Smart Cage Active Contours and their application to brain image segmentation

Israel Saeta Pérez
Advisors: Lluís Garrido and Laura Igual Muñoz

2013-09-09

Abstract

In this work we present a new segmentation method named Smart Cage Active Contours (SCAC) that combines a parametrized active contour framework named Cage Active Contours (CAC), based on affine transformations, with Active Shape Models (ASM). Our method effectively restricts the shapes the evolving contours can take without the need of the training images to be manually landmarked. We apply our method to segment the caudate nuclei subcortical structure of a set of 40 subjects in magnetic resonance brain images, with promising results.

Acknowledgements

I'd like to thank my supervisors for all their help and guidance on the elaboration of this work. I'd also like to thank my family and friends, and especially my little brother and my girlfriend, for their patience and understanding during this long busy summer.

Contents

1 Introduction

Image segmentation [1] is classically defined as the process of partitioning an image into multiple non-overlapping groups of pixels, known as segments, superpixels or simply segmentation regions. These groups of pixels are usually connected and separated by linear or surface boundaries. Ideally, segmentation results correspond to dividing the image into regions following a certain reasoning, like anatomical regions in medical imaging.

There are multiple methods to perform image segmentation, each of them exploiting a different feature of the image and using a different concept of how to define and locate the boundaries. As often occurs, there is no single method that can be considered superior to others in general. The fitness of a segmentation method depends largely on the type of image, the characteristics of the areas to be segmented and also the particular conditions of the images to work with, like the level of noise and contrast, if they are black and white or full-colored, their size, etc. Actually, many segmentation methods are especially designed or tuned to tackle a very specific task, as is the case of atlas-guided segmentation.

Segmentation of medical images is a very active research topic due to the importance of its applications in medicine. While powerful image acquisition techniques like MRI, CT or PET are already available, the information present on these images is usually so complex that the ability to automatically separate a certain region of interest, like an organ, from the rest of the image, can be of great help to turn plain images into medical (and potentially life-saving) knowledge. Otherwise, doctors would have to perform the segmentation of every image manually, a time consuming and error-prone process subject to inter-observer variability.

Medical image segmentation is in general a very complex and challenging problem, mainly due to the great variability of the properties of the structures to be segmented and the particular difficulties of some cases, like the lack of contrast or other clear visual boundaries, the presence of noise or blur due to motion, partial volume effects, etc. Even focusing on a specific organ and image modality (MRI, CT, PET, etc.), the variability can affect very negatively the overall performance of a segmentation algorithm: while it might be quite good at segmenting the general and close-to-the-mean case, it can fail miserably for another partially deformed or distorted image. Since these results might need to be used for applying a therapy, like surgery or radiotherapy, it is important for the segmentation techniques used to be robust and accurate, what makes applied medical image segmentation even more hard and challenging. For a general review on medical image segmentation, see [1, 2, 3, 4].

An increasingly popular approach in medical image segmentation involves incorporating *a priori* knowledge about the structures to be segmented into a deformable model [5] in the form of initial conditions, constraints on the shape parameters or into the model fitting procedure. This is the case of Active Shape Models [6, 7], that construct a model of shape of the selected structures constraining the possible shapes the model can generate. However, one of main drawbacks of Active Shape Models is that they need the images from where the model is derived to be manually landmarked to allow the establishment of a point-to-point correspondence.

In this work, we present a segmentation method that combines Active Shape Models with Cage Active Contours [8], a parametric snakes framework that

makes use of computer graphics deformation techniques to deform the whole segmentation contour based on the relative position of all its pixels with respect to a reduced set of control points. These relative positions are parametrized using mean value coordinates [9], a generalization of barycentric coordinates to arbitrary planar polygons that is easy to compute and retain desirable smoothness properties. Constructing the Active Shape Model from the set control points frees us from having to manually identify landmarks on the images, while keeping the ability to establish a point-to-point correspondence to learn from. We call our method "Smart Cage Active Contours" (SCAC) in line with the alternative name "Smart Snakes" that Cootes and Taylor propose for Active Shape Models in [7].

In the experimental part of this work, we apply Smart Cage Active Contours to the segmentation of the left and right caudate nuclei of a set of 40 subjects. After the registration of all the magnetic resonance volumes to a normalized stereotactic space, the images are segmented slice by slice. The results are then compared with those obtained following a basic atlas-guided methodology [10].

1.1 Objectives of this work

We can identify two main objectives in the development of this work:

1. Develop a new segmentation method, "Smart Cage Active Contours" (SCAC), combining previous work on image segmentation using affine transformations, with Active Shape Models.

2. Evaluate the performance of this new method in the segmentation of subcortical structures in brain medical images and compare it with a classical atlas-based methodology.

1.2 Structure of this report

This report is structured as follows. First, we present the different techniques our work is based on, including deformable models, mean value coordinates, the Cage Active Contours (CAC) framework, Active Shape and Appearance Models and atlas-guided segmentation. Second, we explain how do we combine the previous techniques to construct the SCAC segmentation method. Third, we describe how do we apply SCAC to the particular case of the segmentation of the caudate nuclei in brain magnetic resonance images. Next, we describe our experimental setup, present the results obtained and discuss them. Last, we extract some conclusions and indicate possible future research directions.

2 Previous work

2.1 Deformable models

Deformable models are based on an initial contour or surface, ideally reasonably close to the desired region of interest, that evolves iteratively following a relaxation process to try to fit that region. The evolution of the deformable models if based in Physical principles: it is driven by internal and external (or image) forces. The internal forces are computed using only information from the boundary

curve or surface itself and tries to ensure smoothness. The external forces are calculated from the image to drive the boundary towards the desired region of interest. In some interactive models, a third type of forces called "constraint forces" is introduced, allowing the user to specify some points the curve has to pass through.

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:

$$\mathbf{r}_{t+1} = \mathbf{r}_t - \alpha \nabla E(\mathbf{r}_t) \tag{2.1}$$

where α is the evolution rate parameter, $\mathbf{r}$ is a vector describing the contour and $E(\mathbf{r}_t)$ is the energy functional to be minimized. The energy functional can be decomposed as

$$E(\mathbf{r}_t) = E_{int}(\mathbf{r}_t) + E_{ext}(\mathbf{r}_t) + E_c(\mathbf{r}_t)$$

where E_{int}, E_{ext} and E_c correspond to the internal, external and constraint energies, discussed in the next section.

Ideally we would like to use energy functionals that:

- Have few local minima. Otherwise, since the minimization algorithms are iterative, they are prone to get stuck easily.
- Have little dependence on starting points. Otherwise, if a small variation would lead to very different results, the method becomes chaotic and unstable.
- Continuous and twice differentiable. Some iterative optimization methods use the second derivative to estimate the size and direction of the step.

We will now concentrate on one of the most popular deformable models, known as active contour models or "snakes" [11]. Snakes are 2D deformable models, and are given this name because of the way the contour moves during evolution. In the snakes framework, the contour curve can be described as:

- Point-based snakes. The curve is an ordered collection of discrete points, called snaxels.
- Parametric snakes. The curve is defined in a continuous parametric form of the type $\mathbf{r}(s) = (x(s), y(s))$, using basis functions like b-splines or Fourier exponentials defined at certain knots in the curve.

Point-based snakes can be regarded as an extreme case of parametric snakes, where the curves are defined in terms of b-splines of degree zero with 2 parameters (position x and y) for each curve point. Point-based snakes thus generally require more parameters than parametric snakes, resulting on slower optimization algorithms. Moreover, parametric snakes can incorporate the smoothness and other a priori constraints in the basis functions used, eliminating the necessity of extra energy terms.

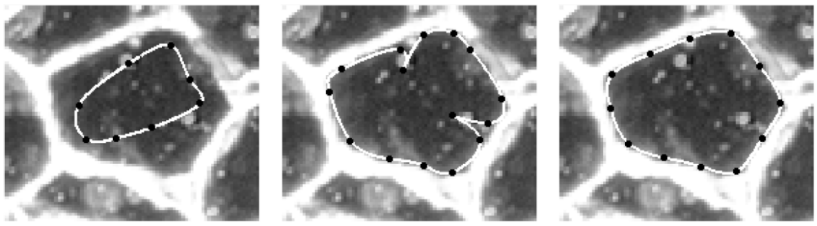


Figure 2.1 – From left to right, snake segmentation initialization, results without including the reparametrization energy and results including it. From [12].

2.1.1 Energies for parametric snakes

Internal energies

As have been already mentioned, INTERNAL ENERGIES try to ensure the smoothness of the contour. The most common internal energy is the one introduced by Kass [11], which is a linear combination of the length of the curve and its curvature:

$$E_{int} = \int_C \lambda_1(s) \left\|\mathbf{r}'(s)\right\|^2 ds + \int_C \lambda_2(s) \left\|\mathbf{r}''(s)\right\|^2 ds \tag{2.2}$$

By adjusting the weights λ_i of each term the relative importance of having shorter or smoother contours at each section of the curve can be tuned. The parameter λ_1 controls the "tension", while λ_2 controls the "rigidity".

As pointed out in [12], the second term only guarantees smoothness if the curve is parametrized by its curve length, i.e. if the parameter s is the curvilinear abscissa and the arc length between two knots is constant. To favor this condition in parametric snakes, [12] includes an additional term called "reparametrization energy" in the internal energy. Otherwise, the knot points can accumulate in a certain section of the curve and generate pointy curves (see Figure 2.1).

Constraint energies

In interactive setups, CONSTRAINT ENERGIES allow the user to enter a priori knowledge specifying the points the curve should pass through. An easy way to model this constraint into an energy, described in [12], is using the distance from that specified point to the closest point of the curve:

$$E_c = \sum_{i=1}^{N_c} \min_s \left\|\mathbf{r}(s) - \mathbf{r}_{c,i}\right\|^2 \tag{2.3}$$

where $\mathbf{r}_{c,i}$ are the N_c fixed points. This energy can be visualized as springs with one end at fixed points and the other end sliding on the curve.

External (or image) energies

IMAGE ENERGIES are the key element of the snakes framework to drive the contour curve to the desired area of interest. We can divide these energies into edge-based energies and region-based energies. The first ones compute a property of the curve considering only the pixels on the evolving contour or close to it,

while the second ones take into account pixels belonging to the inside or the outside of the curve, if it is closed.

Edge-based energies The most basic popular edge energy used is based on the gradient of the image intensity:

$$E_{edge} = -\int_C \|\nabla I(\mathbf{r}(s))\|^2 ds \tag{2.4}$$

where C denotes the contour, and the minus sign implies that the curve should evolve to lay over areas with a large intensity gradient, which are expected to be region boundaries. As pointed out by [13], a fundamental problem of this energy is that it does not incorporate information about the direction of the gradient. A more informed approach can try to maximize the component of the intensity gradient perpendicular to the curve:

$$E_{edge} = -\int_C \nabla I(r(s)) \cdot \hat{\mathbf{n}}(\mathbf{r}(s)) ds \tag{2.5}$$

where $\hat{\mathbf{n}}(r(s))$ is the unit vector perpendicular to the curve at the point $\mathbf{r}(s)$.

