphi_j^{V'}\mu + (1-\mu) * \varphi_j^{V'}$.

3.2 Components of Cage Active Contours

Cage Active Contours are deformable models that have three major components: an *initial contour*, an *initial cage*, and an *energy* function.

Let us define formally what we mean by each:

Definition 1. *A curve on a plane is a continuous mapping* $\gamma : [a, b] \to \mathbb{R}^2$ *such that* $[a, b] \in \mathbb{R}$.

Definition 2. *A Jordan curve is a non-intersecting, continuous closed curve.*

Definition 3. *A contour will be used to define the image of a closed curve* $\gamma : [a, b] \to \mathbb{R}^2$. *such that* $\gamma(a) = \gamma(b)$.

From now on, however, we will use the word curve to mean contour for literary purposes unless it is explicitly distinguished.

The CAC's *initial contour* is a Jordan curve so that by the Jordan Curve Theorem we can assure that it divides the plane into two regions Ω_1, Ω_2 which correspond to the interior and the exterior of the Curve respectively.

We define cage as

Definition 4. *A cage is an ordered group of points* $V = (v_1, v_2, \ldots, v_N)$ *on the plane* $\mathbb{R}^2$.

The *initial cage* V of N points must define[3] a simple N-sided polygon since it is a requisite to be able to parametrize points on the plane using *mean value coordinates*. These barycentric coordinates have very good properties which will also open the possibility to different applications such as silhouette descriptors, morphing or image interpolation in seciton 6.

The *energy* E is a function with respect to a contour, however, since the contour C is parametrized by a cage V, and the contours that will be able to define will depend exclusively on V, we can define the energy function as

$$\begin{aligned} E\colon (\mathbb{R}^2)^{\mathrm{N}} &\to \mathbb{R} \\ V &\longmapsto E(V) \end{aligned} \tag{3.6}$$

The function must be defined in a way so that it is minimum when the object to segment is in the interior region and the background in the exterior. Of course this idea stems from the assumption that the object differs from that of the background in appearance. The goal is then to minimize the energy with respect to a cage:

$$\min_{v_1, v_2, \ldots, v_N} E(v_1, v_2, \ldots, v_N) \tag{3.7}$$

Since the energy function is in terms of the cage, we can minimize the function by applying gradient descent on the energy with respect to the control points.

From the very simple models on gray-scale image defined in [30], we can develop more sophisticated energies as more complex properties are taken into consideration.

[3] A cage defines a polygon by joining its vertices in order, the last with the first and removing the middle point of any consecutive collinear triplet. It is important to note that a cage is *not* a polygon, since a cage can have three consecutive collinear points while a polygon cannot by definition.

3.3 Energy Functions

As we have seen in section 3.5, CAC allows for more complex region-based energies to be used than other methods such as level sets. In [30], three different energy functions derived from analogous forms in level sets are defined: the Mean Energy, the Gaussian Energy and the Histogram Energy.

3.3.1 Mean Energy

The first energy assumes that the gray-level values of pixels inside Ω_1 and Ω_2 can be modeled with a single value which corresponds to the mean. This method follows the formulation presented in (2.4).

The measure of the region energy can be defined as follows

$$E_{\text{Mean}} = \sum_{h=1}^{2} \sum_{p \in \Omega_h} \frac{1}{2}(I(p) - \mu_h)^2 \tag{3.8}$$

where the mean gray-level value of Ω_h is defined as

$$\mu_h = \frac{1}{|\Omega_h|} \sum_{p \in \Omega_h} I(p), \tag{3.9}$$

Since the objective is to minimize this energy, we need to compute its derivatives with respect to the cage vertices V. The gradient of E_{Mean} with respect to v_j is

$$\nabla_{v_j} E_{\text{Mean}} = \sum_{h=1}^{2} \sum_{p \in \Omega_h} (I(p) - \mu_h)(\varphi_j(p)\nabla I(p) - \nabla_{v_j}\mu_h),$$

where

$$\nabla_{v_j}\mu_h = \frac{1}{|\Omega_h|} \sum_{p \in \Omega_h} \nabla I(p)\varphi_j(p). \tag{3.10}$$

The previous formulation assumes a dependency of μ_h with respect to the region Ω_h. In classical level set approaches, such as [16], authors do not take into account this dependency and use a coordinate descent approach to minimize the functional: the mean μ_h and the evolution equation are updated iteratively to evolve the equation. In the case of CAC, the authors assume $\nabla_{v_j}\mu_h = 0$ since no difference is appreciated in their experiments.

3.3.2 Gaussian Energy

The Gaussian energy presented in [30] decreases when the values of each region Ω_h have stable statistics μ_h and σ_h. In other words, the curve will stop evolving when each region has points who's values have a higher probability of being in that region than otherwise.

$$E_{\text{Gauss}} = \sum_{h=1}^{2} \sum_{p \in \Omega_h} -log(P_h(I(p)))) \tag{3.11}$$

where P_h is the probability an intensity of p, $I(p)$ belongs to the normal distribution defined by region h's seed.

$$log(P_h(I(p)) = -\log(\sqrt{2\pi}\sigma_h) - \frac{(I(p)-\mu_h)^2}{2\sigma_h^2} \tag{3.12}$$

In order to minimize the energy, the authors calculate the gradient the derivative of the energy with respect to each control point v_i to apply gradient descent:

$$\nabla_{v_j} E_{\text{Gauss}} = \sum_{h=1}^{2} \sum_{p\in\Omega_h} \frac{I(p)-\mu_h}{\sigma_h^2} \nabla I(p)\varphi_i(p) \tag{3.13}$$

Notice that we do not apply the rule chain to μ_h or σ_h like the original Gaussian CAC energy because they are previously chosen constants.

3.3.3 Histogram Energy

Also presented in [30] is the histogram energy an adaptation to CAC of a level set energy proposed in [51] (see section 2.3). This energy is minimal when the probability density function of gray-values from each region p_1 and p_2 are most different. In order to measure this dissimilarity, the authors use the *Bhattacharyya distance* between two density functions [15]. This distance is defined as:

$$d_B = -log(B_{\text{coeff}}(p_1, p_2)) \tag{3.14}$$

where the B_{coeff} is a similarity measure known as the *Bhattacharyya coefficient*:

$$B_{coeff}(p_1, p_2) = \int \sqrt{p_1(k)p_2(k)}dx \tag{3.15}$$

In order to simplify, the authors of CAC approximate the distribution with a probability histogram of K partitions, where the value of each bin is denoted $p_1(k)$ and $p_2(k)$. This way, the discrete version of (3.15) can be expressed as a finite sum and so the histogram energy can be expressed as:

$$E_{\text{histogram}} = \sum_{k=1}^{K} \sqrt{p_1(k)p_2(k)} \tag{3.16}$$

The histogram bin $p_h(k)$, $h \in \{1, 2\}$, is computed by means of the Parzen window [51]

$$p_h(k) = \frac{1}{|\Omega_h|} \sum_{p\in\Omega_h} g_{\sigma_h}(k - I(p)), \tag{3.17}$$

where g_{σ_h} is the Gaussian function of zero mean and variance σ_h.

The gradient with respect to a vertex v_j is:

$$\nabla_{v_j} E_{\text{histogram}} = \sum_{k=0}^{K-1} \left(\frac{1}{2}\sqrt{\frac{p_2(k)}{p_1(k)}} \nabla_{v_j} p_1(k) + \frac{1}{2}\sqrt{\frac{p_1(k)}{p_2(k)}} \nabla_{v_j} p_2(k) \right) \tag{3.18}$$

and where the derivative of (3.17) each bin's function is

$$\nabla_{v_j} p_h(k) = -\frac{1}{|\Omega_h|} \sum_{p\in\Omega_h} g'_{\sigma_h}(k - I(p)) \nabla I(p)\varphi_j(p), \tag{3.19}$$

where $g'_{\sigma_h}(.)$ is the derivative of the Gaussian function. The value of σ_h is automatically computed as proposed in [51].

3.4 Internal Energies

Internal energies are energies that focus on restraining the behavior of the shape of the Curve. Properties like smoothness, length, curvature, flexibility, expandability and contractability, topological preservation as well as shape priors can all be introduced in these energies.

In the context of active contours, the problem of handling the crossings of the active contour $\mathcal{C}$ has already been tackled in the literature [24].

This constraint has been implemented by defining two cost functions. The first cost function allows to detect if a vertex v_i gets at a distance less than d to a vertex v_j, with $j \neq i$, as follows

3.4.1 Cage constraint Energy

The Cage constraints energy grows when a vertex of a cage gets too close to an edge, or another vertex. This is used to avoid the polygon defined by the cage from self-intersecting.

Thus, the internal energies are defined for vertex to vertex and vertex to edge repulsion.

$$\psi_{\text{Constraint}}(v) = \sum_i \left(\psi_{\text{Vertex}}(v_i) + \psi_{\text{Edge}}(v_i)\right). \tag{3.20}$$

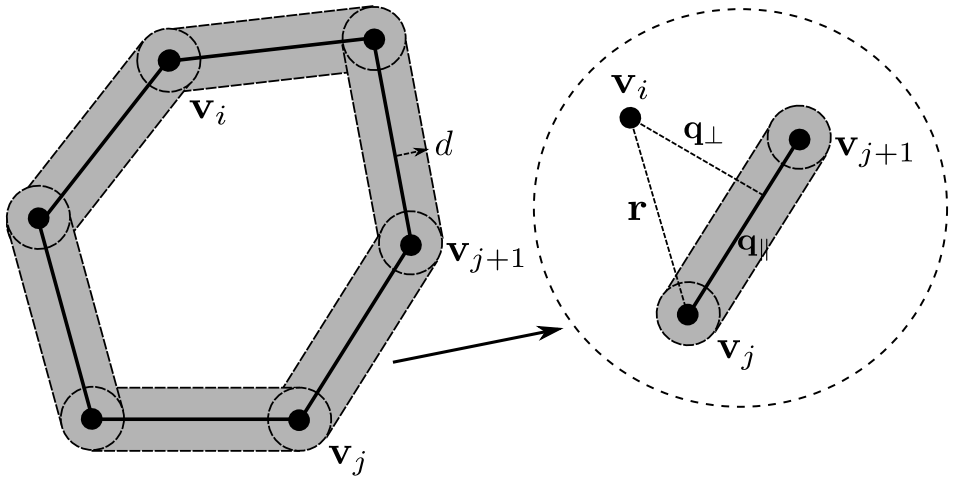

Figure 3.4: Band around the cage, where the vertices are repulsed. Image from[30]

Vertex Constraint Energy

The vertex constraint energy insures that vertices do not come together. This will not only avoid merging points but will also avoid dividing by zero in the gradient of the edge constraint Energy in (3.24) (see section 3.4.1).

$$E_{\text{Vertex}} = \sum_{i=1}^{N-1} \sum_{j=i+1} \psi_{vertex}(v_i, v_j) \tag{3.21}$$

where

$$\psi_{vertex}(v_i, v_j) = (d - \|v_i - v_j\|)^2 \tag{3.22}$$

$$\psi_{\text{vertex}}(v_i, v_j) = \begin{cases} 0 & d \leq \|v_i - v_j\| \\ (d - \|v_i - v_j\|)^2 & d > \|v_i - v_j\| \end{cases} \tag{3.23}$$

Since the gradient with respect to each vertex will be applied to the whole energy, we need to we have to find the partial derivatives. If $d < \|v_i - v_j\|$, we have:

$$\nabla_{v_i}\psi_{vertex}(v_i, v_j) = \begin{cases} 0 & d \leq \|v_i - v_j\| \\ \frac{\partial(d-\|v_i-v_j\|)^2}{\partial v_i} = \frac{2(\|v_i-v_j\|-d)(v_i-v_j)}{\|v_i-v_j\|} & d > \|v_i - v_j\| \end{cases} \tag{3.24}$$

$$\nabla_{v_j}\psi_{vertex}(v_i, v_j) = \begin{cases} 0 & d \leq \|v_i - v_j\| \\ \frac{\partial(d-\|v_i-v_j\|)^2}{\partial v_i} = -\frac{2(\|v_i-v_j\|-d)(v_i-v_j)}{\|v_i-v_j\|} & d > \|v_i - v_j\| \end{cases} \tag{3.25}$$

Finally we get that

$$\nabla_{v_i} E_{\text{Vertex}} = \sum_{j=1}^{i-1} \psi_{vertex}(v_j, v_i) + \sum_{j=i+1}^{N} \nabla_{v_i}\psi_{vertex}(v_i, v_j) \tag{3.26}$$

Edge Constraint Energy

For each edge of the cage defined by v_j and v_{j+1}, all other vertexes $v_i \notin \{v_j, v_{j+1}\}$ we define the edge energy as

$$E_{\text{Edge}} = \sum_{i=1}^{N} \sum_{j \neq i, i-1} \psi_{edge}(v_i, v_j) \tag{3.27}$$

where

$$\psi_{\text{Edge}}(v_i, v_j) = \begin{cases} 0 & R(v_j, v_{j+1}) < 0 \\ 0 & 1 < R(v_j, v_{j+1}) \\ 0 & d < d_\perp \\ (d - d_\perp)^2 & Otherwise \end{cases} \tag{3.28}$$

where $d_\perp = \frac{\|q_\perp \cdot r\|}{\|q_\perp\|}$, $q = (v_{j+1} - v_j)$, $r = (v_i - v_j)$ and $R(v_j, v_{j+1})$ is the projection of r on q normalized with respect to q so that if it is in between zero or one, the vertex v_i is in the band defined by the perpendicularly prolonged segment $\overline{v_j, v_{j+1}}$.

For each vertex, we have to calculate the derivative of E_{edge}. This would imply that for each summand that depends the vertex (with which we derive), would have to be derived. There are three possible positions of a vector in $\psi_{edge}(v_i, v_j, v_j + 1)$ depending on whether they are v_i, v_j or v_{j+1}:

When $R(v_j, v_{j+1}) \geq 0$, $1 \geq R(v_j, v_{j+1})$ and $d \geq d_\perp$

For $k \in \{i, j, j+1\}$

$$\nabla_{v_k}\psi_{\text{Edge}}(v_i, v_j) = 2(d - d_\perp)\nabla_{v_k} d_\perp \tag{3.29}$$

$$\nabla_{v_i} d_\perp = \frac{1}{\|q_\perp\|} \frac{\partial\sqrt{(q_\perp \cdot (v_i - v_j)^2}}{\partial v_i} = \frac{(q_\perp \cdot (v_i - v_j))q_\perp}{\|q_\perp\|\sqrt{(q_\perp \cdot (v_i - v_j))^2}} = \frac{(q_\perp \cdot r)q_\perp}{\|q_\perp\|\|q_\perp \cdot r\|} \tag{3.30}$$

$$\nabla_{v_j} d_\perp = \frac{\|q_\perp\| \frac{\partial \|q_\perp \cdot r\|}{\partial v_j} - \|q_\perp \cdot r\| \frac{\partial \|q_\perp\|}{\partial v_j}}{\|q_\perp\|^2} = \frac{-\|q_\perp\| \frac{(q_\perp \cdot r)(r+q_\perp)}{\|q_\perp \cdot r\|} + \|q_\perp \cdot r\| \frac{q_\perp}{\|q_\perp\|}}{\|q_\perp\|^2} \tag{3.31}$$

$$\nabla_{v_{j+1}} d_\perp = \frac{\|q_\perp\| \frac{\partial \|q_\perp \cdot r\|}{\partial v_{j+1}} - \|q_\perp \cdot r\| \frac{\partial \|q_\perp\|}{\partial v_{j+1}}}{\|q_\perp\|^2} = \frac{\|q_\perp\| \frac{(q_\perp \cdot r) r}{\|q_\perp \cdot r\|} - \|q_\perp \cdot r\| \frac{q_\perp}{\|q_\perp\|}}{\|q_\perp\|^2} \tag{3.32}$$

Therefore the derivative of the Edge energy turns out to be:

$$\nabla_{v_k} E_{\text{Edge}} = \sum_{j \neq k,k-1} \nabla_{v_k} \psi_{edge}(v_k, v_j) + \sum_{i \neq k,k+1} \nabla_{v_k} \psi_{edge}(v_i, v_k) + \sum_{i \neq k,k-1} \nabla_{v_k} \psi_{edge}(v_i, v_{k-1}) \tag{3.33}$$

This means that in a cage of N vertices, each vertex has a total of 3(N-2) derivatives of ψ_{Edge} as summands. This can be too computationally expensive. A possible alternative would be to treat the psi function as a function depending solely on v_i, $\psi_{\text{Edge}}(v_i)$ thus reducing tremendously the number of computations.

3.5 Segmentation process of CAC

Assuming that E, V and C are given, we can begin the process which takes place in the CAC segmentation algorithm. pseudo-code is provided in algorithm 1. In figure 3.5 a diagram of this process is shown. The inputs of this system are the image on one hand and the components of Cage Active Contours on the other, the initial contour, the initial cage and the energy function. A more thorough description of the segmentation process can be found in [30].

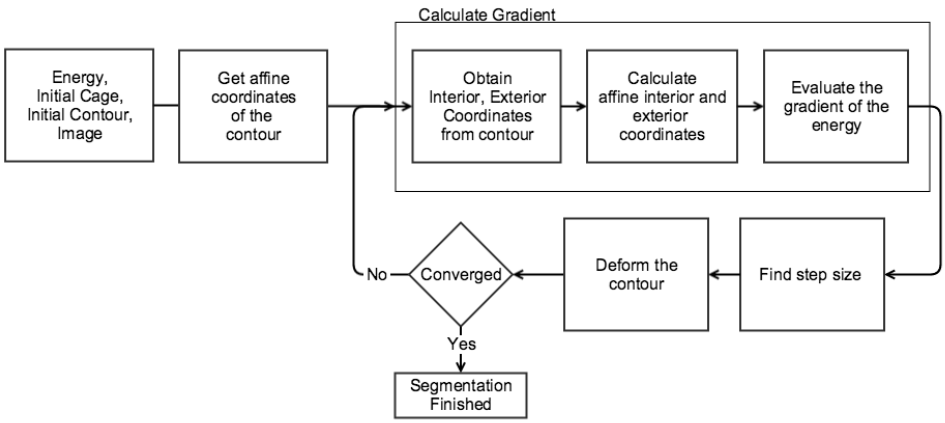

Figure 3.5: CAC diagram.

3.5.1 Gradient descent

In order to segment the image, the Energy function with respect to the contour must be minimized. Since there are a few restrictions on the cage, the iterative gradient descent algorithm is

integrated into the *Segment Object* procedure in lines 1 to 19 of the pseudo-code. The *Calculate Gradient* method iteratively updates the cage $V = \{v_1, \ldots, v_N\}$ represented as V^k in iteration k to V^{k+1} for iteration $k+1$ through the update:

$$V^{k+1} = V^k + \alpha^k S^k, \tag{3.34}$$

where S^k is the search direction and α^k the step. Usually the steepest descent direction $S^k = -\nabla_{V^k} E$ is taken as the search direction.

$$S^k = -\left(\frac{\nabla_{V_1^k} E}{||\nabla_{v^k} E||_{\max}}, \frac{\nabla_{v_2^k} E}{||\nabla_{v^k} E||_{\max}}, \ldots, \frac{\nabla_{v_N^k} E}{||\nabla_{v^k} E||_{\max}} \right) \tag{3.35}$$

where

$$||\nabla_{v^k} E||_{\max} = \max \left\{ ||\nabla_{v_j^k} E|| \ : \ j = 1 \ldots N \right\}. \tag{3.36}$$

The authors assure that the computation of α is a critical issue for the good performance of the algorithm. After this update, the *Deform Points* function is applied to the Contour so that it is updated with respect to the new cage using equation (3.5). It is of extreme importance the fact that the mean value coordinates of the contour are only computed once in the beginning. This is seen in line 4 of the pseudo-code before the iterative loop.

In CAC the authors use gradient descent by means of a two stage process.

First stage

The goal in this stage is to obtain a segmentation as close as possible to the solution by restricting the step and direction in which the cage vertices may evolve. The authors are inspired in balloon forces similar to the ones described in [20]. At each iteration of the gradient descent each cage vertex v_j^k is restricted to move along the line that passes through the center point of v_c and v_j^k. That is, the gradient vector $\nabla_{v_j^k} E$ is projected on the line joining v_c and v_j before normalizing it. This can be seen in figure 3.6 with red lines representing the gradient vector while the blue lines their projection on the line that passes through its corresponding vertex v_j and the center point v_c.

In this first stage, the step α is constant for each iteration and corresponds to a motion of $\alpha = \beta$ pixels of the cage vertices[4], which is the maximum distance points are allowed to approach edges or other vertices to ensure the cage does not cross over.

The authors suppose that if the cage vertices move iteratively from side to side around the reached minimum,they have approximated a certain minimum. Since each cage vertex move in a one-dimensional direction, this phenomena can be mathematically expressed as follows:

$$(\nabla_{\mathbf{v}_j^k} E)^T \nabla_{\mathbf{v}_j^{k-2}} E > 0, \qquad (\nabla_{\mathbf{v}_j^{k-1}} E)^T \nabla_{\mathbf{v}_j^{k-3}} E > 0, \qquad (\nabla_{\mathbf{v}_j^k} E)^T \nabla_{\mathbf{v}_j^{k-1}} E < 0, \qquad \forall k \in V \tag{3.37}$$

This condition has to be satisfied independently, for each cage vertex, in order to stop the first stage and begin the second stage.

[4] β is explained in section 3.4.1.

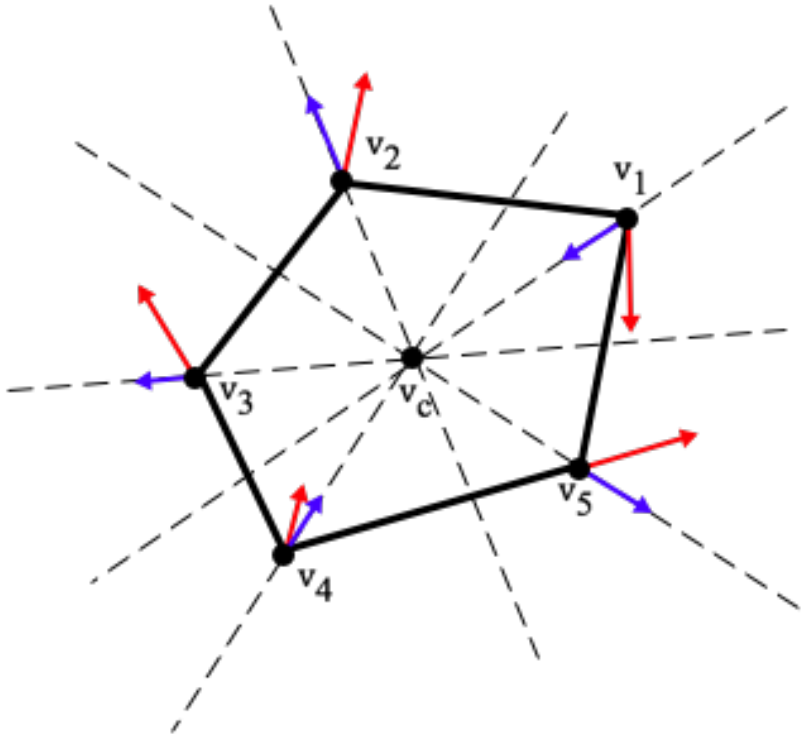

Figure 3.6: Depiciton of the first stage restriction.

Second stage

When the cage meets the conditions in (3.37), an approximation of the segmentation is met and the second stage begins. The restrictions on the direction of the gradient are immediately removed. The objective of this stage is to move the cage vertices in the direction given by the gradient only if the energy is minimized. To do this, the authors of the CAC use a back-tracking of the value alpha as seen in the pseudo-code 1. Once the energy can no longer be reduced, it is considered that the cage has reached a minimum of the energy and the inner region of the contour deformed by the cage using (3.5) is the final segmentation of the object.

In the undesirable case where the cage does not reach the second stage after a long time, we have added a maximum number of iterations which ends the segmentation when reached returning the current segmentation as the final one.

Algorithm 1 Segmentation algorithm of Cage Active Contours

Require: *Initial Contour*, *Initial Cage*, *Energy Function*
procedure SEGMENT OBJECT(*Image*)
    Contour, *Cage* ← *Initial Contour*, *Initial Cage*
    Stage, $\alpha \leftarrow 1$, Maximum number of pixels per step
    affine contour coordinates= GETAFFINECOORDINATES(*Contour*, *Cage*)
    while STOP CONDITION not reached **do**
        Gradient ← CALCULATE GRADIENT(*Contour*, *Cage*, *Image*)
        if *Stage* = 1 **then**
            Project Gradient on the line from their vertex to the mid point of all the vertices.
            Standardize Gradient
            if Cage is stuck **then** Stage ← 2 **end if**
        else
            Standardize Gradient
            $\alpha \leftarrow$ BACKTRACKING(Gradient, α, Contour, Cage, Image)
        end if
        Cage ← $Cage - \alpha \cdot Gradient$ ▷ Update cage
        Contour ← DEFORM POINTS(*affine contour coordinates*, *cage*) ▷ Deform contour
    end while
    return *Contour*, *Cage*
end procedure

function CALCULATE GRADIENT(Contour, Image)
    Ω_1, Ω_2 ← Retrieve the Interior and Exterior Regions of the *Contour*
    $Model_1$, $Model_2$ ← Fetch inner and outer models
    Gradient ← Gradient of the Energy with respect to the control points in the *Cage*
    return *Gradient*
end function

function CALCULATE ENERGY(Contour, Cage, Image)
    Ω_1, Ω_2 ← Retrieve the Interior and Exterior Regions of the *Contour*
    $Model_1$, $Model_2$ ← Fetch inner and outer models
    Energy Value ← Value of the *Energy function* with respect to the Contour
    return *Energy Value*
end function

function BACKTRACKING(Gradient, α, Contour, Cage, Image)
    Previous Energy = CALCULATE ENERGY(Contour, Cage, Image)
    do
        Contour ← DEFORM POINTS(*affine contour coordinates*, $Cage - \alpha \cdot Gradient$)
        Energy = CALCULATE ENERGY(Contour, $Cage - \alpha \cdot Gradient$, Image)
        $\alpha \leftarrow \alpha - \epsilon$
    while Energy > Previous Energy **and** $\alpha > 0$
    return $\alpha+\epsilon$
end function

function GETAFFINECOORDINATES(*Points*, *Cage*)
    affine coordinates ← Obtain the *Points*' *mean value coordinates* with respect to the *Cage*.
    return *affine coordinates*
end function

function DEFORM POINTS(*affine coordinates*, *cage*)
    deformed points ← Deform the points using equation (3.5)
    return *deformed points*
end function

4 Energy enhancement and extensions to color spaces in CAC

In order to improve Cage Active Contours and to extend their applicability to the most used color spaces in Computer vision: RGB, RGB-D and HSV or HSI, we have to consider different types of color coordinate systems. So far Cage Active Contours have only been applied to gray-scale images in both 2D [30] and 3D [72] scenarios. In this case an image is a function defined as:

$$\begin{aligned} I\colon \mathbb{R}^D &\to \mathbb{R} \\ p &\longmapsto I(p) \end{aligned} \tag{4.1}$$

The advantage of this type of image lies in the simplicity of having the information in a single value which interestingly enough is also highly interpretable by humans. However, this has two negative consequences: first is that color information is lost and secondly, since image intensity is directly affected by illumination, methods that rely only on this model are prone to fail under different settings. In this thesis, we propose to both enhance the Gaussian energy model as well as extend it to work on the RGB color space, and the Mean energy model to work on the Hue component of HSV. The image functions of RGB and the HSV/HSI is expressed respectively in (4.2) and (4.3) .

$$\begin{aligned} I\colon \mathbb{R}^D &\longmapsto \mathbb{R}^3 \\ p &\longmapsto I(p) = (r, g, b) \end{aligned} \tag{4.2}$$

$$\begin{aligned} I\colon \mathbb{R}^D &\longmapsto \mathbb{S}^1 \times \mathbb{R}^2 \\ p &\longmapsto I(p) = (h, s, i) \end{aligned} \tag{4.3}$$

Properties of a good Energy

Since we will be defining new energies we have to consider which features characterize a good energy function. We will reffer to these throughout the rest of the text:

E.1 Differentiable

E.2 Few local minima.

E.3 Little dependence on the starting contour

The first property **E.1** ensures regularity and allows the use of mathematical tools to apply iterative optimization methods such as gradient descent. Some methods even require twice differentiable functions in order to use the second derivative for estimating the step direction and size.

The second property **E.2** refers to the number of points in which an energy function is locally stable, where the derivative is zero. If there are many, there is a higher risk of the gradient getting stuck and not converging to a better minimum. In order to converge to a global minimum, few local minima are prefferable.

The third property **E.3** ensures stability and requires less interaction. If the results vary greatly with respect to the initial curve, small variation leads to very different results, causing the method to be chaotic and unstable.

4.1 Gaussian Energy in RGB

In this section we present the Multivariate Mixture Gaussian Energy which enhances the Gaussian energy of 3.3.2 with three improvements:

- Creating a Mixture Gaussian based Energy.
- Generalizing this energy to higher dimensionality.
- Introducing a seed in the inner and outer model.

So far, the energies implemented in the CAC can only capture a region's model with a single component, being either the mean value of a region (mean Energy) or a normal distribution of the values (Gaussian Energy), or maximize the difference between distribution of values of each region (Histogram Energy), with no regard for the resulting object to detect. What these energies have in common is that their strategy is to polarize the values in each region. Although this proves to be useful in some cases, it is very limiting when trying to segment objects and background that have multiple components. Furthermore, by sampling the model of each region at every iteration, not only is it computationally expensive but also the contour relies on a good initialization to capture the description of each region.

The proposed energy attempts to solve these problems by introducing initial information about the object and background through *seeds* enhancing **E.3** and allowing for each region to capture various dominant values inside an image so that in each region different colors or shades can have a representation proportional to their presence.

In order to best capture a model, we need to define a density function which is differentiable in the Color Space so that we are able to minimize it using gradient descent (**E.1**) and that allows us capture best the distribution of values. With these properties, the mixture Gaussian probability density is a candidate that satisfies both of these criteria since any other continuous (and therefore, all differentiable functions) distributions can be expressed as a Mixture of Gaussians given enough components [67, 12]. Moreover, the mixture Gaussian inherits good properties from its normal components as well as a number of good methods to estimate their parameters such as the expectation-maximization [71]. However, instead of using directly the Mixture Gaussian probability density function, we use its logarithm to smoothen the exponential effect. This trick, commonly used in the literature [2, 40], is also used in the Gaussian Model defined in [30].

4.1.1 Multivariate Gaussian Mixture Energy

With these criteria, we present the Multivariate Gaussian Mixture energy (MGME), which is expressed in the following way:

$$E_{\text{MixtGauss}} = \sum_{h=1}^{2} \sum_{p \in \Omega_h} -\log(P_h(p)) \tag{4.4}$$

where P_h the Mixture Gaussian probability density function of the value of pixel p to belong to region h:

$$P_h(I(p)) = \sum_{i=1}^{r_h} \frac{w_i}{2|\Sigma_i|\sqrt{2\pi}} e^{-\frac{(I-\mu_i)^T \Sigma_i^{-1} (I-\mu_i)}{2}} \tag{4.5}$$

This probability density function has r_h normal components, each of which has a mean μ_i a covariance matrix Σ_i, and a weight w_i such that $\sum_{i=1}^{r_h} w_i = 1$ where $w_i \geq 0$ for $i \in \{1, 2, \ldots, r_h\}$.