One of the potential problems of using gradient-based edge energies is that the presence of noise could make the contour curve sharp and pointy. However, as pointed out in [11], this problem is often avoided by the presence of internal smoothing energies, that act as energy wells to good local minima. Also, [11] proposes the use of a spatially blurred energy functional to evolve the snake first at a coarse scale and refine later, avoiding possibly misleading local minima.

Region-based energies Since many of the regions to segment involve closed contours, region-based energies, that use statistical information about the pixels inside and outside the segmentation regions, are becoming increasingly popular [14, 15, 5]. These kind of energies are especially useful to drive the snake towards the region of interest when it is still far away from it, since edge-based energies might not offer valuable information in this case.

A classical example is to segment a white object from a black background. If we initialize the contour far away from the object real boundary, the gradient energy will probably get stuck in a plateau, but an energy that gets minimized when all pixels inside the contour are white and all pixels outside it are black can correctly drive the evolving contour towards the boundaries of the white object.

From a generic statistical point of view, we assume that there are two segmentation regions (inside and outside) in the image that have different statistical properties. Therefore, given the statistical properties of both areas, we want to maximize the likelihood of the pixels determined to be in a certain region to really belong to that region. If we use log-likelihood and image intensity the problem can be formulated as minimizing the functional

$$\begin{aligned} E_{region} \quad = \quad & -\frac{1}{|S_{in}|} \sum_{\mathbf{p} \in S_{in}} log(P(I(\mathbf{p}) | \mathbf{p} \in R_{in}) \\ & -\frac{1}{|S_{out}|} \sum_{\mathbf{p} \in S_{out}} log(P(I(\mathbf{p}) | \mathbf{p} \in R_{out}) \end{aligned} \tag{2.6}$$

where R_{in} and R_{out} denote the different image regions, $S_{in}$and S_{out} denote the points that lay inside and outside the closed contour curve at that iteration, and $I(\mathbf{p})$ is the gray-level intensity of a pixel $\mathbf{p}$. The normalizing terms $|S_{in}|$ and $|S_{out}|$ denote the number of pixels inside and outside the region enclosed by the evolving contour, respectively.

Given an appropriate conditional probability functions $P(I(\mathbf{p})|\mathbf{p} \in R_{in/out})$, this energy attains a minimum when $S_{in} = R_{in}$ and $S_{out} = R_{out}$. For example, if we assume that the pixel intensities of both regions follow a Gaussian distribution [12, 16], the conditional probability can be expressed as

$$P(I(\mathbf{p})|\mathbf{p} \in R_j) = \frac{1}{\sqrt{2\pi}\sigma_j} \exp\left(-\frac{(I(p) - \mu_j)^2}{2\sigma_j^2}\right)$$

where μ_j and σ_j are the mean and the variance of the pixel intensities over the region R_j and $j \in \{in, out\}$. Using this conditional probability function, the region energy takes the form

$$\begin{aligned} E_{gauss} = \ & \log \sigma_{in} + \frac{1}{|S_{in}|} \sum_{\mathbf{p} \in S_{in}} \frac{(I(\mathbf{p}) - \mu_{in})^2}{\sigma_{in}^2} + \\ & + \log \sigma_{out} + \frac{1}{|S_{out}|} \sum_{\mathbf{p} \in S_{out}} \frac{(I(\mathbf{p}) - \mu_{out})^2}{\sigma_{out}^2} \end{aligned} \tag{2.7}$$

where the constant terms have been dropped since they have no influence in the energy minimization.

Similarly, [14] proposes a region energy based on the intensity distance to the mean, with good results:

$$E_{mean} = \frac{1}{|S_{in}|} \sum_{\mathbf{p} \in S_{in}} (I(\mathbf{p}) - \mu_{in})^2 + \frac{1}{|S_{out}|} \sum_{\mathbf{p} \in S_{out}} (I(\mathbf{p}) - \mu_{out})^2 \tag{2.8}$$

In both cases, the statistical parameters (mean and variance) can be either fixed manually or estimated at each iteration from the image regions of the current evolving contour:

$$\mu_{in/out} = \frac{1}{|S_{in/out}|} \sum_{\mathbf{p} \in S_{in/out}} I(\mathbf{p}) \tag{2.9}$$

$$\sigma_{in/out}^2 = \frac{1}{|S_{in}|} \sum_{\mathbf{p} \in S_{in/out}} (I(\mathbf{p}) - \mu_{in}(\mathbf{p}))^2 \tag{2.10}$$

The ability of fixing these parameters let us incorporate *a priori* information about the regions to be segmented in the snakes model, which can be incredibly useful. For example, if we know about the mean intensity of some organ in an image, we can use this to favor the algorithm to drive the contour to a region with this mean intensity.

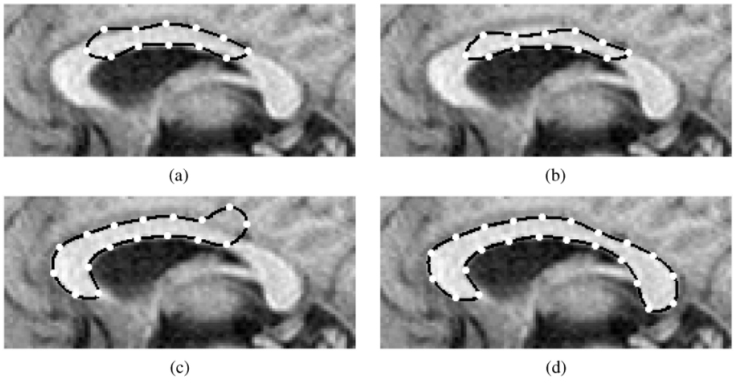


Figure 2.2 – Use of unified energy in image segmentation. (a) Initialization. (b) Use of gradient based energy ($\alpha = 1$) fails to converge in regions where the gradient information is absent. (c) Region-based energy ($\alpha = 0$) is misled by the lack of image contrast. (d) Unified energy ($\alpha = 0.5$) leads to a good segmentation. From [12].

Unified energies

Both edge- and region-based energies have advantages and disadvantages. Edge-based energies are good at precisely localizing the contour near the boundaries, but have a small basin of attraction and therefore fail to converge successfully if the contour is initialized too far away from the real boundary. Conversely, region-based energies have a larger basin of attraction, but fail at localizing explicit edges precisely.

To join the complementary advantages of both types of energy, [12] proposes an UNIFIED ENERGY resulting from the convex combination of them, which takes the form

$$E_{unified} = \alpha E_{edge} + (1 - \alpha) E_{region} \tag{2.11}$$

where α is a parameter that takes values in the interval $[0, 1]$ and allows us to tune the unified energy to each particular case. For example, if the images are noisy and the gradient information is unreliable, α can be set to low values to favor information coming from region-based energies. Conversely, in images where the regions cannot be associated to particular statistical properties (like a similar intensity) but the region edges are clear, values of α close to 1 can be used. See Figure 2.2 for an example of how unified energies can improve the segmentation results.

2.2 Mean value coordinates

2.2.1 Barycentric coordinates

An important problem in computer graphics is to linearly interpolate a function whose values are given at certain points of a closed contour. This has a number of applications, like shading [17, 18], deformation [19, 20, 21, 22], or parametrization [23] methods.

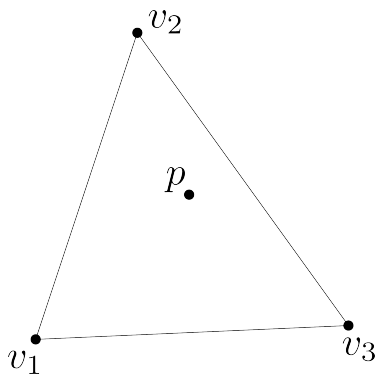


Figure 2.3 – Using barycentric coordinates, the position of the point $\mathbf{p}$ can be described as a linear combination of the position of the vertices of the triangle $\mathbf{v}_i$.

One solution for this interpolation are BARYCENTRIC COORDINATES, discovered by A.F. Möbius in 1827. Given any geometric triangle with vertices $\{\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3\}$, any point $\mathbf{p}$ inside the triangle (see Figure 2.3) can be described as a triple $\mathbf{w} = (w_1, w_2, w_3)$ of positive real numbers that corresponds to the masses that one would have to put at each of the triangle vertices for the point $\mathbf{p}$ to be the center of mass:

$$\mathbf{p} = \frac{w_1\mathbf{v}_1 + w_2\mathbf{v}_2 + w_3\mathbf{v}_3}{w_1 + w_2 + w_3} \tag{2.12}$$

This triple (w_1, w_2, w_3) is known as the barycentric coordinates of the point $\mathbf{p}$. Barycentric coordinates are only unique up to multiplication by a common non-zero scalar, and that is why often NORMALIZED BARYCENTRIC COORDINATES,

$$\phi_i = \frac{w_i}{\sum_j w_j} \tag{2.13}$$

satisfying $\sum_i \phi_i = 1$, are used.

Now, given the value of a (possibly vector-valued) function on the triangle vertices $\mathbf{f}_i = \mathbf{f}(\mathbf{v}_i)$, the value of the function on the interior point $\mathbf{p}$ can be interpolated as

$$\mathbf{f}(\mathbf{p}) = w_1\mathbf{f}_1 + w_2\mathbf{f}_2 + w_3\mathbf{f}_3 = \sum_{i=1}^{3} w_i\mathbf{f}_i \tag{2.14}$$

This function can be, for example, the color intensity at the vertices or a position transformation, allowing the calculation of the color value at any point inside the triangle.