4.1.2 Analysis of the Energy

In order to have a better understanding of this energy, we must look in which circumstances it has a minimum. Intuitively, this energy function tries to include points in a region whose values will contribute most to the energy value by being in it than in the opposite region. In this case, this contribution is the log-likelihood of the pixel belonging to the region in question. More formally, we can state that a minimum is reached when a slight movement of the contour implies a loss of pixels in each region whose values have a higher log-likelihood of belonging to the regions' model than the other.

Again we calculate the gradient with respect to all the control points to minimize the energy using gradient descent:

$$\nabla_{v_j} E_{\text{MixtGauss}} = \sum_{h=1}^{2} \sum_{p \in \Omega_h} -\frac{1}{P_h(I(p))} \nabla_{v_j} P_h(I(p)) \tag{4.6}$$

is the mixture Gaussian defined by the seed in region h which has r Gaussian components. The gradient is

$$\nabla_{v_j} P_h(I(p) = \sum_{i=1}^{r_h} \left(\frac{w_i e^{-\frac{(I-\mu_i)^T \Sigma_i^{-1} (I-\mu_i)}{2}}}{\sqrt{|\Sigma_i|}\sqrt{2\pi}^k} (\mu_i - I(p)) \Sigma_i^{-1} \cdot \nabla I(p) \varphi_j(p) \right) \tag{4.7}$$

4.1.3 Descriptive analysis of the gradient's components

In this section the Mixture Gaussian energy function will be torn apart to its simplest components to understand the role each component plays in the gradient descent. For our example we will use image 4.1. The white patch is the object to segment, the rest is the background, the circle in white is the initial contour, and the cage in blue the initial cage. In green, the next cage is depicted with a direction that has been exaggerated by multiplying by 10 the step size. This will give us an idea of the direction of the cage.

For simplicity's sake, we take the particular case of the mixture energy in gray-scale. The energy in this case is expressed in the following way:

$$E_{MixtGauss} = \sum_{h=1}^{2} \sum_{p \in \Omega_h} -\log(P_h(p)) \tag{4.8}$$

where

$$P_h(I(p)) = \sum_{i=1}^{r_h} \frac{w_i}{2\sigma_i\sqrt{2\pi}} e^{-\frac{(I(p)-\mu_i)^2}{2\sigma_i^2}} \tag{4.9}$$

is the mixture Gaussian density function that best models the seed in region h and which has r_h Gaussian components.

When we derive in order to apply gradient descent we get

$$\nabla_{v_j} E_{MixtGauss} = \sum_{h=1}^{2} \sum_{p \in \Omega_h} -\frac{1}{P_h(I(p))} \nabla_{v_j} P_h(I(p) \tag{4.10}$$

where

$$\nabla_{v_j} P_h(I(p) = \sum_{i=1}^{r_h} \frac{w_i(\mu_i - I(p))}{\sigma_i^3\sqrt{2\pi}} e^{-\frac{(I(p)-\mu_i)^2}{2\sigma_i^2}} \cdot \nabla I(p)\varphi_i(p) \tag{4.11}$$

In order to understand the behavior of the gradient, we have stripped it down to examine its simplest elements: Firstly, we only focus on the gradient with respect to the control point v_i. Secondly, we only consider the influence of the internal energy. Thirdly, we observe the contributions on the x components of each point $p = (x, y)$ on the gray dashed line crossing image 4.1. Before summing over all of the points on the region, we plot their contribution to the direction of v_i.

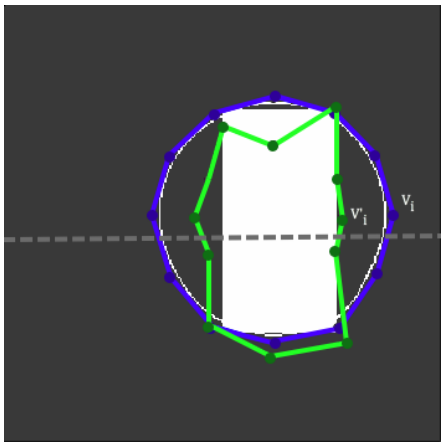

Figure 4.1: First step of the iteration in the segmentation. The initial cage in blue, the contour in white and the next cage in green with ten times the learning rate of step 1.

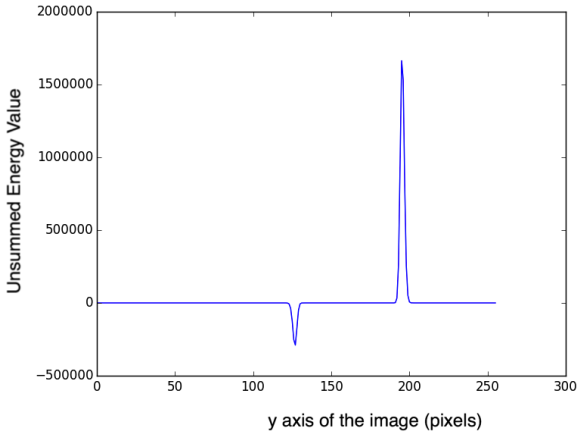

Figure 4.2: Contribution of the points in a slice of image 4.1 on the direction of the gradient with respect to v_i of the x axis.

In figure 4.2, we have that the contribution of each every pixel is zero except at two peaks that corresponds to the change from black to white and vice versa, this show the contribution of the image gradient to the gradient of the energy: a priori we have that pixels with the same intensity but a greater gradient will have more influence. This analysis would be correct if both peaks had the same magnitude. However as we can see points that are closer to vertex v_i have more influence than those that are not. This is due to the factor $\varphi_i(p)$ in (4.11) whose value is inversely proportional to the distance from the point to the control point v_i (see equation (3.2)).

Another point must be made with respect to the contribution of each region to the energy gradient. In (4.10) the gradient of the energy is divided by the probability of a point's membership to a particular region. Therefore if one region has points which belong to its probabilistic model, the influence of those points would be smaller than those that do not belong in the region. This can be understood by stating that, a region will not move when its points conform to the model defined in the region. Per contra, they will do so when it has points who do not belong in that region.

4.1.3.1 Numerical issue

The implementation of the Mixture models present a very challenging numeric problem when it comes to computation. This is because unlike single Normal density functions where we have $\ln e^y = y$, this cannot be simplified analytically in the case of the logarithm of a sum of normal density functions and therefore when calculating the gradient in (4.10), we divide by $P_h(I(p))$. When the probability is zero or very small, it is numerically treated as zero. For example e^{-745} is not zero while e^{-746} is. However, when divided by zero, e^{-710} and smaller give an indetermination.

This numerical problem emerges when calculating the probability of a point p's value in (4.9) and the exponent is very small. The i^{th} component of the density function h is small or zero if the difference between its mean μ_i and the pixel value $I(p)$ are very different and its variance σ_i^2 is small. The workaround that has been proposed is the use of an $\varepsilon > 0$ to substitute small probabilities. Thus, the computation of (4.10) is done in the following way:

$$\nabla_{v_j} E_{MixtGauss} = \sum_{h=1}^{2} \sum_{p \in \Omega_h} -\frac{1}{P_h^{\varepsilon}(I(p))} \nabla_{v_j} P_h(I(p) \tag{4.12}$$

where

$$P_h^{\varepsilon}(I(p)) = \begin{cases} P_h(I(p)) & \text{if } P_h(I(p)) > \varepsilon \\ \varepsilon & \text{if } P_h(I(p)) \leq \varepsilon \end{cases} \tag{4.13}$$

This way, we avoid causing an indetermination when dividing. The problem with this workaround, is that if we give a very small ε we will get that points that had practically no influence now are divided by very small numbers and thus increasing their influence greatly. This is why in the experiments we will add it as a parameter and search for an optimal value ε.

4.2 Mean Energy in Hue

The literature on color image segmentation generally agrees that the main two color models used in segmentation are the RGB and the HSI/HSV (Hue Saturation and Intensity/value) [19, 17].

The RGB model was created to mimic the human eye's response to three main light frequencies red, green and blue. However, this is problematic in terms of intuitiveness since it has no intrinsic relation to the natural color properties, neither to human interpretation of color. In other words we cannot intuitively decompose any color into these three components [47]. This is why in computer vision applications where the idea is to mimic a human's interpretation, other color spaces such as HSI or HSV are used [35]. The first component represents the Hue value, an intrinsic properties of surfaces of constant color which remain practically invariant under changes in illumination [55]. This is the reason which motivates us to extend the energies in CAC to work in HSI. However this space has the drawback of being cylindrical (see figure 4.3) because of the cyclic nature of the Hue component. This impedes us from reusing the previous energies as they are. This cyclic channel brings us to redefine three aspects: Difference between values, the gradient of the image and the descriptive model of each region. In this thesis, we have only developed this method to work on its homologous mean energy described in section 3.3.1 yet we do lay the foundation for further investigation to apply the equivalent energies in this domain.

4.2.1 Converting the RGB model to HSV

Given the intensities of the three primary components R, G, and B of an RGB color, we can find its HSI or HSV representation with the following transformation:

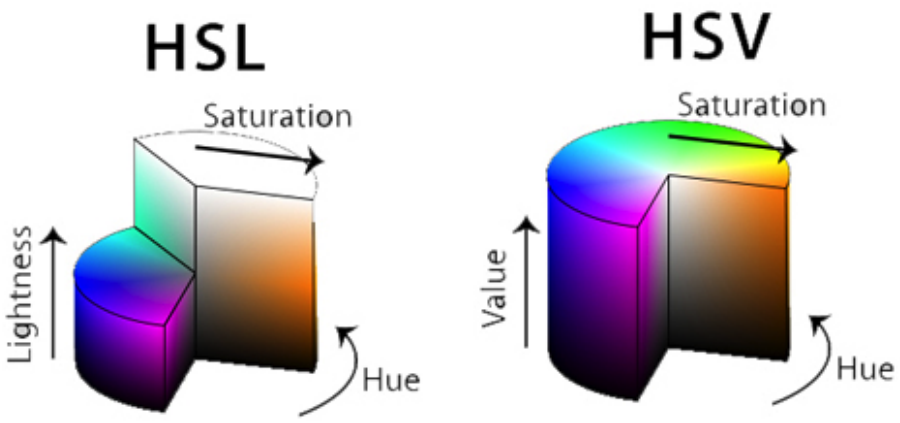

Figure 4.3: Different cylindric colors spaces (image from [21])

$$\begin{bmatrix} Y \\ C_1 \\ C_2 \end{bmatrix} = \begin{bmatrix} \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & -\frac{\sqrt{3}}{2} & \frac{\sqrt{3}}{2} \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{4.14}$$

HSI values are given as:

$$I = Y, S = \sqrt{C_1^2 + C_2^2} \tag{4.15}$$

$$H = \begin{cases} arccos(C_2), & C_1 \geq 0 \\ 2\pi - arccos(C_2), & C_1 < 0 \end{cases} \tag{4.16}$$

In the case of HSV, the first two components Hue and Saturation are the same as in the HSI model but the *value* component is defined as $V = max(R, G, B)$. In image 4.3 we can see the cyclic representation of both models.

Now let us define the homologous mean energy in 3.3.1 in Hue values instead of gray-scale. The distance between two hue values H_1 and H_2 can be defined as

$$d(H_1, H_2) = \begin{cases} \mid H_1 - H_2 \mid, & \mid H_1 - H_2 \mid \leq \pi \\ 2\pi - \mid H_1 - H_2 \mid, & \mid H_1 - H_2 \mid > \pi \end{cases} \tag{4.17}$$

The directed distance between two hue values can be defined as:

$$\vec{d}(H_1, H_2) = \begin{cases} H_2 - H_1, & \mid H_2 - H_1 \mid \leq \pi \\ H_2 - H_1 - 2\pi, & \mid H_2 - H_1 \mid > \pi, H_2 \geq H_1 \\ 2\pi + H_2 - H_1, & \mid H_2 - H_1 \mid > \pi, H_1 > H_2 \end{cases} \tag{4.18}$$

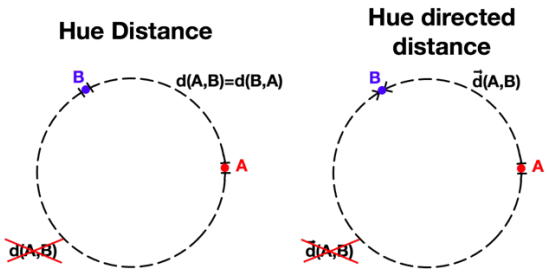

Figure 4.4: Visual representation of hue distance and directed distance or a unit circle.

The mean of cyclic values can be calculated in the following way [74].

$$\mu_h = atan2\left(\frac{1}{\mid \Omega_h \mid}\sum_{p\in\Omega_h} sin\alpha(p), \frac{1}{\mid \Omega_h \mid}\sum_{p\in\Omega_h} cos\alpha(p)\right) \tag{4.19}$$

Figure 4.4 show the representation of the hue distance and the hue directed distance.

4.2.2 Mean Hue Energy

Working on the Hue channel presents challenges in the implementation of otherwise simple procedures. We have therefore chosen to extend the simplest energy in CAC: the mean Energy. The idea, again is to stabilize the color in each region.

$$E_{\text{Hue}} = \sum_{h=1}^{2}\sum_{p\in\Omega_h} \frac{1}{2}\vec{d}(\alpha(p), \mu_h)^2 \tag{4.20}$$

$$\frac{\partial arctan2(x,y)}{\partial x} = \begin{cases} \frac{1}{1+(\frac{y}{x})^2}\frac{-yx'}{x^2}, & x \neq 0 \\ 0, & x = 0, y \neq 0 \\ undefined, & x = y = 0 \end{cases} \tag{4.21}$$

$$\frac{\partial arctan2(x,y)}{\partial y} = \begin{cases} \frac{1}{1+(\frac{y}{x})^2}\frac{y'}{x}, & x \neq 0 \\ 0, & x = 0, y \neq 0 \\ undefined, & x = y = 0 \end{cases} \tag{4.22}$$

Using equations (4.21) and (4.22), we get that the derivative of the mean with respect to a control point is defined as:

$$\nabla_{v_i}\mu_h = \begin{cases} \left(\frac{\frac{1}{|\Omega_h|}\sum_{p\in\Omega_h} cos\alpha(p)}{x^2+\left(\frac{1}{|\Omega_h|}\sum_{p\in\Omega_h} cos\alpha(p)\right)^2}\varphi_i, \frac{1}{1+\left(\frac{1}{|\Omega_h|}\frac{\sum_{p\in\Omega_h} sin\alpha(p)}{x}\right)^2}\varphi_i \right) & x \neq 0 \\ (0,0), & x = 0, y \neq 0 \\ undefined, & x = y = 0 \end{cases} \tag{4.23}$$

However, by including a seed to learn the mean of each region beforehand, we get a simpler gradient form:

$$\nabla_{v_i} E_{\text{Hue}} = \sum_{h=1}^{2}\sum_{p\in\Omega_h} \vec{d}(\alpha(p),\mu_h)\nabla_{v_i}\alpha(p)\varphi_i(p) \tag{4.24}$$

The key in (4.24) is how to calculate the gradient of the hue value $\nabla_{v_i}\alpha(p)$. This can be done using simple central differences. However, we have to regard difference between values as a directed distance. A way of doing this optimally is by considering every 3 by 3 pixel patch and substituting its Hue value with the directed distance of the Hue values of the pixel to the central pixel whose gradient we want to calculate. We then proceed to applying the gradient with a Sobel convolution on the patch.

4.2.3 Extending Hue energies

In this point, we leave the field open to define more complex energies on the Hue channel as well as extend these to the whole HSI or HSV color spaces. For example, the equivalent mixture Gaussian energy for the Hue channel would be

$$E_{\text{MixtGaussHue}} = \sum_{h=1}^{2}\sum_{p\in\Omega_h} -log(WP_h(I(p)))) \tag{4.25}$$

where $WP_h(I(p))$ is the wrapped Wrapped Mixture Model, which defines the . In order to extend to the whole HSI or HSV field, we have to turn to directional statistics, which is mainly concerned with observations which are unit vectors in the plane or in three-dimensional space [48] and in particular Cylindrical data [5].

4.3 Implementation

In figure 4.5 we show the pipeline of our implementation of image segmentation using CAC. We begin by appliying a Gaussian filter on the image to smoothen the image. We choose this as a pre-processing step because we have observed that it enhancees the algorithm in three different points: when calculating the gradient in the image, as it is typically done, when extracting the model of each region, by smoothening out hard peaks and therefore expanding the covariance when it is too small thus reducing the number of points that present the numerical problem described in 4.1.3.1, and finally by avoiding sharp edges and therfore contributing to the numeric problem described in 4.1.3.

Next we allow the user to interact twice. Once for choosing the initial contour and the initial cage, and the other to input the parameters of the Gaussian Mixture Models that model the inner and outer regions $\Omega_1$and Ω_2. In the experiment section, for sake of simplicity, this last step can be done automatically through an erosion of the ground truth to model Ω_1 and a dilation to model Ω_2.

When we have the image, the initial conditions and the model, we apply our selected CAC algorithm to obtain a segmentation. The only result will be a cage. Remember that with the initial conditions and the final cage we can obtain the final segmentation so in a way the amount of memory used is reduced. From this we will extract various applications in section 6.

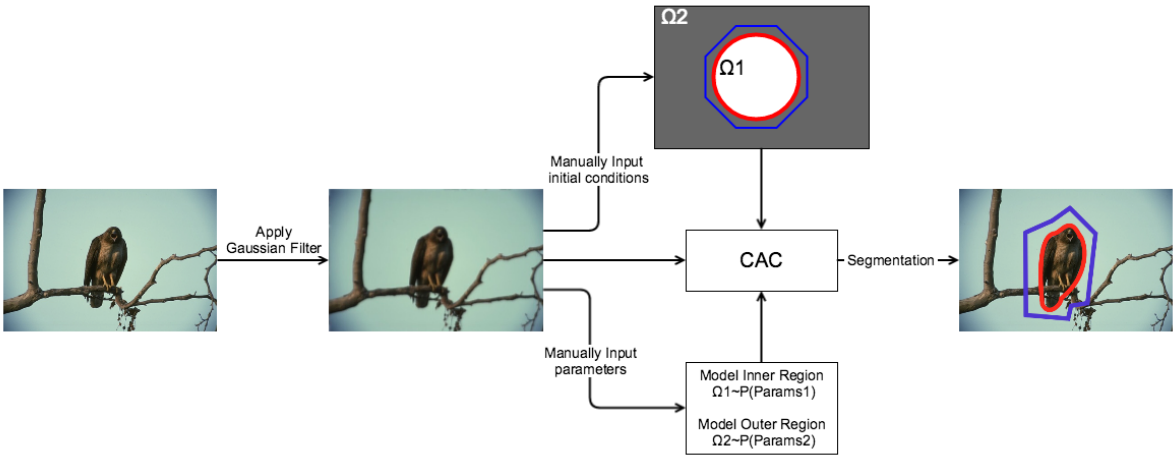

Figure 4.5: Flow diagram of CAC with learning model.

4.3.1 Model Initialization

The initial statistics describing each region can either be inputed manually by introducing the parameters of a distribution or by automatically learning from a seed, a subsample of a region in the image from which it will extract the model which best fits de values of that region. The fitting of the model is done through Expectation Maximization (EM) with different number of Gaussian components by incrementing, fitting, and keeping the model with the lowest Schwarz or Bayesian Information criterion (BIC) .

$$\mathrm{BIC} = k \cdot \ln(n) - 2 \cdot \ln \hat{L} \tag{4.26}$$

where $\hat{L}$ is the maximum likelihood of the Model found through EM, n is the number of data points and k ie the number of free parameters [70]. The advantage of the BIC over other information criteria indexes such as Akaike's (AIT) is that they take into consideration the number of points in the data and thus over-fitting can be avoided in case of having a very small sample, which is the case of the seeds.

The Expectation Maximization module from the Scikit-Learn package has been used in our code [57].

4.3.2 Code

The code, contains a `CAC` class which has the method segment represented in pseudo-code 1, energy which returns the energy. This class is inherited by all the different implementation of

Our code which can be found in `https://github.com/Jeronics/cac-segmenter` we implements six different Energies. From the original the original CAC article[30] our implementation of the Mean Energy and the Gaussian Energies can be found in the respectively in the `MeanCAC` and `OriginalGaussianCAC` classes of in our code. Our Multivariate Mixture Gaussian energy can be found in `MultiMixtureGaussianCAC`, and its particular cases, for gray-scale images, and with a single Gaussian component for color and gray-scale can be found respectively in `MixtureGaussianCAC`, `MultiVariateGaussianCAC` and `GaussianCAC`.

Furthermore, we have used two wrapped functions in C programming in order to calculate the Mean Value Coordinates and detect the inner and outer regions of a contour. These functions were written by by the authors in [30].

5 Experiments

In this section, we will use both quantitative as well as qualitative experiments to test our methods, compare them and draw conclusions about their performance and their properties.

5.1 Datasets

We used three datasets in order to test our methods. The first one is a Synthetic dataset which we created to verify, on one hand, that the models did in fact work in controlled and regular conditions and, on the other, to carry out simple qualitative experiments with which to verify certain assumptions. Some of these images we created can be seen in figure 5.1.

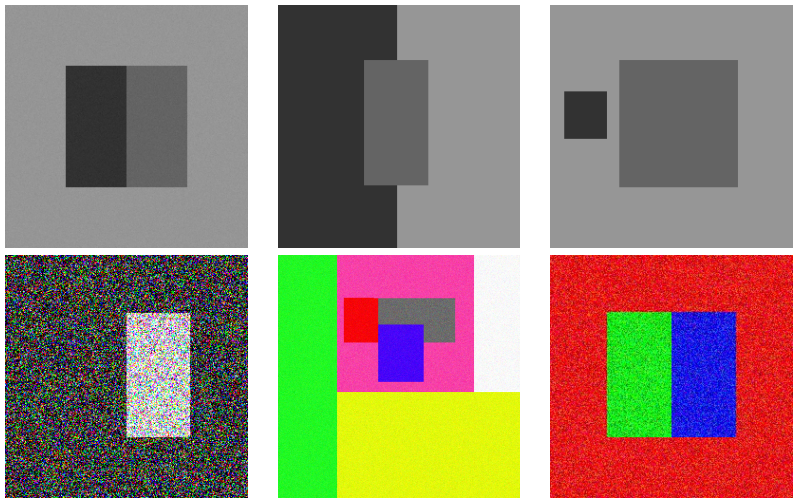

Figure 5.1: Images from the Synthetic dataset

The other two datasets are a subsets of two existing datasets of real images. The first one is a subset of 40 images from the Single Object Database (AlpertGBB07) [3]. This dataset is characterized by having well defined backgrounds from the foreground. We discarded those images that we did not consider fit the criteria for which cage active contours were created, that is, images with single connected objects with no holes and visually distinct from the background. In figure 5.2 shows images from the original dataset, the image of a chain is an example of an object we removed from the dataset because it does not meet the criteria of the CAC of having no relevant holes. This dataset from the Weizmann Institute of Science provides human-segmentations as ground truth. It is worth noting, however, that the ground truth segmentations are done so manually and independently by three individuals, thus result in contradictions at

times. We have chosen the one we have thought best met the CAC's criteria.

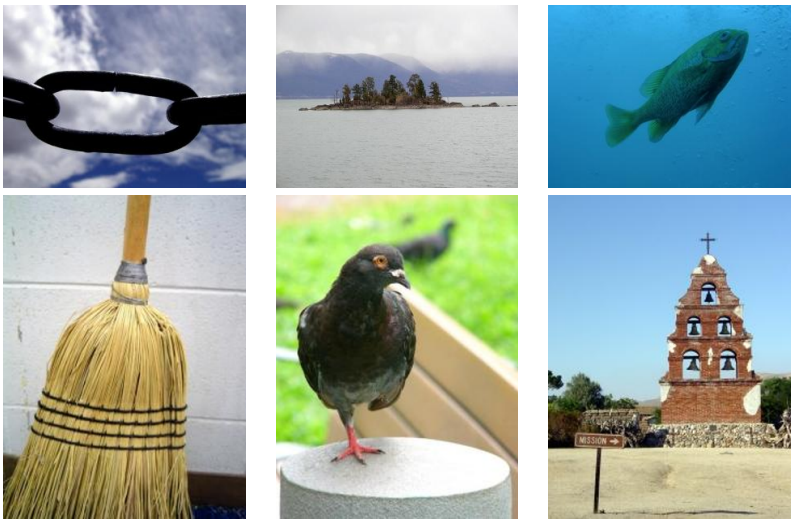

Figure 5.2: Images from the Single Object Database (AlpertGBB07)

The second dataset is the Berkeley Segmentation Dataset and Benchmark (BSDS300)[49]. This dataset consisting of 300 real images which are much more complex than the Single Object Dataset since they are chosen in order to evaluate image segmentation in general and not object segmentation. Nevertheless we have chosen a subset of 20 images from this dataset that was used in [62] and whose ground truth they provide for object segmentation. In figure 5.3, some images from this sub-dataset are shown so that its complexity can be appreciated.

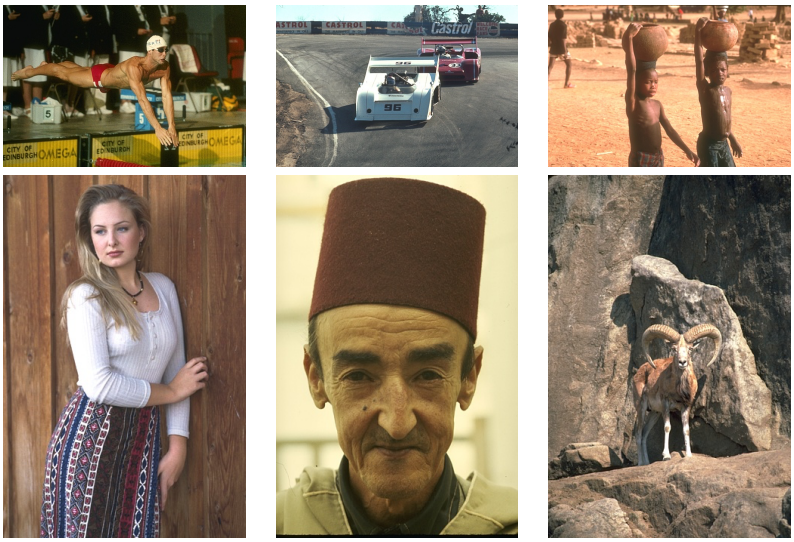

Figure 5.3: Images from the BSDS300 subset

5.2 Validation

5.2.1 Evaluation Measures

The best measure to evaluate a segmentation depends largely on the consequences that any error in the segmentation might have. In practice this depends completely on the application. In medical image, for example, the segmentation of the a brain lesion can serve a variety of purposes. When monitoring size, an error measure that is sensitive to area would be more appropriate. On the other hand, if when treating a tumor, shape fidelity of the segmentation is key in order to know where to apply radiation. In this case, having a measure that is more focused on outline would avoid targeting healthy tissue or not targeting a malicious one [23].

Overlap ratio measures usually range from 0 to 1, from least to most congruent. They are sensitive to misplacement of the segmentation label, so in general they do not capture shape fidelity. Among these measures, the most popular are the Sørensen-Dice similarity index and the Jaccard overlap ratio. The Jaccard Index is numerically more sensitive to mismatch when there is reasonably strong overlap. Dice values are higher for the same pair of segmentations [23].

Let X be the segmentation region and Y the ground truth segmentation region. The Jaccard index is expressed as

$$c_{Jaccard}(X,Y) = \frac{|X \cap Y|}{|X \cup Y|} \tag{5.1}$$

The Sørensen-Dice coefficient is

$$c_{Sorensen-Dice}(X,Y) = 2\frac{|X \cap Y|}{|X| + |Y|} \tag{5.2}$$

Another interesting metric is the Hausdorff distance [7]. The idea of this metric is that a segmentation is only as good as its worst good point. It is expressed as follows:

$$d_H(X,Y) = max(\vec{d_H}(X,Y), \vec{d_H}(Y,X)) \tag{5.3}$$

where the directed haussdorf distance $\vec{d_H}$ is:

$$\vec{d}(X,Y) = \max_{x \in X} \min_{y \in Y} \|x - y\| \tag{5.4}$$

This last one is used when very high precision and shape fidelity is desired. Its drawbacks include high computational cost, because of the huge number of comparisons it requires, and a very disproportionate penalizations for even a single point [36].

We have chosen to evaluated the Sørensen-Dice coefficient because of its simplicity and use in object image segmentation.

Another form of evaluation which we will use is visual criteria. This can serve in qualitative experiments to evaluate for example whether a segmentation has been carried out successfully or not.

5.2.2 Model Validation

Cage Active contours are adaptive methods with no learning. By adaptive we mean that through a few basic rules, imposed in this case on the Energy and the cage, a certain of intelligence emerges. The more elaborated these set of rules are, the more complex objects it will be able

to segment. From simple rules, a more abstract and complex behavior emerges. Furthermore, CAC have no learning since its performance does not improve the more objects it segments.

Usually in model evaluation there are two main points that we want to know:

- The overall score of a method
- The best model for that method

In our case, the method corresponds to an Energy Function on CAC while a model is a set of parameters. A model, in the case where there is no learning, is evaluated as the mean score result throughout the whole dataset. The best method would then be that which best scores in a dataset.

To evaluate a *method* without over-fitting, we use k-fold cross-validation. This is an iterative process in which a k-th part (fold) of the dataset is taken, different models are evaluated on it such that the model with the best score is considered to be *validated*, and goes on to be evaluated using the rest of the dataset. This is repeated k times with k different parts for model validation. At the end, the score in each fold is averaged thus giving the final score of the method.

5.3 Qualitative Experiments

Using the synthetic dataset we have challenge and compared the different energy functions we have created. We wanted to see whether they would succeed in the segmentation of objects they were created to segment. In oder to check various hypotheses of our proposed method, we decided to compare in a very controlled manner, the previous energies in CAC with our contributions.

It is important to stress the fact that qualitative experiments neither prove anything, nor are they conclusive. What they can do is disprove and help shed light on certain phenomena.

5.3.1 Parameters

The parameters used in this experiments are the ones seen in table 5.1 and a list of their values which we have tested for real images is provided in the Values column.

Parameters	**Symbol**	**Values**
Number of Control points	N	16, 20, 24
Cage-contour ratio	r	1.05, 1.1
Sigma	σ	0.25, 0.5, 0.7
Substitute value [1]	ε	$\exp^{-100}$, $\exp^{-200}$, $\exp^{-300}$

Table 5.1: List of values chosen for each parameter.

The first parameter is the number of points the initial cage has. In [30], the authors test the influence of this parameter in different shapes. They conclude that the more points the better the segmentation in complex objects. We argue that it must be correlated with the maximum number of curvature changes in the curve and we leave this as a possible field of study in section 7.

The cage-contour is explained in detail in section 6.1. This is in relation to the distance between the initial cage and the contour. As we have discussed in section 3.1, this ratio is

[1] This parameter is specific to the Mixture Gaussian Energy

correlated with the degree of influence of the contour with respect to the cage. The bigger the ratio, the lower the cage's influence on the contour (see image 3.3).

Sigma is the standard deviation of the Gaussian filter applied to the image. This filter is applied as a pre-processing step of our implementation as discussed in section 4.3. Through observation we have seen how dependent this parameter is on each image. In fact, in images from the qualitative dataset a much higher sigma value is required than in real images.