2.2.2 Extension to arbitrary planar polygons: mean value coordinates

It would be desirable to extend the previous definition of barycentric coordinates to planar polygons of arbitrary shape (including non-convex), as well as to the whole $\mathbb{R}^2$ plane (including points outside the polygon). If these polygons are described by the vertices $\{\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_n\}$, one would like to find smooth

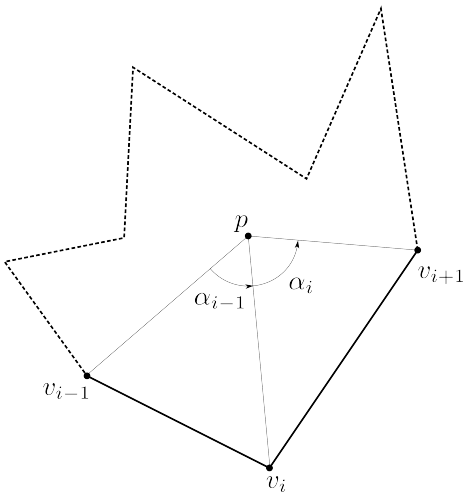


Figure 2.4 – Calculation of the mean value coordinates of the point $\mathbf{p}$ with respect to the vertex $\mathbf{v}_i$ of the polygon, represented by the thick line (both solid and dashed). α_{i-1} and α_i are the signed angles from the point $\mathbf{p}$ to the $(i-1)$-th and the i-th vertices, and to the i-th and the $(i+1)$-th vertices, respectively .

normalized barycentric coordinates $\{\phi_1, \phi_2, \ldots, \phi_n\}$, being $\phi_i : \mathbb{R}^2 \to \mathbb{R}$, such that, for any point $\mathbf{p}$,

$$\mathbf{p} = \sum_{i=1}^{n} \phi_i(\mathbf{p})\mathbf{v}_i \tag{2.15}$$

Additionally, to be valid for interpolation purposes, the Lagrange property must be fulfilled:

$$\phi_i(\mathbf{v}_j) = \delta_{i,j} \tag{2.16}$$

This is, the i-th generalized normalized barycentric coordinate of the i-th vertex is the unity, and the rest are zero. Should this not be true, the interpolated value of a function on the vertex would not coincide with the value coming from the vertex itself!

As discussed in [23], there is no easy obvious choice for such coordinates, since most choices either are not well-defined everywhere in $\mathbb{R}^2$ or do not satisfy the Lagrange property.

As a solution to this problem, MEAN VALUE COORDINATES were proposed by M. Floater in [9] for the interior of convex polygons and later extended to arbitrary planar polygons and the whole $\mathbb{R}^2$ plane in [23]. Let $\mathbf{p}$ represent a point in the $\mathbb{R}^2$ plane and Ψ an arbitrary planar polygon without self-intersections. As can be seen in Figure 2.4 the weight function with respect to the i-th polygon vertex $\mathbf{v}_i$ is defined as:

$$w_i(\mathbf{p}) = \frac{\tan(\alpha_{i-1}/2) + \tan(\alpha_i/2)}{\|\mathbf{v}_i - \mathbf{p}\|} \tag{2.17}$$

where α_{i-1} and α_i are the signed angles[1] from the point $\mathbf{p}$ the $(i-1)$-th and the i-th vertices, and to the i-th and the $(i+1)$-th vertices, respectively . The normalized mean value coordinates are defined as:

$$\phi_i(\mathbf{p}) = \begin{cases} w_i(\mathbf{p}) / \sum_{ij} w_j(\mathbf{p}) & \text{if } \mathbf{p} \notin \Psi, \text{ with } w_i \text{ from Eq. 2.17} \\ (1-\mu)\delta_{i,j} + \mu\delta_{i,j+1} & \text{if } \mathbf{p} \in (1-\mu)\mathbf{v}_j + \mu\mathbf{v}_{j+1} \in e_j \\ \delta_{i,j} & \text{if } \mathbf{p} = \mathbf{v}_j \end{cases} \tag{2.18}$$

where e_j denotes the edge of the polygon between the j-th and the $(j+1)$-th vertices. Note that the w_i are three-point coordinates, in the sense that they depend only on $\mathbf{v}_i$ and its two neighbors $\mathbf{v}_{i-1}$ and $\mathbf{v}_{i+1}$.

With this definition, the normalized mean value coordinates satisfy the following important properties:

1. Affine precision.
$$\mathbf{f}(\mathbf{p}) = \sum_i \phi_i(\mathbf{p})\mathbf{f}(\mathbf{v}_i) \tag{2.19}$$
for any affine function $\mathbf{f}$ to be interpolated.

2. Lagrange property. $\phi_i(\mathbf{v}_j) = \delta_{i,j}$.

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

4. Partition of unity. $\sum_i \phi_i = 1$.

5. Positivity. ϕ_i is positive inside star-shaped polygons.

The first property can be used for image warping. Given a gray-level image in a rectangular region $I : \Omega \to \mathbb{R}$ and a source polygon Ψ with vertices $\mathbf{v}_i$, and a target polygon $\hat{\Psi}$ with vertices $\hat{\mathbf{v}}_i$(both must have the same number of vertices) we would like to find a smooth warp function $f : \Omega \to \Omega$ that maps each $\mathbf{v}_i$ to its respective $\hat{\mathbf{v}}_i$. This warp function can be used to deform the source image I into a target image $\hat{I}$ by simply setting $\hat{I} = I \circ f^{-1} = I \circ g$. It can be shown [23] that the function

$$g(\mathbf{x}) = \sum_{i=1}^{n} \hat{\phi}_i(\mathbf{x})\mathbf{v}_i \tag{2.20}$$

where $\hat{\phi}_i(\mathbf{x})$ are the normalized mean value coordinates of $\mathbf{x}$ with respect to $\hat{\mathbf{v}}_i$, maps each $\hat{\mathbf{v}}_i$ to $\mathbf{v}_i$ and therefore it defines an appropriate warp function. To generate the warped image $\hat{I}$, the intensity of each pixel $\mathbf{x}$ in $\hat{I}$ is set to the intensity of the source pixel $g(\mathbf{x})$ in I, interpolating if necessary (see Figure 2.5 for an example).

2.3 Cage Active Contours

Cage Active Contours (CAC) [8] is a segmentation framework that combines computer graphic deformation techniques and parametrized active contours. It introduces the idea of parameterizing the contour according to a reduced set of

[1]The signed angle is positive if the involved vertices are traversed counterclockwise, and negative if they are traversed clockwise.

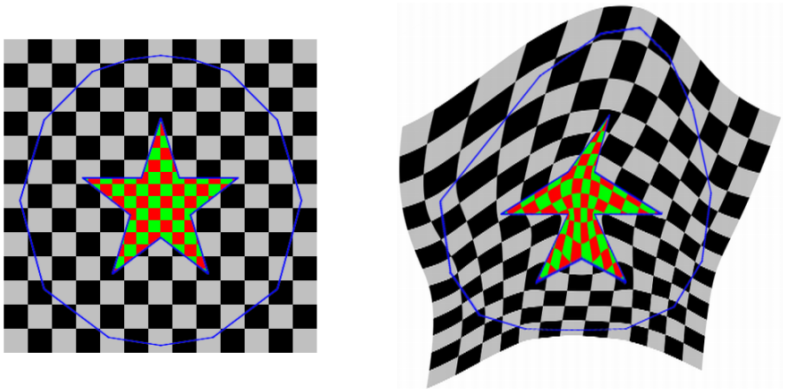


Figure 2.5 – Warping an image with mean value coordinates by moving the vertices of two nested polygons (one convex and one star-shaped), marked in blue. From [23].

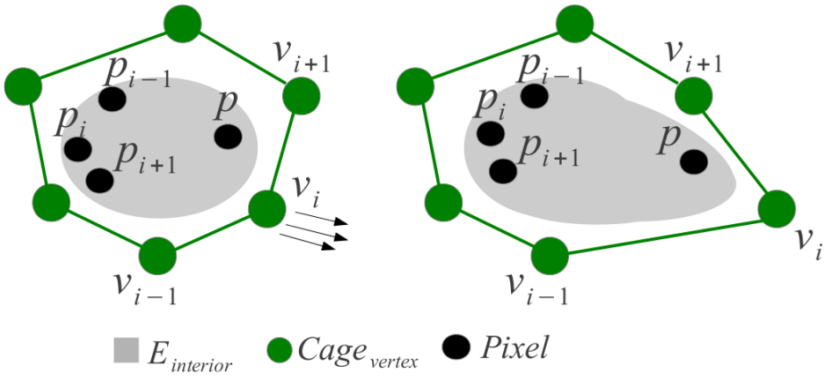


Figure 2.6 – Stretching example of a region using the CAC framework. The cage vertices $\mathbf{v}_i$ are the only ones allowed to move independently, being the displacement of the points p governed by those via an affine transformation. Note how the point $\mathbf{p}$ closer to the moving vertex $\mathbf{v}_i$ is the one that suffer a larger displacement, while the rest of marked points, farther from $\mathbf{v}_i$, are barely affected. Figure from [8].

control points, that drive the evolution of the contour. This parametrization uses mean value coordinates (see Section 2.2.2), a generalization of barycentric coordinates to arbitrary planar polygons.

The control points $\mathbf{v}_i$ are often disposed forming a rectangular shape that encloses the contour. These points are denominated "cage points", what motivates the name of the framework. During the segmentation optimization, the control points are the only ones allowed to move independently, being the contour points displacement governed by the evolution of these control points (see Figure 2.6). Actually, due to the use of region-based energies within the framework, not only the contour points are controlled by the cage points, but also a subset of the points inside and outside the segmentation region.

The mean value coordinates of the points of the evolving regions with respect to the cage control points $\phi_i(\mathbf{p})$ are calculated only once at the beginning of the optimization using Eq. (2.18). Every time a cage control point moves, the new

position of the region points can be computed (interpolated) for each of them[2] using Eq. (2.19) with $\mathbf{f}_i = \mathbf{v}_i$:

$$\mathbf{p} = \sum_{i=1}^{n} \phi_i(\mathbf{p})\mathbf{v}_i \tag{2.21}$$

Due to the nature of the mean value coordinates, the closer an image point is to a cage control point, the more influence has this last one over the image point. That is, when a cage control point moves, the image points closer to it are the ones that suffer a larger displacement, while the image points far from it are barely affected (see Figure 2.6).

Since the only points that can move independently are the cage control points $\mathbf{v}_j$, the optimization is formulated in terms of these control points, i.e. we look for the configuration of $\mathbf{v}_j$ that minimizes the energy functional. Therefore the gradients are expressed with respect to $\mathbf{v}_j$, but thanks to the linearity of Eq. (2.21) their computation is straightforward. The derivative with respect to the variation of the x position of the j-th control point v_j^x is:

$$\frac{\partial E}{\partial v_j^x} = \sum_{\mathbf{p}} \frac{\partial E}{\partial p_x}\frac{\partial p_x}{\partial v_j^x} = \sum_{\mathbf{p}} \frac{\partial E}{\partial p_x}\phi_j(\mathbf{p})$$

and analogously for v_j^x:

$$\frac{\partial E}{\partial v_j^y} = \sum_{\mathbf{p}} \frac{\partial E}{\partial p_y}\frac{\partial p_y}{\partial v_j^x} = \sum_{\mathbf{p}} \frac{\partial E}{\partial p_y}\phi_j(\mathbf{p})$$

The latest two equations can be combined into a more compact form:

$$\nabla_{\mathbf{v}_j} E = \sum_{\mathbf{p}} \phi_j(\mathbf{p})(\nabla_{\mathbf{p}} E) \tag{2.22}$$

The CAC optimization process given the initial position of all control cage points and the initial mask is summarized in Algorithm 1. Here we would like to detail only the first step of the algorithm.

The algorithm is started by means of a binary mask image from which the internal S_{in} and external S_{out} pixels are extracted. In those, pixels that are assumed to belong to the regions of interest are marked with 1 whereas the rest are marked with 0. This binary image is automatically extracted and is an approximation to the real object of interest.

The starting segmentation region is specified as a mask image: A binary image where the pixels within the segmentation region have value 1 and the rest of them (outside the segmentation region) have value 0. The dimensions of the mask image and the image where segmentation optimization is to be performed must coincide.

From the mask image, the position of the pixels inside, outside and at the contour of the segmentation region are obtained using standard image processing techniques:

[2]Both the computation of the mean value coordinates and of the new position for each point are completely independent and therefore are open to parallelization.

Algorithm 1 CAC optimization procedure.

1. Extract the contour, in and out evolving regions from the mask.
2. Calculate the mean value coordinates of the pixel points $\phi_i(\mathbf{p})$ in the previous regions with respect to the initial cage control points $\mathbf{v}_j$ using Eq. (2.18). These mean value coordinates remain fixed during the whole optimization process.
3. While the termination condition is not fulfilled:
   (a) Recover the position of the points of every region using Eq. (2.21).
   (b) Calculate the energy of the current configuration of regions. In [8] edge- and region-based image energies, as well as an internal energy that favors shorter-length contours, are used.
   (c) Calculate the gradient of the energy being used with respect to the movement of the vertices, using Eq. (2.22).
   (d) Use a gradient descent method or L-BFGS (a limited memory quasi-Newton optimization method) to update the position of the control cage points $\mathbf{v}_j$ to a position of smaller energy, if possible.

- The inside region S_{in} corresponds to all pixels with value 1 in the mask image.[3]
- The outside region S_{out} does not correspond to all the pixels of the image that are not part of the inside region, but constitute a thin band near the contour:
$$S_{out} = \text{Dilation}(S_{in}, d_{out}) - S_{in} \tag{2.23}$$
where d_{out} is the dilation width (see Figure 2.7). This way we can avoid considering image information far from the active contour that is not important for the optimization process and can even affect it negatively.
- The contour C is extracted from the binary mask following the boundary of the pixels with value 1 with the pixels with value 0 either clockwise or counterclockwise.

In all cases the position of the points of the contour is described parametrically in terms of the position of the cage control points v_j, and therefore Cage Active Contours is regarded a parametric snake method (cf. Section 2.1).

An interesting property of the Cage Active Contour method is that the smoothness of the solution is preserved directly thanks to the position interpolation process. Therefore there is no need to introduce an additional internal energy term like the one of Eq. (2.2) to favor smoothness.

The framework was evaluated using synthetic and real-world images showing reasonably good results, but leaving large room for improvements in the field of

[3] In the SCAC implementation we use an inner band near the contour instead (see Figure 3.2 at page 31).

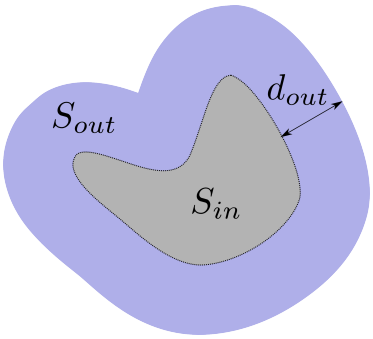


Figure 2.7 – Dilation to find the S_{out} region. The dotted line corresponds to the segmentation contour.

medical imaging. One of the possible improvements is to combine Cage Active Contours with Active Shape Models, described in the next section.

2.4 Active Shape and Appearance Models

Since their publication in 1995, Active Shape Models (ASM) [7] have been a popular technique to incorporate a priori knowledge about the acceptable ways a contour model can deform to fit desired boundaries. The basic premise behind ASM is that a model should only be able to deform in ways characteristic of the class of objects it represents. For example, a model for the contour of an apple can deform to account for the varieties in size and relative shape apples can have, but the probability of that shape to become triangular is very low.

In ASM, contour models are built by learning patterns of variability from a representative training set of landmarked images. These models can afterwards be used to fit image boundaries using methods like parametric snakes, constrained so these snakes can only deform in ways consistent with the training set, increasing robustness. Although the framework is general, the shape and appearance models built are specific for each application.

2.4.1 Building the model

This technique relies on objects to be represented by a set of landmark points that are placed manually in the same logical positions in all training examples. These points can represent either the boundary of the object or any feature or point of interest inside it. In order to compare relative displacement of equivalent points from different shapes, they must first be aligned with respect to the same set of axes. To do so, the shapes of the training set are rotated, translated and aligned with their mean using an iterative method to minimize the squared distance between equivalent points (see Figure 2.8).

After the shapes have been aligned, the variations in the positions of each of the points along the training set is analyzed to build a so-called Point Distribution Model (PDM) that encodes the main modes of variation of the points. These modes of variation are uncorrelated between them and can be used as parameters to represent the shapes. This way, simple limits on each parameter act as constraints so only shapes similar to those in the training set

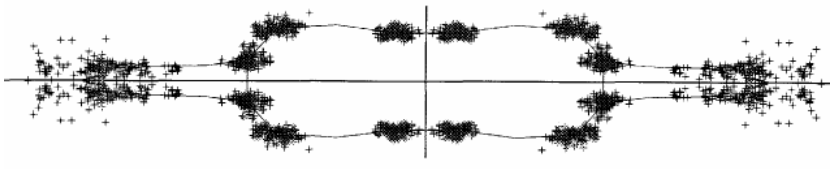


Figure 2.8 – Scatter of some points from aligned set of resistor shapes, with the mean shape overlaid. Figure from [7].

can be generated.

To construct this PDM, each example shape of the training set in the coordinate frame of the aligned shape is represented by a single point $\mathbf{x}$ in a $2n$-dimensional space, being n the number of landmark points:

$$\mathbf{x} = (x_1, y_1, x_2, y_2, \ldots, x_n, y_n)$$

If there are N examples, the N points in this $2n$-dimensional space are expected to lay in a region of space that represents the allowable shapes of the object being modelled. Assuming that the allowable shapes region is an ellipsoid[4], Principal Component Analysis (PCA) [24] is applied to the data to extract the principal modes of variation as orthogonal deviations from the mean. This way, any shape can be represented as

$$\mathbf{x} = \bar{\mathbf{x}} + P\mathbf{b} \tag{2.24}$$

where $\bar{\mathbf{x}} = (\bar{x}, \bar{y})$ is the mean of the positions of every point in the coordinate frame of the aligned shape, P is the matrix whose columns are the eigenvectors $\mathbf{P}_i$ $(i = 1, 2, \ldots, n)$ corresponding to the principal components and $\mathbf{b}$ is a $2n$-dimensional vector parameter that represents the linear decomposition of the deformation of the shape in terms of the determined principal components. Usually, only the first r components with higher corresponding eigenvalues λ_i, which accumulate most of the variance, are taken into account, which are often enough to approximate every shape reasonably well. Mathematically

$$r = \min \left\{ k \text{ such that } \sum_{i=1}^{k} \lambda_i \geq e_{var} \right\}$$

where e_{var} is the accumulated variance limit. In this last case, $i = 1, 2, ..., r$ and the parameter vector $\mathbf{b}$ would have dimension r.

To restrict the model to generate only plausible shapes, limits are imposed upon the values of the linear parameter $\mathbf{b}$. If the variance of each main component correspond to λ_i, [7] states that suitable limits are of the order of

$$-3\sqrt{\lambda_i} \leq b_i \leq 3\sqrt{\lambda_i}, \tag{2.25}$$

where b_i is the coefficient of $\mathbf{b}$ corresponding to the i-th principal component $(i = 1, 2, \ldots, r)$. It is important to note that, in general, imposing equivalent

[4]This implies the assumption of that the variation modes of the PDM move the landmark points along straight lines. The assumption fails when bending or relative rotational effects occur in the shapes (cf. [7]).

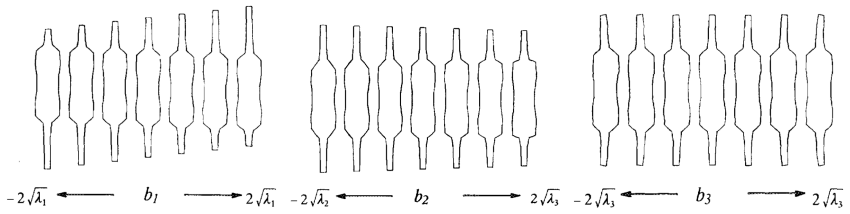


Figure 2.9 – Effects of varying the first (left), second (center) and third (right) parameters of a resistor model. Figure from [7].

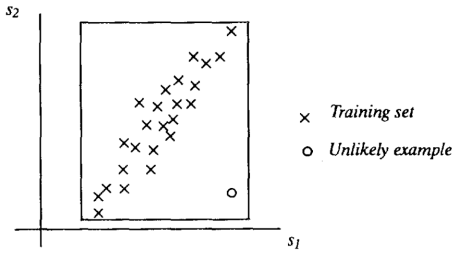


Figure 2.10 – If two or more shape parameters (s_1 and s_2) are correlated over a set of shapes then simple ranges to the parameters do not restrict shapes to ones similar to those in the original set. Figure from [7].

limits over the original coordinates of the points of the shapes does not guarantee that the model generates only allowed shapes, because of the possible correlations between coordinates (see Figure 2.10). Here, PCA ensures that all the principal components are orthogonal (non-correlated).

2.4.2 Using the model in image search

After having constructed the shape model for the objects of our interest, we would like to use it to find new examples of objects in images. For example, if we have modelled the contour of an apple, we would like to input an unannotated image of an apple and use an algorithm based on the model to find its contour intelligently. This algorithm is very similar to the one used for active shape models (snakes), with certain constraints.

Starting from an initial contour that can be, for example, the mean shape, we proceed iteratively. Let $\mathbf{X} = (X_1, Y_1, ..., X_n, Y_n)$ represent the current position in the Cartesian frame of the landmark points of the model, initialized using the mean shape, Genetic Algorithms or taking into account any prior knowledge. At each step, we first use the classical tools from parametric snakes optimization (see Section 2.1), to find the expected optimal adjustments to the positions of the points $d\mathbf{X} = (dX_1, dY_1, ..., dX_n, dY_n)$ from the current estimate $\mathbf{X}$. From this, we find the translation t, rotation θ and scaling factor s that better maps the current estimate $\mathbf{X}$ to $(\mathbf{X} + d\mathbf{X})$, using standard matrix methods [7]. If this transformation is denoted by $T(s, \theta, t)[\mathbf{X}]$, the problem consists on finding the parameters s, θ and t such that the norm of the residual adjustments $d\mathbf{x} = (\mathbf{X} + d\mathbf{X}) - T(s, \theta, t)[\mathbf{X}]$ is minimized:

$$\arg\min_{s,\theta,t} \|(\mathbf{X} + d\mathbf{X}) - T(s, \theta, t)[\mathbf{X}]\|$$

Now we can calculate the change in the model parameters $d\mathbf{b}$ to account for the residual adjustments $d\mathbf{x}$ needed. Note that this deformation will not be fully achievable in general, since the shape is only able to deform in the modes imposed by the selected principal components of the model constructed.