Finally, in section 4.1.3.1, we have seen the need for the ϵ this parameter. It is only specific to energies with Mixture Gaussian density functions. Through observation we have seen that this it is quite correlated with the sigma value, however this still remains to be tested and is left as future work (section 7).

5.3.2 Experiment 1: Independence from initialization.

In this first experiment we test the influence of adding a seed to the dependency on the initial contour with respect to the ground truth. To observe this relationship we will test the original Gaussian Energy with no seed against one with a seed. In this way, we can be sure that the results are not conditioned by other improvements.

5.3.2.1 Experiment setup

We use a simple synthetic gray-scale image shown in figure 5.4. The object to segment is the black rectangular region whose values were generated by a Normal distribution of $\mu = 50$, $\sigma = 10$, while the exterior was generated with $\mu = 150$, $\sigma = 10$. We generated a set of circular curves with a diameter of 20 pixels, with centers located on the horizontal axis through the middle of the image and with 10 pixels of separation between neighboring centers.

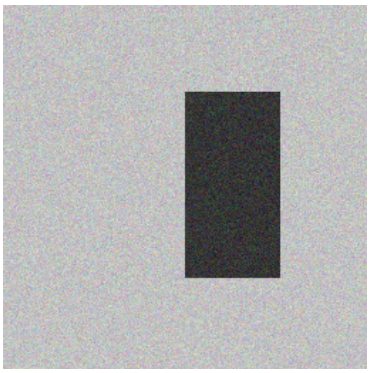

Figure 5.4: Synthetic image used for experiment 1.

5.3.2.2 Results

In figure 5.5, the first column represents the different initializations of the cage-contour configuration where, in each row, it is initialized closer to the solution. In the second and third columns, we have the resulting segmentation of the non-seeded and the seeded Gaussian energy respectively.

As expected, the non-seeded energy does not converge to the solution when the initial configuration is not near to the solution. This stems from the fact that its statistics are re-calculated

at every step and therefore it adopts the interior region as the description of the object it wants to segment. The peculiar shape adopted by the curve can be explained by the effect of the outer region component in the energy. As we have seen in section 4.1.3, a region energy only influences the energy gradient if it contains significant changes in the image's gradient in points whose probability of being in that region is small. On the other hand, the outer region does meet this condition, thus driving the gradient towards the edge of the object.

Conversely, in the case of the seeded energy, the cage converge in general. Again the outer region motivates the energy's gradient direction. Moreover, what is interesting in the seeded energy is the reluctance of the outermost points of the cage to move towards the object. The parametrization using mean value coordinates adds a component of locality to the descent, as we described in 4.1.3, which limits the influence of distant points on the gradient. This "short-sightedness" of the cage already complicates the improvement of property **E.3** of a good energy. Once the points are close enough to detect the gradient however, their path is blocked by the cage constraint which avoids self-intersection.

5.3.3 Experiment 2: Interpreting multiple regions.

In order to test the capability of a region to capture multiple values, we test the Mixture Gaussian energy. Again, we use only the gray-scale version of this energy to compare the improvement against the Gaussian energy (both with seeds), to see whether the improvement works and to see how far we can push it.

5.3.3.1 Experimental setup

The image used (figure 5.6) has three regions which from darkest to lightest have been generated with normal distribution with parameters ($\mu = 150$, $\sigma = 10$), ($\mu = 100$, $\sigma = 10$) and ($\mu = 50$, $\sigma = 10$) respectively. The object which we wish to segment is the second region (rectangle on the right) which has values between the other two regions. This should be problematic for the Gaussian Energy since it will most likely polarize the image's values.

In this experiment we will evaluate the Mixture Gaussian Energy's capacity to overcome this problem as well as test its robustness when varying the object's values. The initial contour will be place in the middle of the image with its center between the two inner rectangles.

5.3.3.2 Results

In figure 5.7 we show the results of the normal and the mixture Gaussian energies in the second and third column respectively with different intensities of the object's values. In the plots on the first column, the distribution of the inner and outer seeds are plotted in blue and red respectively. These are shown over the probability of both seeds to show the magnitude of their contributions to the Energy function. As we can see the highest peak corresponds to the region with the highest intensity value while the smaller peaks correspond to the two rectangular regions in middle. It can be observed that in each new row, the increases of the intensity of the object's value's is manifested on the graph as the distribution of the inner region's seed shifts towards higher intensity.

The results of the energy with one Normal distribution immediately confirm our hypothesis of its polarizing effect. In the first three rows, it considers both squares as the inner region despite one of them being in the outer. The mixture Gaussian on the other hand is successful in

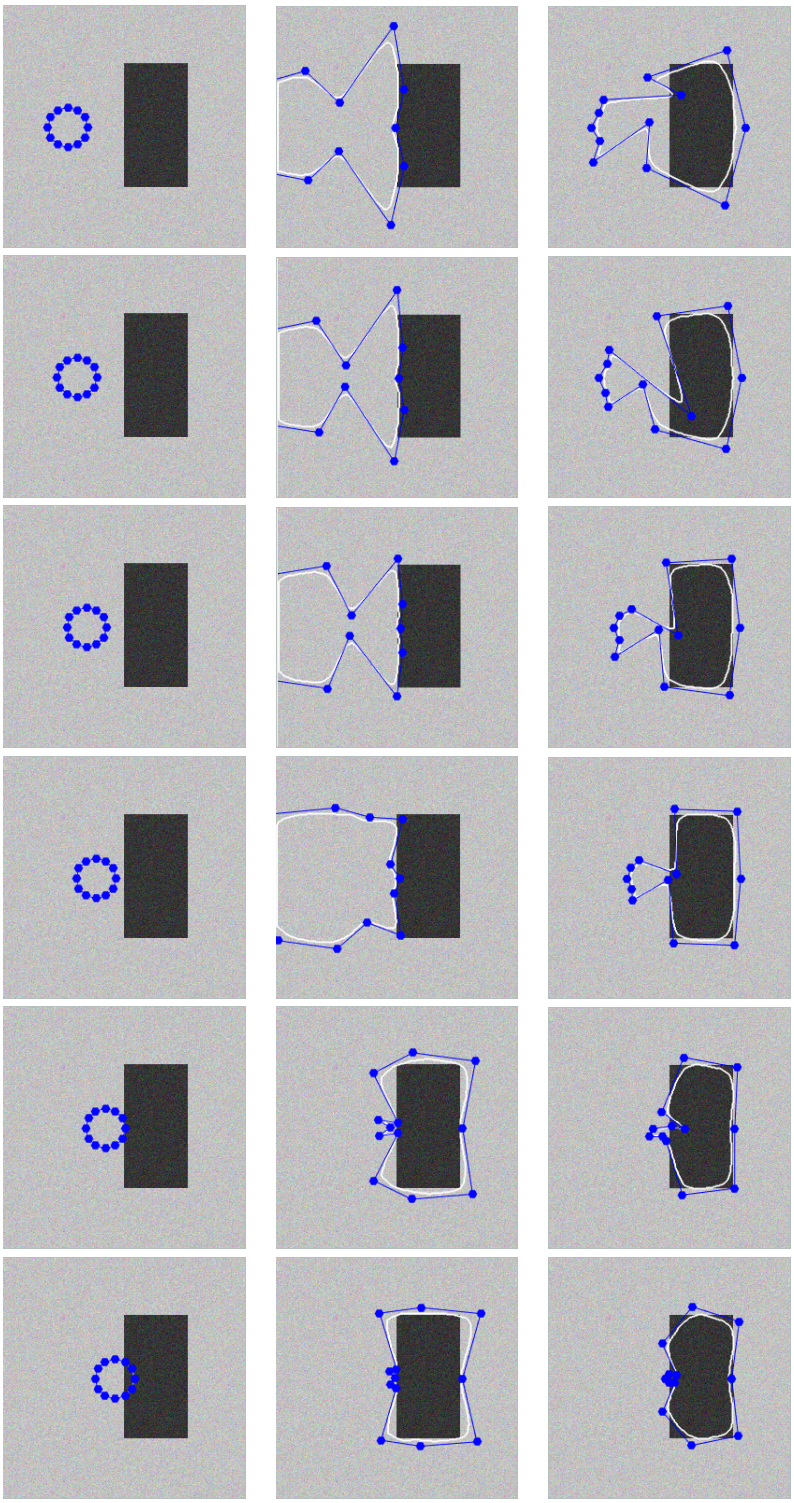

Figure 5.5: From left to right, initialization, segmentation results of the Gaussian Energy with no seed and with seed. Parameters: 12 control points, ratio=1.05, sigma=0. 5, $\epsilon = e^{-200}$)

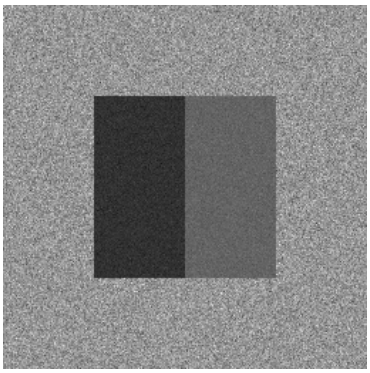

Figure 5.6: Synthetic image used for experiment 3.

segmenting the inner region yet it behaves badly when it's values are similar to one of the outer region's components has the same intensity, because both regions' gradient claim this region for themselves.

5.3.4 Experiment 3: Invariance to illumination

The purpose of this experiment is to see how the Hue Energy can outperform in some cases the Multivariate Gaussian Energy (the same as the Gaussian energy but in RGB) in particular with changes in intensity.

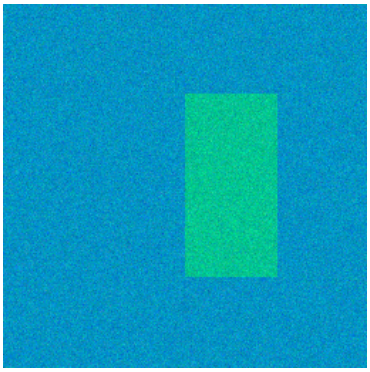

Figure 5.8: Synthetic image used for experiment 2.

5.3.4.1 Experimental setup

In this experiment, robustness to change of illumination is tested with the Multivariate Gaussian Energy and the Hue Mean Energy. The image if figure 5.8 will be used. The interior and the exterior are generated by Gaussians with parameters $\mu = (0, 145, 110)$ and covariance matrix $covars = Id * 10$, and $\mu = (0, 110, 145)$ and $covars = Id * 10$ respectively. In order to test for changes of illumination we linearly decreased the *luminance* (amount of light hitting the sensor from a surface)[73], using Adobe Photoshop. The initial curve is always located inside the object.

5.3.4.2 Results

In figure 5.9, the left column represents the segmentation results of the Hue Mean model whereas the right column represents the Multivariate Gaussian Energy. As we can see, as the image gets darker, the Hue Mean Energy systematically outperforms the other. This is due to the fact that Hue is an intrinsic property of an object, and its transformation perfectly conserves color since the value or intensity is kept in the V/I channel in HSV/HSI.

To a certain extent, these results exceed our expectation. In the case of the last image, also successfully segmented by the mean Hue energy, the object is barely visible. Although the modification was done digitally, we can conclude that in a setting with either very high or very low luminosity, if a slightest hint of color is successfully captured, this method would be able even to surpass human segmentation. This confirms the need for experimenting with energies in different color spaces.

5.4 Quantitative results

We have carried out two different quantitative experiments. In the first one, we compare the different energies in CAC to evaluate our improvements and, in the second, we compare our methods to other existing ones to see where ours stand. We have evaluated these on both real image datasets described in section 5.1 using the Sørensen-Dice coefficient.

5.4.1 Experiment 4: Comparison with different energies in CAC

The goal of this experiment is to quantify the improvements of our energies with respect to the previous ones defined in [30]. To do so, we have evaluated different Energies that contain gradual improvements.

5.4.1.1 Experimental design

In table 5.2, we see a categorical description of each energy we have implemented. The category *Seed* refers to whether or not the energy's description of a region is given by a seed from the foreground or background, as opposed to learning it in each iteration. The *color space* refers to the type of image it accepts, be it gray-scale, RGB or Hue. The *Multi-component* category refers to the energy's capability of learning more than one distribution per region.

Energies	**Seed**	**Color space**	**Multi-comp**
Mean	No	gray-scale	No
Gaussian	No	gray-scale	No
Seeded Gaussan	Yes	gray-scale	No
Mixture Gaussan	Yes	gray-scale	Yes
Multivariate Gaussan	Yes	RGB	No
Multivariat Mixture Gaussan	Yes	RGB	Yes
Hue Mean	Yes	Hue	No

Table 5.2: Different energies evaluated

In order to compare each method, we have run 3-fold cross validation (see sectio 5.2.2) over four different parameters whose values can be seen in table 5.1.

5.4.1.2 Results

In table 5.3, we see the mean Sørensen-Dice coefficient (S-D) expressed in percentage and its standard deviation (std.) with each method.

Energy	**AlpertGBB07**		**BSDS300**	
	S-D (%)	**Std. dev.**	**S-D (%)**	**Std. dev.**
Multivariate Mixture Gaussian	**77.88**	1.3	**55.58**	2.89
Multivariate Gaussian	72.23	3.4	51.06	6.3
Mixture Gaussian	65.15	2.3	46.16	5.70
Gaussian	57.13	3.38	44.90	5.06
Mean	57.11	2.87	37.85	4.34
Hue Mean	56.97	5.78	41.43	4.02

Table 5.3: Comparison of the different CAC Energies.

Immediately, we can see how the Sørensen-Dice coefficient in both datasets is correlated with the number of improvements of each method. The best method is the one with the most complex energy: the Multivariate Mixture Gaussian Energy. The worst however seems to be the Hue mean energy.

It is interesting to see that when comparing the improvements defined in table 5.2, we have that the most impactful improvement is the extension to the RGB color space. This improvement of about 20 points in both the AlperGBB07 and the BSDS300 database unequivocally confirms the positive impact of our main contributions. The energies with multiple components also provide an enhancement in performance but not to such a great extent. This could be a consequence of the numerical problem described in 4.1.3.1. It remains to be studied for future work if other alternative solutions exist apart from our ε. Moreover, the introduction of a seed also seems to have outperformed the non-seeded energies.

Method		
	Mean time (sec.)	**Std.**
Multivariate Mixture Gaussian	38.72	15.74
Multivariate Gaussian	32.22	17.03
Mixture Gaussian	24.18	14.20
Seeded Gaussian	21.16	15.53
Gaussian	22.78	11.96
Mean	21.17	10.02
Mean Hue	25.17	14.06

Table 5.4: Comparison of computational time of the segmentation of CAC with different Energies for 300x225 images.

In terms of computational results, we have seen in table 5.4 that the CAC in general is a very slow image segmentation method. Although the introduction of a seed decreases the computational time by avoiding the computation of the region's description in every iteration, we see that in the Multivariate Mixture Gaussian energy, the Expectation-Maximization is extremely slow.

5.4.2 Experiment 5: Comparison with other methods

Our motivation in this experiment is to place our contribution's results into context with respect to other similar methods in the literature. In order to do this we will compare our methods with other existing ones to see how well they perform and to get an idea of how much of a leap over the previous CAC Energies we have improved.

5.4.2.1 Experimental design

We have taken three active contour methods and used their implementation in Creaseg [25], a free software for the evaluation of image segmentation techniques. These methods are: the Geodesic Active Contours presented by Vicent Caselles [13], the Chan&Vese [16], and the Shi [65].

The Method by Caselles, is a level set based on object boundary detection energy. Like all level-set methods the contours can contract, expand, merge and split. In this particular this method adds restriction on the curve based on geodesic (minimal) distances. It has proven to be robust in boundary detection with badly defined gradients as well.

The Chan&Vese provide a stable method which allows for detection with object whose boundaries are not well necessarily defined by a gradient. The energy used, is the mean energy in (2.4) on which the CAC's was inspired.