Following Eq. 2.24, we wish to find $d\mathbf{b}$ such that

$$\mathbf{x} + d\mathbf{x} \approx \mathbf{x} + P(\mathbf{b} + d\mathbf{b}) \tag{2.26}$$

The variation in the mode coefficients to approximate the deformation is therefore given by

$$d\mathbf{b} = P^T d\mathbf{x}, \tag{2.27}$$

where we can impose the limits stated above on the values of b_k to deform only to shapes consistent with the training set.

2.4.3 Active Appearance Models

Active Appearance Models (AAM) [25] combine the statistical model of shape of Active Shape Models with a statistical model of the (gray-level) intensity appearance of the object of interest, in a shape-normalized frame. This way, the model can not only accommodate to the shape of new valid image examples, but can also synthetise them using the gray-level model. Continuing with the example of the apple, with AAM we will not only be able to find its contour, but also to generate a synthetic image of the apple itself.

As in ASM, a set of training images are landmarked, aligned and analyzed using PCA to construct a Point Distribution Model of their shapes, controllable using a finite set of shape parameters $\mathbf{b}_s$ in the linear model:

$$\mathbf{x} = \bar{\mathbf{x}} + P_s \mathbf{b}_s \tag{2.28}$$

To build the appearance model, all training images are first warped to match the mean shape (using a triangulation algorithm) and normalized. The gray-level appearance of any shape-normalized training image can then be described by a vector $\mathbf{g}$. Applying PCA to the $\mathbf{g}$ vector of all images we obtain the linear model

$$\mathbf{g} = \bar{\mathbf{g}} + P_g \mathbf{b}_g, \tag{2.29}$$

where $\bar{\mathbf{g}}$ is the mean normalized gray vector, P_g is a matrix whose columns correspond to the orthogonal modes of intensity variation and $\mathbf{b}_g$ is a set of gray-level parameters. It is important to note that both parameters control the model in a normalized frame: $\mathbf{b}_s$ in the normalized frame of aligned (rotated, translated and resized) shapes and $\mathbf{b}_g$ in the normalized frame of images warped to the mean shape.

Now the approximate shape and appearance of any training image can be encoded in the parameter vectors $\mathbf{b}_s$ and $\mathbf{b}_g$. To avoid any possible correlations between the shape and the gray-level variations, a concatenated vector parameter $\mathbf{b} = (W_s \mathbf{b}_s, \mathbf{b}_g)$ is generated (where W_s is just a diagonal matrix of weights to account for the difference in parameter units) joining both shape and gray-level parameters. Applying PCA to it gives a further model

$$\mathbf{b} = Q\mathbf{c} \tag{2.30}$$

Figure 2.11 – AAM applied to faces. Left: first two modes of shape variation. Right: first two modes of gray-level variation. Figure from [25].

where $Q = (Q_s, Q_g)^T$ is the matrix of eigenvectors and $\mathbf{c}$ the appearance parameters, that now encode both the shape and gray-levels of the model:

$$\mathbf{x} = \bar{\mathbf{x}} + P_s W_s Q_s \mathbf{c} \tag{2.31}$$

$$\mathbf{g} = \bar{\mathbf{g}} + P_g Q_g \mathbf{c} \tag{2.32}$$

To synthesize an image given $\mathbf{c}$, we can generate the shape normalized image from the vector $\mathbf{g}$ using Eq. (2.32) and then warp it to the control points described by $\mathbf{x}$ using Eq. (2.31) and a triangulation algorithm (see Figure 2.11 for an example application to faces).

The method presented above is designed to work with 2D images, but 3D approaches have also been developed [26, 27, 28].

Active Appearance Models Search We now turn to the central problem these models try to address: Given a new unlandmarked image containing an object an appearance model has been built for, adjust the parameters $\mathbf{c}$ so that a synthetic example can be generated that matches the image as closely as possible, i.e. minimize the intensity difference $|\delta\mathbf{I}|^2$, being

$$\delta\mathbf{I} = \mathbf{I_i} - \mathbf{I}_m \tag{2.33}$$

where $\mathbf{I}_i$ is the intensity of the pixels of the real image and $\mathbf{I}_m$ the intensity of the pixels of the image synthesized by the model. To ease this apparently difficult, high-dimensional optimization problem, [25] assumes a linear relationship between the image intensities and the parameters[5]:

$$\delta\mathbf{c} = A\delta\mathbf{I} \tag{2.34}$$

where A is a matrix found by multivariate linear regression on a sample of known model displacements $\delta\mathbf{c}$ and the corresponding difference images $\delta\mathbf{I}$, generated by a perturbation of the parameters of real or synthesized images.

The procedure to fit the model to a new input image given an initial estimate of the model parameters $\mathbf{c}_0$ is summarized in Algorithm 2. The use of a multi-resolution implementation that iterate to convergence at each level before going to the next level of detail has reported a faster and more robust convergence.

[5]This assumption is later experimentally shown to be valid only over a limited range of values of the parameters $\mathbf{c}$: around 0.5 standard deviations from the mean.

Algorithm 2 Image search in AAM

1. Repeat until no improvement can be made or convergence is declared:
   (a) Synthesize a model image $\mathbf{g}_m$ from $\mathbf{c}_0$.
   (b) Calculate the shape positions $\mathbf{x}$ from $\mathbf{c}_0$ and warp the input image to these positions, obtaining the sample image $\mathbf{g}_s$.
   (c) Evaluate the current error vector $\delta\mathbf{g}_0 = \mathbf{g}_s - \mathbf{g}_m$.
   (d) Compute the predicted displacement $\delta\mathbf{c} = A\delta\mathbf{g}$.
   (e) Perform a line search using $\mathbf{c}_1 = \mathbf{c}_0 - k\delta\mathbf{c}$, with k from 1 to 0.25 (gradient descent), aiming to reduce the magnitude of the error $|\delta\mathbf{g}_1|^2$.

2.4.4 ASM and AAM for medical imaging

In [6], the shape of the left ventricle of the heart as seen in an electrocardiogram is encoded into an ASM, marking 11 key positions and generating the rest of landmark points by equally spacing points along the boundaries between these key positions. An appearance model is then constructed from the one-dimensional gray-level profiles normal to the contour at each landmark point. This model is also applied to a 3D model of the brain ventricles, using the projected segmentation of each slice as the starting boundary for the next one.

In [29] Active Appearance Models are used to construct a deformable brain atlas (see Section 2.5 for an introduction to atlas-guided segmentation) model from a set of manually landmarked images. Following the AAM algorithm described above, the atlas can match new images of brain MR cross-sections by minimizing the pixel/voxel differences.

In [30], a composite 3D AAM of the surfaces of multiple brain structures is constructed, as well as an individual model for the left and right caudate nuclei. Segmentation starts with affine registration to initialize the model within the image, then a search using the composite model. This provides a reliable but coarse segmentation, used to initialize the search with the individual caudate models.

A novel feature present in this last work is that the models are built directly from the labelled volumes, without the need of manually placing corresponding landmarks on each training image. Following the steps described in [31], the image volumes are divided in multiple quadrahedra. The appearance of the pixels inside each quadrahedrum is controlled, via an affine transformation, by the movement of the vertices of the figure, that act as control points (cf. Figure 2.12). The Point Distribution Model is then built using the control points as landmarks. A similar approach is presented by the same authors in [32].

2.5 Atlas-guided segmentation

As mentioned in the Introduction, automated medical image segmentation is a challenging task due to the high variability of the properties of the structures to be segmented as well as image artifacts resulting from partial volume effects, the presence of noise and motion blur or the lack of contrast or other clear

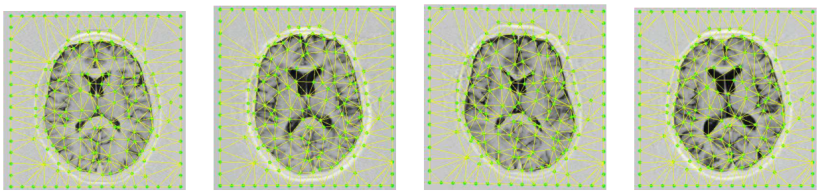


Figure 2.12 – Example slices with control points superimposed. Figure from [31].

visual boundaries between structures. Therefore, introducing *a priori* anatomical information is crucial for simplifying the segmentation process. We have already seen an example of how this can be done in practice in the case of region-based energies: manually specifying the mean and/or the variance of the intensity of the regions.

ATLAS-GUIDED SEGMENTATION is a powerful template-based technique that uses all this anatomical *a priori* information. There, an anatomical map of the region of interest (e.g. the brain), which constitutes the so-called atlas, is constructed and labelled manually. Labelling (segmenting) an input image then reduces to finding an appropriate transformation that aligns the input image with the atlas, and then propagating the atlas labels to the image.

One can think of atlas-guided labelling as a process equivalent to transferring to a an aerial photography taken from a plane the labels of a map of the area. The features present in the photo (coastline shape, mountains, etc.) can be used to match it to the map. Once the transformation that establish a correct correspondence between the photo and the map has been found, all points of interest can be propagated from the map to the photo itself. Figure 25 shows an example of aerial image registration using feature matching.

The main shortcoming of atlas-guided segmentation is that it depends strongly on the images to be similar to the atlas used. Choosing a atlas which is too different might lead to severe segmentation errors. Also, if the structure to be segmented shows a high variability, it might not be possible to find a suitable atlas that works well with the whole population of study. To overcome this last issue, models using multiple atlases have been developed. The atlases are selected using meta-information such as age, sex, disease, etc. or using similarity metrics. To combine the results from all the atlases selected, voting rules are commonly applied [34].