The Shi method is a level set with region-based energy that can be implemented in real time. That is, it can accomplish a good result in fewer iterations.

We have decided to use the Chan&Vese, the Caselles and the Shi methods because in the original paper of Creaseg [25], these are reported to have the best results.

Like in [25], we have used the default parameters given by each author to compare with our algorithms. We on the other hand have performed a 3-fold Cross validation on the parameters as described in the previous experiment. We have decided to compare them to the Gaussian Energy, and the Multivariate Mixture Gaussian Energy (each described in table 5.2).

5.4.2.2 Results

In table 5.5, we see the mean Sørensen-Dice coefficient and its standard deviation with each method. Our Multivariate Mixture Gaussian Energy scored best in the AlpertGBB07 dataset and third best in the BSDS300 dataset. These positive results were expected given that it uses RGB information while the methods from Creaseg use gray-scale images. For this reason we have also decided to show the Mixture Gaussian energy which is the equivalent energy in this color space. In this case, the Shi and the Caselles method were outperformed in The AlpertGBB07.

In terms of computational time, we see that the Caselles and the Chan&Vese are extremely fast while the Shi, that is supposed to be fast, took the longest because of the default number of iterations in the Creaseg Implementation.

Method	AlpertGBB07		BSDS300	
	S-D. (%)	Std.	S-D (%)	Std.
ChanVese	70.71	14.14	64.45	14.59
Shi	61.34	20.23	52.15	21.50
Caselles	58.68	17.02	**63.15**	13.76
CAC: Multivariate Mixture Gaussian	**77.88**	1.3	55.58	2.89
CAC: Mixture Gaussian	65.15	2.3	44.90	5.06
CAC: Hue Mean	56.97	5.78	41.43	4.02

Table 5.5: Comparison of the best segmentation Energies in CAC with other existing related methods.

Method	Mean time (sec.)	Std.
Caselles	3.47	0.69
ChanVese	3.50	0.24
Shi	114.52	113.84
CAC: Multivariate Mixture Gaussian	38.72	15.74
CAC: Mixture Gaussian	24.18	14.20
CAC: Gaussian	22.78	11.96

Table 5.6: Comparison of computational time in 300x225 images of the best CAC energies with other existing related methods in 300x225 images.

5.5 Specific examples

In this section, we are going to discuss some of the good and bad results we obtained using specific example images and visual evaluation. These results were obtained by the Multivariate Mixture Gaussian energy's best parameter (Number of control points=20, ratio=1.1, sigma=0.25, $\epsilon = e^{-200}$)

In figures 5.10 and 5.11, we see two images which were exclusively successful using the Multivariate Gaussian Mixture Energies. In the image of the purse, the spotted texture and the two background colors are perfectly captured using this energy, where as in the image of the log, the different textures are also reasonably well interpreted.

The image of the broken tree, however, displays an irregular behavior at the bottom ot the image. We have observed that this phenomena is common among images whose objects are cut by the margin of the image. Figures 5.12 and 5.13 are also examples of this. Following the discussion in section 4.1.2, we have mentioned that a region only contributes to the gradient if it has pixels whose values do not belong to the probability distribution of the region. In the case of these images then, the inner region has virtually no effect on the gradient because it resides inside the object. The external energy, does not respond either because it has no external component which belongs to the outer region and thus does not pull the vertex outside of the image because there is no change in gradient which motivates it into doing so.

The images in figure 5.14 show some positive results of CAC. As we can see, CAC are not meant for high precision segmentation but rather they provide a smooth general contour of the image.

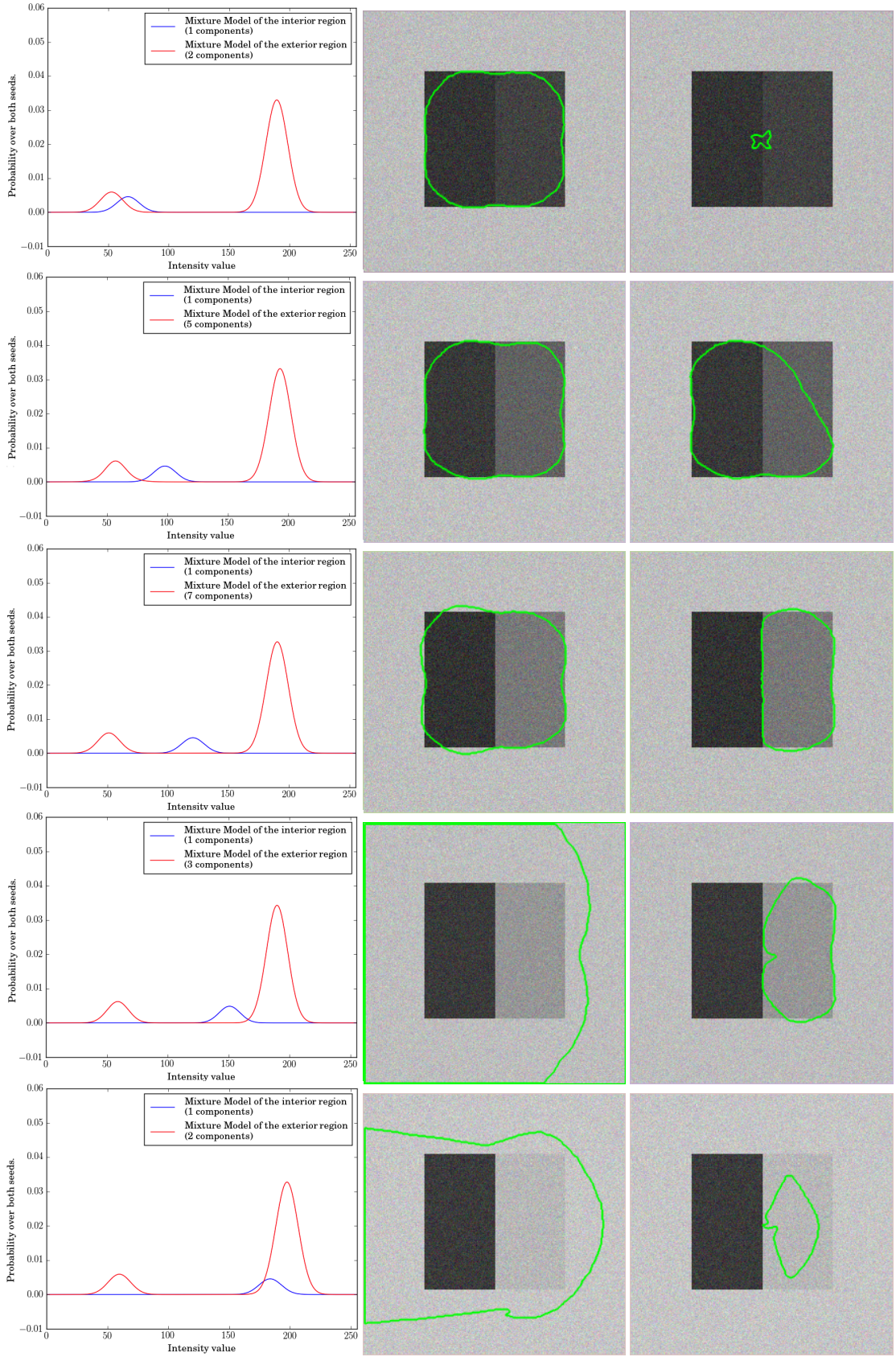

Figure 5.7: Experimental validation of the capacity of the Mixture Gaussian energy (left column) to capture a region with values between the background's values. Parameters: 12 control points, ratio=1.05, sigma=1.25, $\epsilon = e^{-200}$)

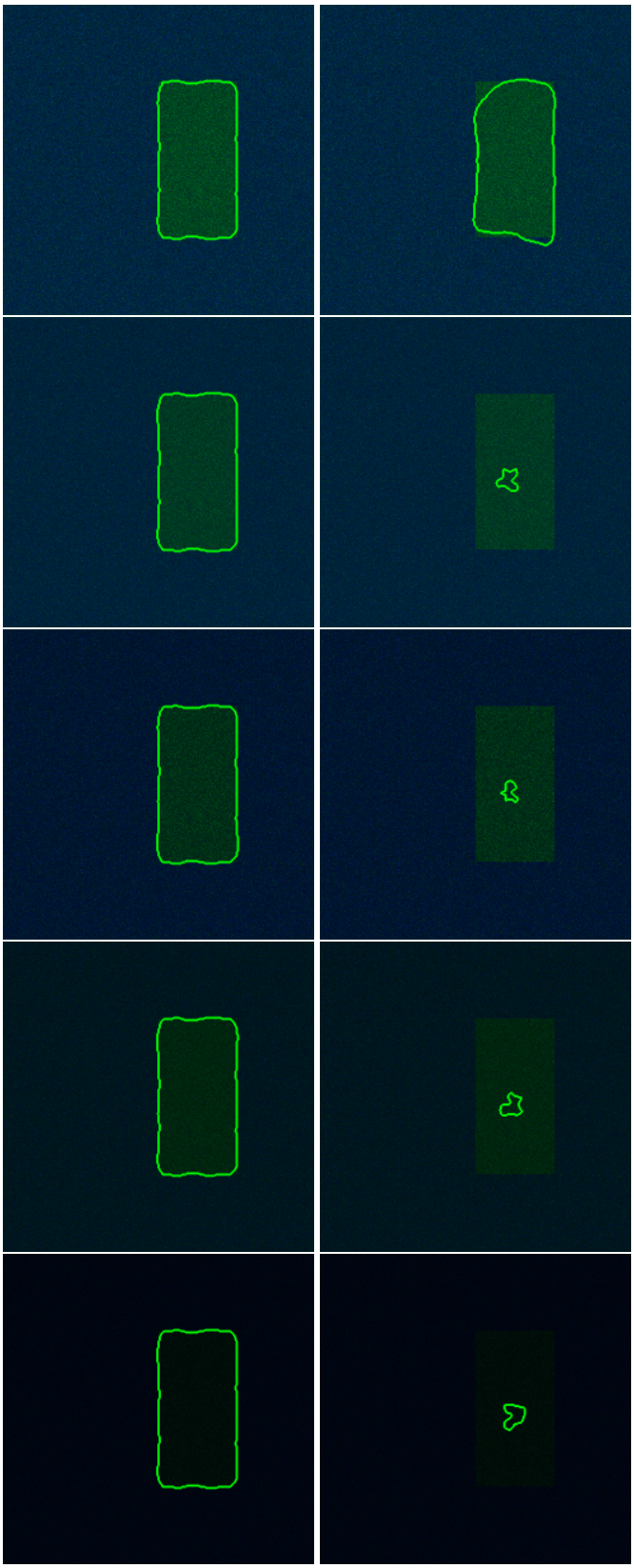

Figure 5.9: Segmentation results of the mean Hue energy (left column) and of the Multivariate Gaussian Energy (right colum) with different illuminations. Parameters: 12 control points, ratio=1.05, sigma=0.25, $\epsilon = e^{-200}$)

Figure 5.10: Segmentation result with Multivariate Mixture Gaussian Energy

Figure 5.11: Segmentation result with Multivariate Mixture Gaussian Energy.

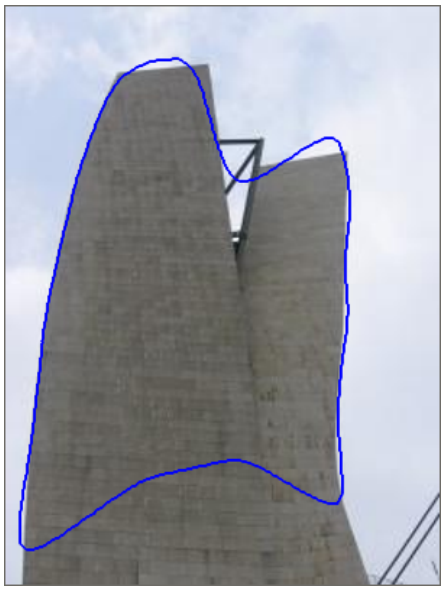

Figure 5.12: Segmentation result with Multivariate Mixture Gaussian Energy

Figure 5.13: Segmentation result with Multivariate Mixture Gaussian Energy.

Figure 5.14: Positively segmented images using the Multivariate Mixture Gaussian Energy.

6 Shape categorization and application to morphing

The original use of the cage in the algorithm is to parametrize the image and to deform the contour during the evolution to minimize the energy. We want to emphasize the advantages of having the cage parametrization and its properties. This chapter provides some first results of the future research for Cage Active Contours.

6.1 Properties of the cage parametrization

As a motivation for this line of research, we propose applications of the cage parametrization in image morphing and warping and as shape descriptors.

To formalize these applications we need a way to compare similar contour shapes. If we fix an initial contour and a cage configuration, we will get that for every new cage, a deformation of the initial contour by equation (3.5) defines a deformed contour shape. Through observation, we see that similar cages, provide similar contour images in certain initial conditions. We want to find a criteria which allows us to link an ordered configuration of points (i.e. a cage) with contour shapes so that we may use existing tools from polygon similarity[11] and from point configuration similarity, like Procrustes analysis distance, to determine shape similarity between different contours.

More formally, we present the following definitions which will lead up to the proposition and its proof.

Definition 5 (Contour Family)**.** *Given an initial contour C and an initial cage $V = (v_1, v_2, \ldots, v_N)$, the family of contours $\mathcal{F}_C^V$ is the set of all the possible contours that can be produced with all cages of N points by a deformation through* (3.5) *and it is expressed as:*

$$\mathcal{F}_C^V = \{C^W | W \in (\mathbb{R}^2)^N\}$$

where for any cage $W \in (\mathbb{R}^2)^N$

$$C^W = \{q \in \mathbb{R}^2 | q = \sum_i^N \varphi_i^V(p) w_i, \ \forall p \in C, W = (w_1, w_2, \ldots, w_N)\}$$

and $\varphi_i^V(p)$ are the mean value coordinates of p with respect to cage V.

Definition 6 (Similarity)**.** *We define a similarity on the plane as an affine transformation $f : \mathbb{R}^2 \to \mathbb{R}^2$ composed of rotations, translations and uniform changes in scale.*

Definition 7 (Contour similarity)**.** *Two contours are similar if there exists a similarity which maps one to the other.*

Definition 8 (Cage similarity)**.** *Two cages $U = (u_1, u_2, \ldots, u_N)$ and $W = (w_1, w_2, \ldots, w_N)$ are similar if there exists a similarity function such that $f(u_i) = w_i$ for each $i \in \{1, 2, \ldots, N\}$.*

Definition 9 (Shifted cage)**.** *A shifted cage of another cage $W = (w_1, w_2, \ldots, w_N)$ is a permutation conserving the order of W. There are N shifts (as many as number of points).*

$$W = W^0 = (w_1, w_2, \ldots, w_N)$$

$$W^1 = (w_2, w_3, \ldots, w_N, w_1)$$

$$\ldots$$

$$W^k = (w_{k+1}, w_{k+2}, \ldots, w_k)$$

$$\ldots$$

$$W^{N-1} = (w_N, w_1, \ldots, w_{N-2}, w_{N-1})$$

In definition 5 we have defined the contour Family of an initial configuration of a contour C and a cage V. However, there are certain properties that we would like to impose on this Family. Namely, we are interested in those Families where similar cages or similar shifted cages define the same contour.

First we need a definition:

Definition 10. *A **regular initial cage-contour configuration with ratio r** is a set (V, C, r) consisting of an initial cage $V = (v_1, v_2, \ldots, v_N)$ that defines an N-sided regular polygon and an initial contour C that is a circumference concentric to the polygon such that the ratio of the radius of C and the radius of the polygon is r:1. For simplicity we say the ratio is r.*

Now that we have these concepts formally and well defined, we are able to prove the desired property of the family:

Proposition 1. *Given a regular initial cage-contour configuration (V,C,r), then for every contour C^W and C^U in the contour family F_V^C, C^W and C^U are similar if*

1. *W and U are similar cages,*
   or
2. *U is a shifted cage of a similar cage of W.*

Proof. The first point is trivial. We want to see if there exists a similarity function g that sends C^W to C^U. So for every point of $q^W \in C^W$ there has to exist a point $q^U \in C^U$ such that $g(q^U) = q^W$. By construction of C^W and C^U, we know that there exists a point $p \in C$ such that

$$q^W = \sum_{i=1}^{N} \varphi_i^V(p) w_i$$

and a point $p' \in C$ such that

$$q^U = \sum_{i=1}^{N} \varphi_i^V(p') u_i$$

Since we know that cages W and U are similar, we have that, by definition 8, there exists a similarity f that maps cage U to W (i.e $w_i = f(u_i)$ for all $i \in \{1, 2, \ldots, N\}$). It turns out that with $g = f$ and $p^U = p^W$ define the similarity between contours:

$$q^W = \sum_{i=1}^{N} \varphi_i^V(p) w_i = \sum_{i=1}^{N} \varphi_i^V(p) f(u_i) = f\Big(\sum_{i=1}^{N} \varphi_i^V(p) u_i\Big) = f(q^U)$$

Therefore we have that the same similarity that maps W to U, sends their contours to each other rendering them similar.

To prove the second implication, a more elaborate solution is required. We only need to prove this in the case of if U is the shifted cage of W since if we have that, any similar cage would only imply a similarity function. To see that a cage and its shifted cage produces a similar curves, let us take two cages $W^0 = (w_1, w_2, \ldots, w_3)$ and one of its shifted (we take the shift k=1 for simplicity) $W^1 = (w_1^1, w_2^1, \ldots, w_N^1) = (w_2, w_3, \ldots, w_N, w_1)$.

If we see that their images[1] of C, respectively C^{W^0} and C^{W^1} are congruent, that is $C^{W^0} = C^{W^1}$, then they would be similar because the identity function would be the similarity between them.

To see this we have to see if every point q in C^{W^0} is in C^{W^1}. We have that every point in C^{W^0} can be expressed as

$$q = \sum_{j=1}^{N} \varphi_j^V(p) w_j$$

where $p \in C$ is in the initial contour. If we can find a point p^1 in C such that

$$q = \sum_{j=1}^{N} \varphi_j^V(p^1) w_j^1$$

it would do.

The mean value coordinates of a point p with respect to control point v_i are calculated using the angles α_1, α_2 with its neighboring control points v_{i-1} and v_{i+1} respectively. In figure 6.1, we have an example with the circumference contour C and the cage $V = (v_1, v_2, .., v_N)$ (N=6 in the image). Point p has the mean value coordinates $\varphi^V(p) = (\lambda_1, \lambda_2, \ldots, \lambda_N)$. If we apply a rotation R^1 of $\alpha_{R^1} = -\frac{2\pi}{N}$ radians and center p_c. We have that $R^1(v_i) = v_{i+1}$ and the rotated point $p^1 = R^1(p)$ would still be on the contour C. Furthermore, It would maintain the distance to the rotated control point $R^1(v_i) = v_{i+1}$, as well as the angles to their rotated points, because of the property of angle invariance through similarities.

Therefore we can say that for every point, p, there exists a point $p^1 = R^1(p)$ such that, the mean value coordinates are the same but shifted: this can be done for any $R^k(p) = -\frac{2\pi}{N} * k$ for $k \in 1, 2, \ldots, N$;

$$\varphi^V(R^1(p)) = (\lambda_2, \lambda_3, ..., \lambda_N, \lambda_1)$$
$$\varphi^V(R^2(p)) = (\lambda_3, \lambda_2, ..., \lambda_{N-1}, \lambda_2)$$
$$\ldots$$

[1] In this context, image refers to the target set of a function.

$$\varphi^V(R^k(p)) = (\lambda_{k+1}, \lambda_{k+2}, ..., \lambda_{N-k-1}, \lambda_k)$$

$$\dots$$

$$\varphi^V(R^{N-1}(p)) = (\lambda_N, \lambda_1, ..., \lambda_{N-2}, \lambda_{N-1})$$

So, now that we have these points, we know that given any point $q \in C^{W^0}$, there does exist a point $p'^1 \in C$ so that $q = \sum_j^N \varphi_j^V(p) w_j^2$ and it is in particular $p' = R^1(p)$. Since we have the following:

$$q = \sum_j^N \varphi_j^V(p) w_j^1 = w_1\lambda_1 + w_2\lambda_2 + \dots + w_N\lambda_N = w_2\lambda_2 + w_3\lambda_3 + \dots + w_N\lambda_N + w_1\lambda_1 = \sum_j^N \varphi_j^V(p') w_j^1$$

Since we can generalize for any shift $k \in \{1, 2, \dots, N\}$ with rotation R^k, the proposition is proven. □

Furthermore, it would be interesting to prove the opposite implication thereby creating a class of equivalence between shapes in the contour family $\mathcal{F}_C^V$ defined by the cage V and initial contour C from the initial configuration. This will be left for future work.

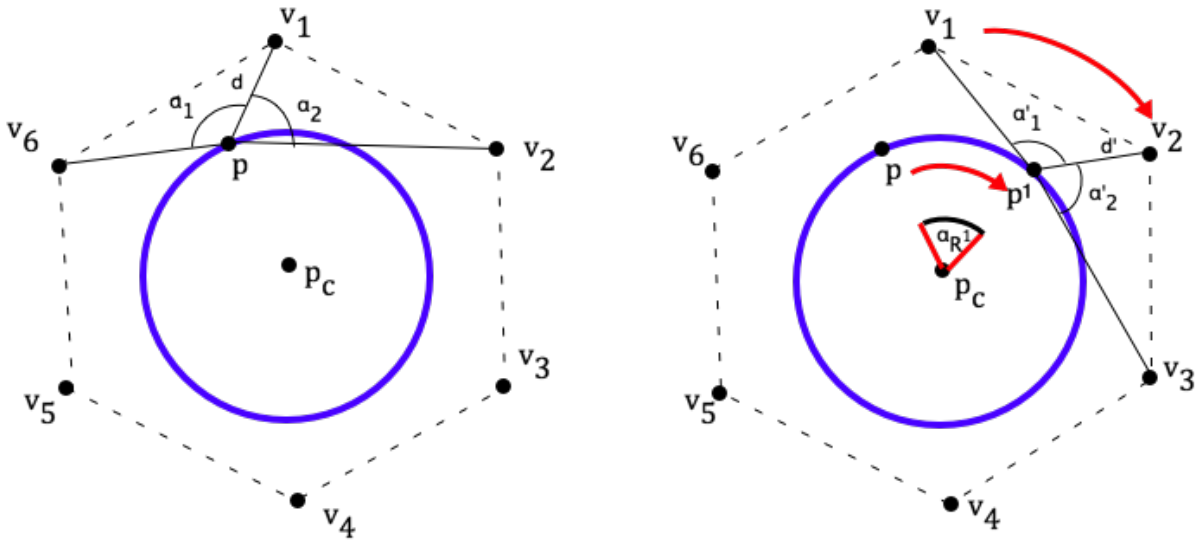

Figure 6.1: Illustration of the existence of a point p^1 needed to prove the second implication in proposition 1

6.2 Shape description

One of the challenges seen especially in medical imaging is that it is often hard to find relevant points in a region that might help to determine structure or orientation of an object that apparently has none. These points are commonly called *landmarks* and are used to build the shape models of an object. It is often the case in medical imaging that these points are unseen, latent or that they are simply characterized by their shape.

Shape description can also be applied to image retrieval. This application supposes huge databases of images which from we want to retrieve a certain subset or a specific image given specific characteristics. For example in medical imaging, if we want to withdraw all the cases of patients with a similarly shaped elements, such as *caudates*, *putamen*, or *nucleous*, a fast search would be needed to compare with all patient files. This search would require two things: invariance in translation, rotation and scale, and that each element in this dataset could be indexed so that fast and effective retrieval and comparison may be applied.

The literature in shape comparison is a rich and vast field of research [9, 1]. One of the best methods of shape description are Discrete Fourier Transforms (DFT). These provide a description of the curvature of a shape with respect to a variable t which indicates the point in the curve it is in. These are invariant to translation and uniform change in scale, but the starting point of the shape is critical to know whether two shapes [9] are similar through a rotation or not.

Another interesting method is the Curvature Scale-Space (CSS) shape descriptor. This descriptor provides a representation of a contour which represents the time of inflection or union of pairs of points of the shape as it is progressively smoothed [1]. This descriptor also presents the same problem in rotation, where a shift must be applied to find the right starting point. Figure 6.2 shows the example of how as the contour of a shark is smoothed, the less relevant points are joined.

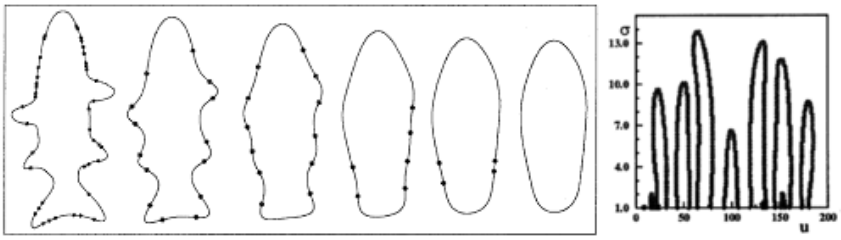

Figure 6.2: CSS smoothing process of a shape and decreasing number of the points with curvature change (image from [1]).

Both of these methods, which are the most used in this field [4, 75] provide very good solutions to indexing, description and even rotation [75]. Similarly, Through a segmentation with CAC, once we fix a regular initial cage-contour configuration with N points (defined in 10), we have an approximated shape of the curve which can be described exactly with the N points in the segmented cage, making it a good index as well as a useful descriptor. Also, as we have proved in proposition 1, that similar cages with similar shifted cages represent the same contour. This is useful for fast comparison since we only require at most N comparisons to see the invariance through rotation.

However, CAC offers an advantage over these methods in being a segmentation method, while the previous shape descriptors require a segmentation step which provides a simple connect region. We obtain both in a single process and thanks to the cage parametrization, the segmentation can be corrected by a user by moving the points in the cage.

6.3 Image Morphing and Warping

Image morphing is the interpolation between two *images* while warping is the deformation of the shape of an image. We are interested in morphing objects into each other.

If we have a regular cage-contour configuration, (C, V, r), and we segment two objects O_1 and O_2 in images I^1 and I^2 respectively, then

1. By proposition 1, if the resulting cages V^1 and V^2 are similar or similar to a shifted cage, the contours are similar.

2. By property **C.2**, if there exists a similarity f between cages, then by that similarity the mean value coordinates of O_1 with respect to V^1 are equal to the mean value coordinates of $f(O_2)$ with respect to V^2.

3. In the proof or proposition 1, we show that we can always find a shift of a shifted cage so that we may find the similarity f.

By these three points, we have that if the segmentation of O_1 and O_2 gives two cages V_1 and V_2 respectively that are similar or shifted of similar cages, the same similarity sends O_1 to O_2.

If we want to morph two objects $O_1 \in I^1$ and $O_2 \in I^2$ which respectively have segmentation V^1 and V^2, then we have that we can define an intermediate cage:

$$V^w = V^1 * w + V^2 * (1 - w) \tag{6.1}$$

where $w \in [0, 1]$ such that if two cages are similar, they are also similar to their intermediate. For any two cages in general this interpolation is depicted in figure 6.3.

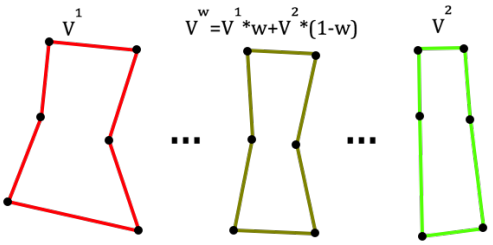

Figure 6.3: Intermediate cage in morphing

In the new interpolated image, we now want to find pixel by pixel its corresponding values in each image, and apply a weighted mean to obtain the interpolated value. since they are similar, we have

$$I^w(p^w) = w * I^1\left(\sum_{i=1}^{N} \varphi_i(p^w)v_i^1\right) + (1-w) * I^2\left(\sum_{i=1}^{N} \varphi_i(p^w)v_i^2\right) \tag{6.2}$$

Now the problem emerges in practice since it is practically impossible for two cages to be similar after a segmentation however they can be similar to a slight permutation of the cage. Thanks to the smooth properties of the deformation this allows for decent morphing through interpolation of the cages. Figures 6.4 and 6.5 are examples we created by interpolation of cages. The former is done automatically by finding the shift of the cages that best corresponds to a similarity using a turning function we implemented, while in the fruit images, we assigned a correspondence in the cages that are not similar to show the smoothness this deformation provides regardless.

Figure 6.4: Morphing a family car to a sports car automatically through mean value coordinates from a segmentation with CAC (Initial image from `http://www.wellclean.com/wp-content/themes/artgallery_3.0/images/car1.png`, final image from `http://www.wellclean.com/wp-content/themes/artgallery_3.0/images/car3.png`)

Figure 6.5: Morphing from an apple to a pear with a CAC segmentation (Initial and final images from [69])

7 Conclusions and Future Work

In this thesis we have made various contributions to the object segmentation method Cage Active Contours (CAC) originally introduced in [30]. Our contributions include 1) the introduction of two different energy functions on different color spaces which have greatly enhanced the potential of an otherwise limited method. 2) the experimental validation of our improvements with the previous energies in CAC as well as with three different related methods. 3) the formalization, in mathematical terms, of some the implications and uses of the resulting segmentation *cage* (a component of CAC which is used to parametrize the contour). 4) a highlight of the possible applications of this cage to image morphing, warping as well as in shape description. 5) the creation of a public implementation in Python (with some wrapped functions in C) of cage active contours.

7.1 Conclusions

One of the main contributions in this work is the creation of two different energy functions. These are enhanced versions of simpler energies proposed so far in CAC which have also been extended to two different color spaces. The first one is the Multivariate Mixture Gaussian energy which is an extension to the RGB color space of the Gaussian energy defined in [30] with two more improvements: the ability to capture multiple value components in each region by using Mixture Gaussian density function, and the incorporation of an initial seed which will provide the energy with prior information about the foreground and background's distributions.

The second energy is the Mean Hue Energy, which is the analogous of the Mean Energy in [30] but on the Hue component of the HSI/HSV color spaces. Because of the Cyclic nature of the Hue component, we have had to turn to cyclic spaces to use concepts such as distance, directed distance as well as a way of finding the gradient of a of the image with respect to a control point's value.

Through quantitative and qualitative experimentation on three different datasets (section 5), we have observed that the contribution with the most impact are the extension of the Gaussian energy to the RGB color space followed by the ability of Mixture Gaussian energies to describe a region with more than one component thus obtaining a more sophisticated representation of a region, as opposed to the polarization of pixel values in previous energies. We have also seen how the incorporation of a seed in each region does not only simplify the process but it also increases its computational speed. In the case of color images, we have seen how the Mean Hue energy, despite its simplicity, can occasionally outperform the Multivariate Mixture Gaussian energy thanks to its focus on the Hue component of the image, which reflects an intrinsic property of objects, invariant to illumination.

Furthermore, we have mathematically formalized the concepts of *cage*, *contour*, *family of*

contours and others to be able to prove that two contours are similar if their cages are similar given some intial conditions. This theoretical proof, along with the properties of mean value coordinates, used widely in computer graphic applications, have allowed us to define the conditions which allow for automatic morphing and warping between similar objects, as well provide some initial intuitions for the possibility of shape description.

Our last contribution is a public implementation of Cage Active Contours in Python with some wrappers in C. The code contains different Energy functions we have presented including the ones presented in [30], as well as some tools for automatic morphing and warping. The code can be found in `https://github.com/Jeronics/cac-segmenter/`.