2.5.1 Single-subject and probabilistic atlases

In the context of medical imaging, an ATLAS can be defined [34] as a combination of two images: an intensity image, used as a standardized template, and a segmented image with the labels of every region of the template. Mathematically, an atlas can be defined as two mappings from points in a n-dimensional space to intensity values and labels, respectively,

$$A : \Omega_A \subset \mathbb{R}^n \to \mathbb{R}$$
$$L_A : \Omega_A \subset \mathbb{R}^n \to \mathcal{L}$$

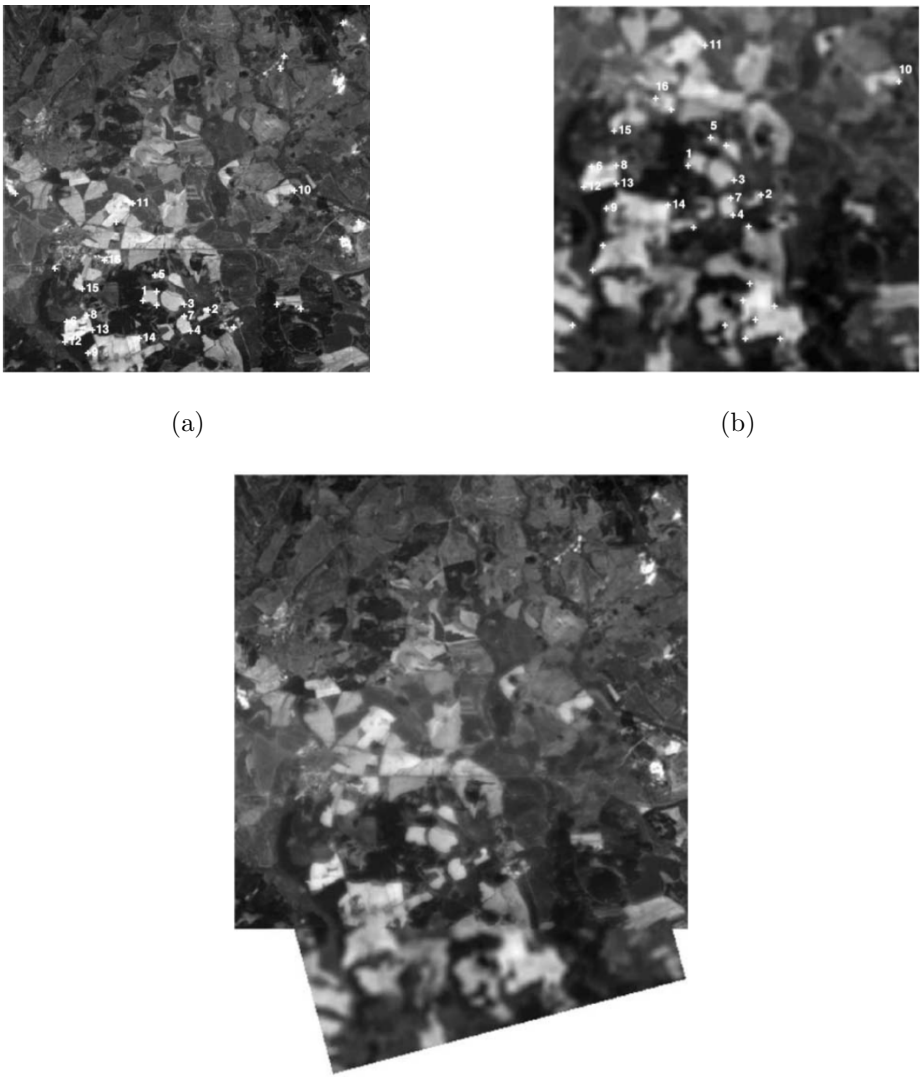


Figure 2.13 – Registration of two satellite images using feature matching. The numbered spots on images (a) and (b) correspond to the matched features. This information is later used to deform the images and construct the registration result (c). Figure from [33].

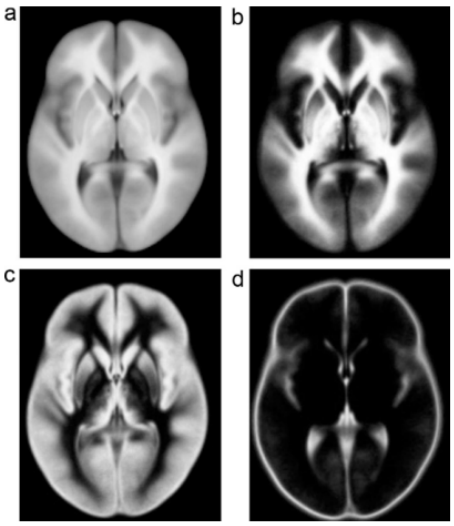


Figure 2.14 – Tissue probability maps of the International Consortium for Brain Mapping (ICBM) probabilistic atlas, constructed over MR images of the brain of 452 subjects. (a) MR image, (b) white matter probability map, (c) gray matter probability map, (d) cerebrospinal fluid probability map. Figure from [34]. This and other probabilistic atlases can be downloaded from http://www.loni.ucla.edu/ICBM/Downloads/Downloads_ICBMprobabilistic.shtml.

where the first mapping corresponds to the template image (here gray-level) and the second one to the labelling image. Here $\mathcal{L} = \{1, 2, 3..., C\}$, where C is the number of classes, and Ω_A represents the domain of the atlas images. In medical imaging, atlas-guided segmentation is often applied to MRI volumes, and therefore most atlases have spatial dimension $n = 3$.

The atlas definition given above correspond to TOPOLOGICAL, SINGLE-SUBJECT, DETERMINISTIC ATLASES such as the popular Talairach atlas for the brain [35]. To better characterize the inter-subject variability of the structures to be segmented, a number of so-called POPULATION-BASED, PROBABILISTIC OR STATISTICAL ATLASES have been constructed. The procedure to construct these atlases usually consists on registering MR images coming from several hundreds of subjects to a well-known deterministic atlas (the Talairach one in the case of brain images), normalizing the intensities of all images and finally averaging the results of all them to create probabilistic maps for each type of tissue (see Figure 2.14 for an example). In this last case, we can define a probability function for each correspondingly labelled volume:

$$L_{A,c} : \Omega_A \rightarrow [0, 1]$$

where $c \in \mathcal{L}$.

2.5.2 Image registration

As pointed out above, once an atlas has been selected, segmenting an input image reduces to matching the image to the atlas template (warping it) using

a parametrized transformation for all voxels and then propagating the atlas labels back to the image. If the second image to be aligned is defined as $B : \Omega_B \subset \mathbb{R}^n \to \mathbb{R}$, then the transformation that maps the coordinates of that image to the coordinates of the atlas image can be expressed as $T : \mathbb{R}^n \to \mathbb{R}^n$. The key process of finding the appropriate transformation mapping T that aligns both images is known as IMAGE REGISTRATION and constitutes an active field of study and research on itself.

Note that for the image registration algorithms to work, in addition to the coordinate transformation T, we need an interpolation mechanism to transform the input image to the space of the reference image in a way we can overlay both images and see the differences. The problems that might arise from the differences in the sample spacing of both images are discussed in [36].

For deterministic atlases the labelling function T_L can be defined as

$$\begin{aligned} T_L : &\Omega_B \to \mathcal{L} \\ &\mathbf{x} \ \to L_A(T(\mathbf{x})) \end{aligned}$$

In the case of population-based atlases, an analogous function can be defined for each class, that determines the probability of any voxel of belonging to that class.

There are already plenty of good surveys about medical image registration methods in the literature [36, 37, 33, 34]. Here we only provide some information about the two main types of registration transformations available.

Types of transformations

Image registration for automatic segmentation of complex organs like the brain is usually decomposed in two steps.

The first one is used to obtain a coarse, computationally cheap, initial alignment corresponding to an AFFINE OR RIGID TRANSFORMATION to correct for the gross image differences. In rigid transformations all distances between points in the image are preserved. Rigid transformations have a total of 6 degrees of freedom: three translations and three rotations. In addition to rigid-body transformations, some algorithms add three degrees of freedom for anisotropic scaling and another three for skewing, resulting in a total of 12 degrees of freedom. These last type of transformations are called AFFINE TRANSFORMATIONS and have the properties of that they can be written in matrix form and that all the parallel lines are preserved.

This kind of image registration is sometimes used for intra-subject registration, since it can be used to align images of single subject but that come from different image modalities or are deformed due to the characteristics of the acquisition equipment, movements of the patient, etc.

Since there are many organs that do not only stretch or shear, a second step of local, NON-AFFINE REGISTRATION is often used to adapt the general model to an specific anatomy, with a higher computational cost. This type of registration is usually used for inter-subject registration, although it can also be used, for example, to study the evolution of a certain organ within a single subject.

We can regard non-affine registration as a deformation field that indicates the displacement needed at each voxel in one image to align it with the corresponding

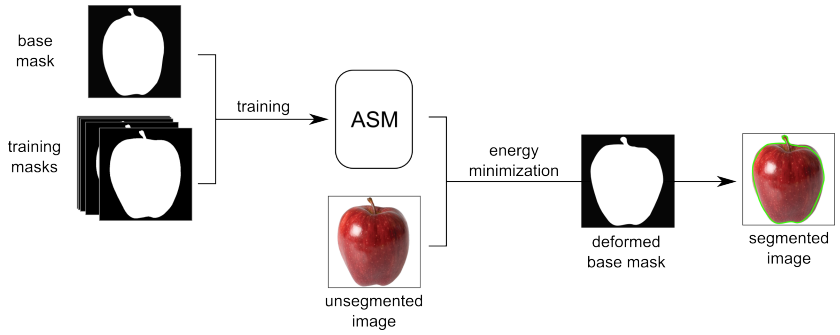


Figure 3.1 – Overview of the SCAC approach applied to the segmentation of an apple. The ASM is constructed from the deformations that map the base mask to the training masks, that correspond to example shapes of apples. This ASM is later used during energy minimization using Cage Active Contours to segment the image of the apple.

voxel of a second image. This alignment can be performed using a variety of methods like thin-plate splines, a linear combination of polynomial terms, b-spline surfaces, fluid or elastic differential equations [36].

3 Smart Cage Active Contours

In this part of the work we explain how do we combine the Cage Active Contours (CAC) framework with Active Shape Models (ASM), both described in Section 2, to construct the Smart Cage Active Contours (SCAC) segmentation method.

Let us assume that we have a set of K binary masks M_i $(i = 1, 2, \ldots, K)$ representing the shape of a certain type of structure we want to model, indicating for all pixels of each image, which ones correspond to the structure and which ones do not. We will call this set of mask images "training masks".

Let us further assume that we have an unsegmented image where the cited structure, that we want to segment, appears.

The steps of the method are illustrated in Figure 3.1 and can be summarized as follows:

1. Construct an Active Shape Model from the set of training masks, using CAC and an energy based on the sum of squared intensity differences between a base mask and the manual segmentations.

2. Use CAC with edge- or region-based energies to segment the image by adjusting the (constrained) parameters of the constructed ASM.

These steps are explained in the following subsections in detail.

In this work we will assume that all images are adequately aligned, translated and scaled so there is no alignment step necessary in the ASM procedure and therefore all deformations are expressed with respect to the local, aligned coordinate frame.