7.2 Future Work

In this work, we have also observed and pointed out the limitations that hinder the performance of the segmentation using Cage Active Contours. We have seen that CAC are not designed for high precision segmentation of arbitrary images, but rather, they provide a smooth general contour of the image which can be used for other purposes and applications. The limitations of this method can be divided into two categories: those that are dependent on energy functions and those that are dependent on the cage.

With these considerations in mind, we have proposed a set of solutions which could be evaluated and considered for future work.

The first point to address is the cage. We have seen that the restrictions of the first stage on the vertex movements are too strong and prevent CAC from adopting complex figures. Also, they prevent the cage from rotate to better adapt to a shape. Possible solution include the weighted mean between the real direction and the projected direction or the exploration of other internal energies that could be applied to the cage for stability.

In terms of energy functions, the most challenging task ahead, by far, is to improve the Mean Hue energy which, despite its theoretically good properties in illumination invariance, has performed poorly on the quantitative experiments on real images. However we are confident that through some improvements we could be able to rise its performance to a competent level. The first improvement we propose is the adaptation of the Gaussian Energy to this space and ultimately to the Mixture Gaussian model. This extension requires the use of analogous density functions in the cyclic domain which are known as Wrapped Density Functions. The second improvement would come from the extension of an energy to the whole HSI or HSV space. As we have concluded in section 4.2.3 this requires tools from directional statistics and in particular, the study of cylinder spaces which these color spaces define.

As far as the implementation goes, it would be interesting to allow for a more intuitive and user-friendly interactive interface. Some of the features this could include:

1. User interaction with the cage so that control points would be able to be dragged to provide a better initialization, and to correct a segmentation at a certain point.

2. A Morphing and Warping interface that would allow for fine-tuning of morphed objects or reassigning of correspondence in points as we have done in the morphing between the pear and apple in figure 6.5.

An interesting point we have begun to raise in this thesis is the formalization in mathematical terms of the different components in Cage Active Contours to study their properties and limitations in different applications once a segmentation has been achieved. In the process of bringing

solutions, more questions are brought forward to discuss and tackle in future work. Namely: Given a regular cage-configuration condition:

- What is the complexity of contours that can be expressed in a contour Family?
- Which characteristics must a cage fulfill in order to avoid its corresponding contour to self intersect?
- Is there a bijective map between the equivalence classes of cages and the equivalence classes of contours in a contour Family?

References

[1] Abbasi, S., Mokhtarian, F., and Kittler, J. (1999). Curvature scale space image in shape similarity retrieval. *Multimedia Syst.*, 7(6):467–476.

[2] Allili, M. S. and Ziou, D. (2005). An automatic segmentation of color images by using a combination of mixture modelling and adaptive region information: a level set approach. In *ICIP (1)*, pages 305–308. IEEE.

[3] Alpert, S., Galun, M., Basri, R., and Brandt", A. ("2007"). "image segmentation by probabilistic bottom-up aggregation and cue integration.". In *"Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition"*.

[4] Amanatiadis, A., Kaburlasos, V., Gasteratos, A., and Papadakis, S. (2011). Evaluation of shape descriptors for shape-based image retrieval. *Image Processing, IET*, 5(5):493–499.

[5] Anderson-cook, C. M. and Otieno, B. S. (2014). *Cylindrical Data*. John Wiley & Sons, Ltd.

[6] Arbelaez, P., Maire, M., Fowlkes, C., and Malik, J. (2011). Contour detection and hierarchical image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(5):898–916.

[7] Babalola, K., Patenaude, B., Aljabar, P., Schnabel, J., Kennedy, D., Crum, W., Smith, S., Cootes, T., Jenkinson, M., and Rueckert, D. (2008). Comparison and evaluation of segmentation techniques for subcortical structures in brain mri. In Metaxas, D., Axel, L., Fichtinger, G., and Székely, G., editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2008*, volume 5241 of *Lecture Notes in Computer Science*, pages 409–416. Springer Berlin Heidelberg.

[8] Barbosa, D., Dietenbeck, T., Schaerer, J., D'hooge, J., Friboulet, D., and Bernard, O. (2012). B-spline explicit active surfaces: an efficient framework for real-time 3D region-based segmentation. *IEEE Transactions on Image Processing*, 21(1):241–251.

[9] Bartolini, I., Ciaccia, P., and Patella, M. (2005). Warp: Accurate retrieval of shapes using phase of fourier descriptors and time warping distance. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27(1):142–147.

[10] Bernard, O., Friboulet, D., Thévenaz, P., and Unser, M. (2009). Variational B-spline level-set: A linear filtering approach for fast deformable model evolution. *IEEE Transactions on Image Processing*, 18(6):1179–1191.

[11] Bykat, A. (1979). On polygon similarity. 9(1):23–25.

[12] Carreira-Perpinan, M. (2000). Mode-finding for mixtures of gaussian distributions. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(11):1318–1323.

[13] Caselles, V., Catte, F., Coll, T., and Dibos, F. (1993). A geometric model for active contours. *Numerische Mathematik*, pages 694–6999.

[14] Caselles, V., Kimmel, R., and Sapiro, G. (1997). Geodesic active contours. *International Journal of Computer Vision*, 22:61–79.

[15] Cha, S.-H. (2007). Comprehensive survey on distance/similarity measures between probability density functions.

[16] Chan, T. and Vese, L. (2001). Active contours without edges. *IEEE Transactions on Image Processing*, 10(2):266–277.

[17] Cheng, H. D., Jiang, X. H., Sun, Y., and Wang, J. L. (2001). Color image segmentation: Advances and prospects. *Pattern Recognition*, 34:2259–2281.

[18] Cheng, H. D., Jiang, X. H., and Wang, J. (2002). Color image segmentation based on homogram thresholding and region merging. *Pattern Recognition*, 35:373–393.

[19] Chien, C. L. and Tseng, D. C. (2011). Color image enhancement with exact hsi color model.

[20] Cohen, L. (1991). On active contour models and balloons. *Image Understanding*, 53:211–218.

[21] ColorCodeHex (1999). "color hex color codes".

[22] Coquillart, S. (1990). Extended free-form deformation: a sculpturing tool for 3d geometric modeling. *SIGGRAPH*, 24(4):187–196.

[23] Crum, W., Camara, O., and Hill, D. (2006). Generalized overlap measures for evaluation and validation in medical image analysis. *Medical Imaging, IEEE Transactions on*, 25(11):1451–1461.

[24] Delingette, H. and Montagnat, J. (2000). Topology and shape constraints in parametric active contours. Technical report, Institut National de Recherche en Informatique et en Automatique.

[25] Dietenbeck, T., Alessandrini, M., Friboulet, D., and Bernard, O. (2010). Creaseg: A free software for the evaluation of image segmentation algorithms based on level-set. In *Image Processing (ICIP), 2010 17th IEEE International Conference on*, pages 665–668.

[26] Du, L., Ren, L., Carin, L., and Dunson, D. B. (2009). A bayesian model for simultaneous image clustering, annotation and object segmentation. In Bengio, Y., Schuurmans, D., Lafferty, J., Williams, C., and Culotta, A., editors, *Advances in Neural Information Processing Systems 22*, pages 486–494. Curran Associates, Inc.

[27] Floater, M. S. (2003). Mean value coordinates. *Computer Aided Geometric Design*, 20(1):19–27.

[28] Floater, M. S. and Gotsman, C. (1999). How to morph tilings injectively. *Journal of Computational and Applied Mathematics*, 101(1–2):117 – 129.

[29] Galun, M., Sharon, E., Basri, R., and Brandt, A. (2003). Texture segmentation by multiscale aggregation of filter responses and shape elements. In *IN ICCV*, pages 716–723.

[30] Garrido, L., Guerrieri, M., and Igual, L. (2015). Image segmentation with cage active contours. *IEEE Transactions on Image Processing*.

[31] Gladilin, E. (2002). *Biomechanical Modeling of Soft Tissue and Facial Expressions for Craniofacial Surgery Planning*. PhD thesis, Free University of Berlin, Germany.

[32] Gouraud, H. (1971). Continuous shading of curved surfaces. *IEEE Transactions on Computers*, C-20(6):623 – 629.

[33] Han, X., Xu, C., and Prince, J. L. (2003). A topology preserving level set method for geometric deformable models. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, 25(6):755–768.

[34] Hormann, K. and Floater, M. (2006). Mean value coordinates for arbitrary planar polygons. *ACM Transactions on Graphics*, 25(4):1424–1441.

[35] Huntsherger, T., Jacobs, C., and Cannon, R. (1985). Iterative fuzzy image segmentation. *Pattern Recognition*, 18(2):131 – 138.

[36] Huttenlocher, D. P., Klanderman, G. A., and Rucklidge, W. A. (1993). Comparing images using the hausdorff distance. *IEEE Trans. Pattern Anal. Mach. Intell.*, 15(9):850–863.

[37] Jacob, M., Blu, T., and Unser, M. (2004). Efficient energies and algorithms for parametric snakes. *IEEE Transactions on Image Processing*, 13(9):1231–1244.

[38] Jiang, C., Zhang, X., Huang, W., and Meinel, C. (2004). Segmentation and quantification of brain tumor. In *Virtual Environments, Human-Computer Interfaces and Measurement Systems, 2004. (VECIMS). 2004 IEEE Symposium on*, pages 61–66.

[39] Joschi, P., Meyer, M., DeRose, T., Green, B., and Sanocki, T. (2007). Harmonic coordinates for character articulation. In *SIGGRAPH*.

[40] Jun, X., Tsui, H., and Deshen, X. (2002). Multiple objects segmentation based on maximum-likelihood estimation and optimum entropy-distribution (mle-oed). In *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, volume 1, pages 707–710 vol.1.

[41] Kass, M., Witkin, A., and Terzopoulos, D. (1988). Snakes: Active contour models. *International Journal of Computer Vision*.

[42] Kobayashi, K. and Ootsubo, K. (2003). t-ffd: free-form deformation by using triangular mesh. In *ACM Symp. on Solid Modeling and Applications*, pages 226–234.

[43] Lankton, S. and Tannenbaum, A. (2008). Localizing region-based active contours. *IEEE Transactions on Image Processing*, 17(11):2029–2039.

[44] Li, C., Kao, C., Gore, J. C., and Ding, Z. (2008). Minimization of region-scalable fitting energy for image segmentation. *IEEE Transactions on Image Processing*, 17(10):1940–1949.

[45] Li, Z., Wu, X.-M., and Chang, S.-F. (2012). Segmentation using superpixels: A bipartite graph partitioning approach. In *CVPR*, pages 789–796. IEEE Computer Society.

[46] Lipman, Y., Levin, D., and Cohen-Or, D. (2008). Green coordinates. In *SIGGRAPH*.

[47] Lucchese, L. and Mitra, S. K. (2001). Color image segmentation: A state-of-the-art survey.

[48] Mardia, K. V. and Jupp, P. E. (2008). *Basic Concepts and Models*, pages 25–56. John Wiley & Sons, Inc.

[49] Martin, D., Fowlkes, C., Tal, D., and Malik, J. (2001). A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proc. 8th Int'l Conf. Computer Vision*, volume 2, pages 416–423.

[50] Martin, D. R., Fowlkes, C. C., and Malik, J. (2004). Learning to detect natural image boundaries using local brightness, color, and texture cues. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(5):530–549.

[51] Michailovich, O., Rathi, Y., and Tannenbaum, A. (2007). Image segmentation using active contours driven by the Bhattacharyya gradient flow. *IEEE Trans. on Image Processing*, 16(11):2787–2801.

[52] Mille, J. and Cohen, L. (2009). A local normal-based region term for active contours. In *Int. Conf. on Energy Minimization Methods in Computer Vision and Pattern Recognition*, pages 168–181. Springer.

[53] Moghaddam, M. J. and Soltanian-Zadeh, H. (2011). Medical image segmentation using artificial neural networks, artificial neural networks. In Suzuki, P. K., editor, *Artificial Neural Networks - Methodological Advances and Biomedical Applications*. InTech.

[54] Nealen, A., Müller, M., Keiser, R., Boxermann, E., and Carlson, M. (2005). Physically based deformable models in computer graphics. In *Eurographics 2005. STAR - State of the Art Reports*, pages 71–94.

[55] Ohba, K., Sato, Y., and Ikeuchi, K. (2000). Appearance-based visual learning and object recognition with illumination invariance. *Machine Vision and Applications*, 12(4):189–196.

[56] Osher, S. and Sethian, J. A. (1988). Fronts propagating with curvature-dependent speed: Algorithms based on hamilton-jacobi formulations. *Journal of Computational Physics*, 79(1):12 – 49.

[57] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12:2825–2830.

[58] Pham, D. L., Xu, C., and Prince, J. L. (2000). Current methods in medical image segmentation1. *Annual Review of Biomedical Engineering*, 2(1):315–337.

[59] Phong, B. T. (1975). Illumination for computer generated pictures. *Commun. ACM*, 18(6):311–317.

[60] Precioso, F., Barlaud, M., Blu, T., and Unser, M. (2005). Robust real-time segmentation of images and videos using a smoothing-spline snake-based algorithm. *IEEE Transaction on Image Processing*, 14(7):910–924.

[61] Price, B. L., Morse, B. S., and Cohen, S. (2010). Geodesic graph cut for interactive image segmentation. In *CVPR*, pages 3161–3168. IEEE Computer Society.

[62] Rother, C., Kolmogorov, V., and Blake, A. (2004). Grabcut -interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics (SIGGRAPH)*.

[63] Rousson, M. and Deriche, R. (2002). A variational framework for active and adaptative segmentation of vector valued images. In *IEEE Proceedings of the Workshop on Motion and Video Computing*, pages 56–61.

[64] Sezgin, M. and Sankur, B. (2004). Survey over image thresholding techniques and quantitative performance evaluation. *J. Electronic Imaging*, 13(1):146–168.

[65] Shi, Y. and Karl, W. (2008a). A real-time algorithm for the approximation of level-set-based curve evolution. *Image Processing, IEEE Transactions on*, 17(5):645–656.

[66] Shi, Y. and Karl, W. C. (2008b). A real-time algorithm for the approximation of level-set-based curve evolution. *IEEE Transactions on Image Processing*, 17(5):645–656.

[67] Titterington, D., Smith, A., and Makov, U. (1985). *Statistical Analysis of Finite Mixture Distributions*. Wiley, New York.

[68] Vergés Llahí, J. (2005). *State-of-the-Art Survey on Color Segmentation Methods*.

[69] Škrjanec Marko (2013). Automatic fruit recognition using computer vision. (Mentor: Matej Kristan), Fakulteta za računalništvo in informatiko, Univerza v Ljubljani.

[70] Wit, E., Heuvel, E. v. d., and Romeijn, J.-W. (2012). All models are wrong...: an introduction to model uncertainty. *Statistica Neerlandica*, 66(3):217–236.

[71] Xu, L. and Jordan, M. I. (1995). On convergence properties of the em algorithm for gaussian mixtures. *Neural Computation*, 8:129–151.

[72] Xue, Q., Igual, L., Berenguel, A., Guerrieri, M., and Garrido, L. (2014). Active contour segmentation with affine coordinate-based parametrization. In *Int. Conference on Computer Vision Theory and Applications*.

[73] Yu, S. X. (2009). Angular embedding: From jarring intensity differences to perceived luminance. In *CVPR*, pages 2302–2309. IEEE Computer Society.

[74] Zhang, C., Wang, P., and Fellow, I. (2000). A new method of color image segmentation based on intensity and hue clustering. In *in International Conference on Pattern Recognition (ICPR). 2000*, pages 613–616.

[75] Zhang, D. and Lu, G. (2003). A comparative study of curvature scale space and fourier descriptors for shape-based image retrieval. *Journal of Visual Communication and Image Representation*, 14(1):39 – 57.