Notation

In the next sections the following notation is used:

- $\Omega \subset \mathbb{R}^2$ denotes the space of the images. Since the images are not infinite but have limits, the region cannot be the whole $\mathbb{R}^2$ plane but a subset, usually rectangular-shaped.
- $I : \Omega \to [0, 1]$ represents an image. The image is described as a mapping that assigns an intensity value between 0 (black) and 1 (white) to the points of the plane inside the image region.
- $\mathbf{p} = (x, y) \in \Omega$ denotes the coordinates a 2D point in the plane.
- $\Phi(\mathbf{p}) = (\Phi_x(\mathbf{p}), \Phi_y(\mathbf{p})) = \hat{\mathbf{p}}$ denotes the coordinates where the point $\mathbf{p}$ moves after being applied the affine transformation Φ.

3.1 Construction of the Active Shape Model

Here we describe the process to construct an Active Shape Model that encodes the possible deformations from the mean shape to generate the different shapes of the considered structure. The process is summarized in Algorithm (3).

Note that the base mask M_0 and the initial positions of the cage control points $\mathbf{c}_0$ are part of both the input and the output. This is to emphasize that the ASM is constructed using these M_0 and $\mathbf{c}_0$ as reference and will not be valid if it is used with a different initialization. Therefore, the base mask and the initial positions of the cage control points are considered "part" of the ASM itself.

3.1.1 Finding the cage deformations

As mentioned in the Introduction, one of the disadvantages of the original ASM methodology is that a set of equivalent landmark points needs to be defined on each training image, typically over the contour shape. These points have to be placed at the same "logical" positions of the shape on every image for the method to work correctly. Otherwise the model would be unable to represent correctly the position of each point since it will include terms describing the noise caused by errors in point locations.

To circumvent this issue, rather than constructing a PDM of the points that constitute the contour, we construct instead a PDM of the cage points that control the contour. This PDM will encode the variations on the positions of the cage control points $\mathbf{v}_j = (v_j^x, v_j^y)$ to deform a base mask M_0 to match the manual segmentation M_i (taken here as ground truth) of the structure at each training image. This base mask should be general enough to be able to accommodate to the different structure shapes present in the training set, and the best way to determine it depends on the particular application of the SCAC method.

Assume we are given a certain M_0 which corresponds to a given (binary) base segmentation mask. Our purpose is to deform M_0 to match each M_i, as described below. In the experimental Section 5 we will see how M_0 is defined.

To find out which cage deformation corresponds to this matching we make use of the CAC framework with a custom energy function based on the sum

Algorithm 3 Construction of the Active Shape Model for SCAC

Input:

- Set of training masks M_i, where $i = 1, \ldots, K$.
- Base mask M_0.
- Initial positions of the cage control points $\mathbf{c}_0$.
- Contour C, inner region S_{in} and outer region S_{out} extracted from the base mask of the ASM M_0.

Output:

- ASM including:
  - The r principal component vectors $\mathbf{P}_i$
  - Their r associated eigenvalues λ_i
  - The mean configuration of control points $\bar{\mathbf{c}}$.
  - The base mask M_0 (same as input).
  - The initial positions of the cage control points $\mathbf{c}_0$ (same as input).

Procedure:

1. For all training masks M_i:
   (a) Set the initial position of $\mathbf{c}_i$ to $\mathbf{c}_0$.
   (b) While the termination condition is not fulfilled:
      i. Find the current position in the image of the points of C, S_{in} and S_{out}, that are parametrized with respect to $\mathbf{c}_i$, using Eq. (2.15).
      ii. Compute the image energy for the current configuration using Eq. (3.1).
      iii. Calculate the image energy gradients using Eqs. (3.5) and (3.6).
      iv. Use the gradient descent algorithm to decide the next value of $\mathbf{c}_i$, that lowers the total energy, if it exists.
2. Apply PCA to $\{\mathbf{c}_i\}_{i=1}^{K}$, obtaining the r principal components $\mathbf{P}_i$, their associated eigenvalues λ_i and the mean configuration of control points $\bar{\mathbf{c}}$.

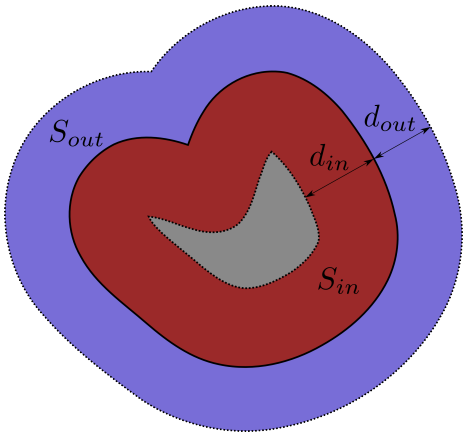


Figure 3.2 – S_{in} and S_{out} regions extracted from the base mask (in solid, thick stroke) using an erosion of d_{in} and a dilation of d_{out}.

of squared intensity differences (SSD) between the base mask and the manual segmentation:

$$E_{SSD} = \frac{1}{|S_{in} \cup S_{out}|} \sum_{\mathbf{p} \in S_{in} \cup S_{out}} \|M_i(\Phi(\mathbf{p})) - M_0(\mathbf{p})\|^2 \tag{3.1}$$

where $\mathbf{p} = (x, y)$, $M_0 : \Omega \subset \mathbb{R}^2 \to [0, 1]$ denotes the intensity of base segmentation mask and $M_i : \Omega \subset \mathbb{R}^2 \to [0, 1]$ the manual segmentation mask, both defined over the image space Ω.

The S_{in} and S_{out} regions correspond to bands around the contour constructed from the dilation and erosion of the base mask M_0:

$$S_{in} = M_0 - \text{Erosion}(M_0, d_{in}) \tag{3.2}$$
$$S_{out} = \text{Dilation}(M_0, d_{out}) - M_0 \tag{3.3}$$

where d_{in} and d_{out} are adjustable dilation parameters corresponding to the width of the in and out bands (see Figure 3.2). This constitutes a difference with respect to the original CAC formulation, where the S_{in} region corresponds to the base mask itself, i.e. $S_{in} = M_0$. The contour C is extracted from the mask using the Canny method [38]. After extraction, the relative position of all points belonging to these three regions with respect to the cage control points $\mathbf{v}_j$ are calculated. From then on, their position is governed by the movement of the $\mathbf{v}_j$.

Remember that the Cartesian position of a point in the image can be recovered from the position of the control points and the mean value (affine) coordinates with respect to it:

$$\Phi(\mathbf{p}) = \sum_i \varphi_i(\mathbf{p})\, \mathbf{v}_i = (\Phi_x(\mathbf{p}), \Phi_y(\mathbf{p})) \tag{3.4}$$

The derivative of the energy expression (3.1) with respect to the x coordinate

of the vertices of the cage, v_j^x is:

$$\frac{\partial E_{SSD}}{\partial v_j^x} = \frac{\partial E_{SSD}}{\partial \Phi_x(\mathbf{p})} \frac{\partial \Phi_x(\mathbf{p})}{\partial v_j^x} =$$
$$= \frac{2}{|S_{in} \cup S_{out}|} \sum_{\mathbf{p} \in S_{in} \cup S_{out}} \|M_i(\Phi(\mathbf{p})) - M_0(\mathbf{p})\| \left(\frac{\partial M_i(x,y)}{\partial x} \right)_{\Phi(\mathbf{p})} \phi_j(\mathbf{p}) \quad (3.5)$$

and similarly for the y coordinate:

$$\frac{\partial E_{SSD}}{\partial v_j^y} = \frac{\partial E_{SSD}}{\partial \Phi_y(\mathbf{p})} \frac{\partial \Phi_y(\mathbf{p})}{\partial v_j^y} =$$
$$= \frac{2}{|S_{in} \cup S_{out}|} \sum_{\mathbf{p} \in S_{in} \cup S_{out}} \|M_i(\Phi(\mathbf{p})) - M_0(\mathbf{p})\| \left(\frac{\partial M_i(x,y)}{\partial y} \right)_{\Phi(\mathbf{p})} \phi_j(\mathbf{p}) \quad (3.6)$$

The initial position of the set of cage control points $\mathbf{v}_0$ can be specified manually or be automatically determined from the base mask using user-defined criteria. In Section 4 we discuss a suitable choice for our application to brain image segmentation, although the problem of finding the optimal initial positions of the cage control points (optimal as in providing best segmentation results) is out of the scope of this work and has been proposed as future work.

3.1.2 Principal Component Analysis

After finding out the cage deformations that adjust the atlas base mask to the manual segmentations, the next step is to perform the principal component analysis (PCA) of these deformations. The deformation vectors are constituted by the concatenated x and y positions of the control points of every deformed cage for each image. If there are n cage control points these feature vectors are $2n$-dimensional and take the form:

$$\mathbf{c} = (v_1^x, v_1^y, v_2^x, v_2^y, \ldots, v_n^x, v_n^y) \quad (3.7)$$

From the analysis we extract the identified orthogonal[6] principal components $\mathbf{P}_i$, their associated eigenvalues λ_i (which correspond to the variances) the percentage of the total variance they explain and the mean position of the control points $\bar{\mathbf{c}}$ (that produces the mean shape). We take only the first r most important principal components that accumulate e_{var} of the variance and therefore can explain most deformations.

The ASM of the structure is therefore given by:

- The r principal component vectors $\mathbf{P}_i$
- Their r associated eigenvalues λ_i
- The mean positions of the control points $\bar{\mathbf{c}}$.
- The base mask M_0.

[6] As mentioned in [7], the orthogonality of principal components is crucial for the PDM to effectively limit the allowable shapes to possible ones (see Figure 2.10).

- The initial positions of control points $\mathbf{c}_0$.

It is important to understand that each of these principal components can involve multiple control points and coordinates at the same time, what provides significative expression power. Imagine we were modelling a circle and the possible deformations were scale variations to make the circle smaller or bigger. If the cage forms a square around the circle, a single principal component would suffice to model the deformations. Rigid translations can also be modelled with a single principal component (see Figure 2.9 for an example with resistor shapes). However, due to their linearity, the principal components cannot express rotations.

3.2 Segmentation of new images

We now have a model that encodes the possible deformations from the mean shape to generate the different shapes of the considered structure. This model can be used in combination with any image energy, including edge- and region-based or an unification of both (see Section 2.1), to obtain a segmentation of any new input image. The iterative optimization algorithm to apply to the image is summarized in Algorithm 4.

We only need to know two things: how to calculate the energy and its gradients in terms of the ASM parameters b_i and how to limit the values of b_i to belong to produce only allowed shapes.

3.2.1 Energy and gradients in terms of the ASM parameters

When using the ASM, the cage control points cannot move independently: their movement is controlled by the ASM parameters b_i, and therefore the optimization has to be formulated in terms of b_i. But thanks to the linearity of the ASM it is straightforward to compute the energies and their gradients with respect to b_i.

Given $\mathbf{b}$, the positions of the control cage points $\mathbf{c}$ can be calculated using Eq. (2.24):

$$\mathbf{c} = \bar{\mathbf{c}} + P\mathbf{b} \tag{3.8}$$

where $\mathbf{c}$ is the vector containing the x and y coordinates of all control points as defined in Eq. (3.7). From the position of the control points, we can derive the location of the points belonging to C, S_{in} and S_{out} and, from there, the image energies.

Conversely, we can calculate the value of $\mathbf{b}$ that corresponds to a certain known position of the control points:

$$\mathbf{b} = P^T(\mathbf{c} - \bar{\mathbf{c}}) \tag{3.9}$$

We make use of the chain rule to calculate the derivatives with respect to b_i:

$$\frac{\partial E_{image}}{\partial b_j} = \sum_{i=1}^{2n} \frac{\partial E_{image}}{\partial c_i} \frac{\partial c_i}{\partial b_j} = \sum_{i=1}^{2n} \frac{\partial E_{image}}{\partial c_i} P_j^i \tag{3.10}$$

where P_j^i denotes the value at i-th row and j-th column of the principal components matrix. The previous equation can be expressed in a more compact form:

$$\nabla_{\mathbf{b}} E_{image} = P^T(\nabla_{\mathbf{c}} E_{image}) \tag{3.11}$$

Algorithm 4 Segmentation of new images using CAC and ASM

Input:

- ASM of the structure to segment, including $\mathbf{P}_i$, λ_i, $\bar{\mathbf{c}}$, $\mathbf{c}_0$ and M_0.
- Contour C, inner region S_{in} and outer region S_{out} extracted from the base mask of the ASM M_0.
- Image I containing the structure to segment.
- Image energy functionals to use, including any hyper-parameters needed.
- Parameters m and s of E_{ASM}.

Output:

- Segmentation mask indicating which pixels belong to the structure in the given image M_I.

Procedure:

1. Set the initial ASM parameters as $b_i = 0 \; \forall k$. This gives us the mean shape.
2. While the termination condition is not fulfilled:
   (a) Calculate the position of the control points $\mathbf{c}$ from the values of $\mathbf{b}$ using Eq. (3.8).
   (b) Find the current position in the image of the points of C, S_{in} and S_{out}, that are parametrized with respect to $\mathbf{c}_i$, using Eq. (2.15).
   (c) Compute the image energy for the current configuration.
   (d) Calculate the image energy gradients in terms of the parameters $\mathbf{b}$ using Eq. (3.11).
   (e) Compute the ASM internal energy E_{ASM} and its derivatives using Eqs. (3.12) and (3.13) and add them to the total energy and its gradient.
   (f) Use the gradient descent algorithm to decide the next value of $\mathbf{b}$, that lowers the total energy, if it exists.
3. Deform the base mask M_0 using the cage control points deformation from $\mathbf{c}_0$ to the last value of $\mathbf{c}$, following the steps described in Section (3.2.3). The deformed M_0 gives us the required segmentation.

Therefore the energy gradients can be computed with respect to the movement of the cage control points and then apply Eqs. (3.8) and (3.11) to find the gradient with respect to the principal components parameters $\mathbf{b}$.

3.2.2 Limiting the possible shapes

For the model to generate only allowed shapes we place limits over the values of the linear parameters b_i. In [7] a range within three standard deviations is proposed:

$$-3\sqrt{\lambda_i} \leq b_i \leq 3\sqrt{\lambda_i}$$

To enforce similar limits during active contour segmentation, we use the internal energy:

$$E_{ASM} = \sum_i \left(\frac{b_i}{s\sqrt{\lambda_i}} \right)^{2m} \tag{3.12}$$

with derivative

$$\frac{\partial E_{ASM}}{\partial b_j} = \frac{2m}{s\sqrt{\lambda_j}} \left(\frac{b_j}{s\sqrt{\lambda_j}} \right)^{2m-1} \tag{3.13}$$

Values of b_i within s standard deviations will produce energies smaller than the unity, while values of b_i out of this range will give energies larger than one. The values of s and m can be tuned to specify the "variation width" and the importance of these limits: the larger the value of m, the harder the limits will be (see Figure 3.3).

The internal energy of Eq. (3.12) is included in the energy functional to be minimized:

$$E = E_{image} + E_{ASM} \tag{3.14}$$

3.2.3 Deforming the base mask

Once the optimization is finished and we have the final position of the control points $\mathbf{v}_f$, we generate deformed masks for each image using Eq. (2.20) to find out which pixels of the given image belong to the structure. This corresponds to step 3 of Algorithm 4.

Let $\mathbf{v}_0$ denote the position of the cage control points prior to optimization, $\mathbf{v}_f$ their final position after optimization on a certain image and M_0 the base mask. The steps to find the deformed mask M_I are:

1. Use Eq. (2.18) to calculate the mean value coordinates with respect to the final cage control points $\mathbf{v}_f$ of all points $\mathbf{p}$ of the mask M_I.[7] .

2. Compute the position $\mathbf{x}$ on the base mask M_0 that corresponds to the previous points $\mathbf{p}$ using Eq. (2.20).

[7]Since this calculation is quite time consuming, in our application we only consider the region inside the minimal rectangle that covers all the control points of the final cage. We are therefore making the assumption of that all non-zero pixels of the mask will lay inside this rectangle.

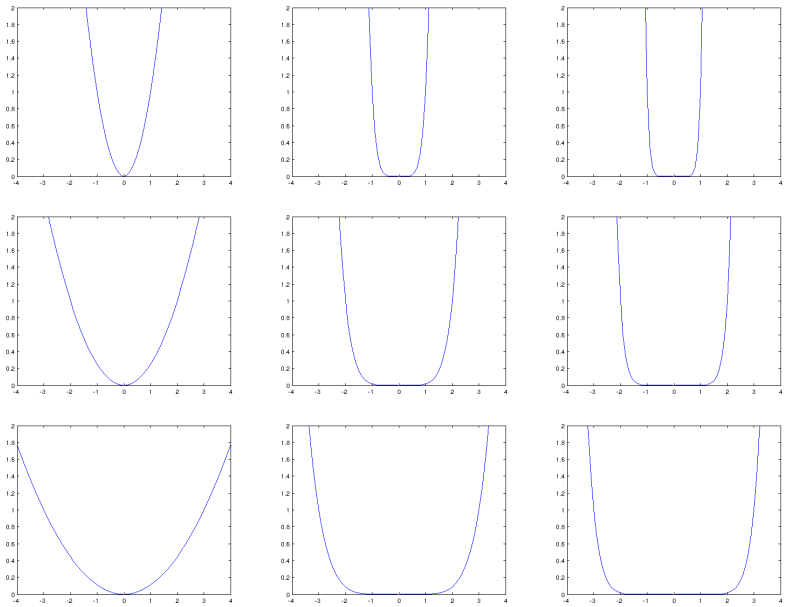


Figure 3.3 – $E_{ASM}(b)$ for different values of s and m, with $\lambda = 1$. From top to bottom: $s = 1, 2, 3$. From left to right: $m = 1, 3, 5$. Note how higher values of s enlarge the range of movement, while increasing m makes the barrier $b = s$ stronger.

3. Set the intensity of the point $\mathbf{p}$ in the deformed mask M_I to the intensity of the point $\mathbf{x}$ in the base mask M_0, i.e. set $M_I(\mathbf{p}) = M_0(\mathbf{x})$. Since $\mathbf{x}$ will generally not lay exactly over a single pixel of the base mask, an interpolation mechanism is used.

Due to the interpolation, some pixels of the deformed masks might have a gray-level value that is neither 0 nor 1 but in between. Therefore a threshold (for example at 0.5) is applied to decide which pixels belong to the structure.

4 Application to brain image segmentation

We now describe how do we apply SCAC to the segmentation of the right and left caudate nuclei in magnetic resonance images. The application has been implemented using Matlab 2011 and uses the library SPM 8[8]. The SPM 8 library is a toolbox for Matlab that uses the concepts of Statistical Parametric Mapping [39] to implement a series of routines to register, segment, normalize and analyze brain imaging data. It was designed to analyze fMRI data, but can be used out-of-the-box to work with MRI data.

Let us assume that we have the MR grayscale brain image volumes of a set of subjects and their corresponding manually segmented left and right caudate nuclei as a binary mask volumes, as well as a suitable atlas with segmentation labels for the caudate nuclei, being these labellings also described as binary mask

[8]Available at http://www.fil.ion.ucl.ac.uk/spm/.

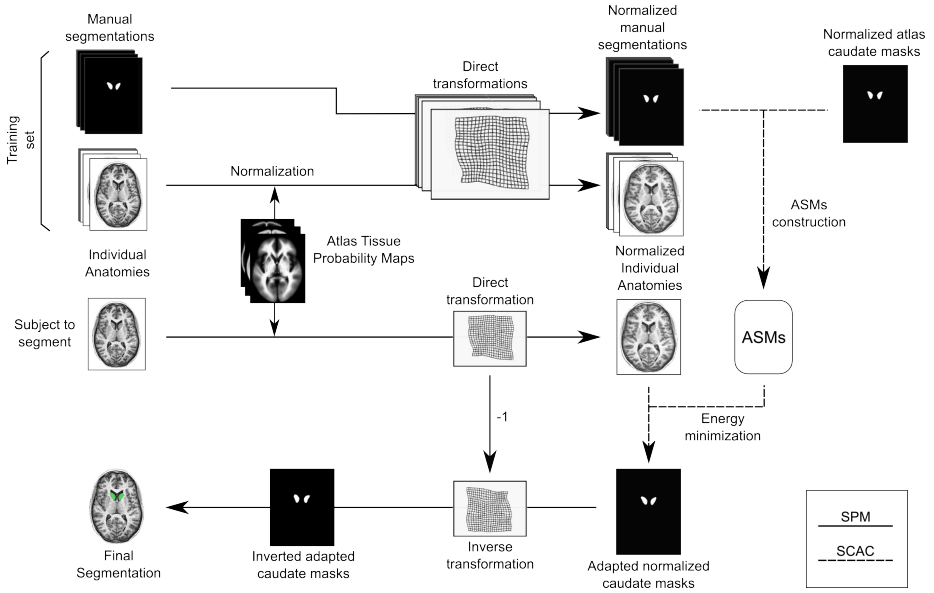


Figure 4.1 – Overview of the application of the SCAC approach to the segmentation of the caudate nuclei in brain magnetic resonance images. The solid lines correspond to the operations performed using SPM and the dashed ones to the SCAC process itself.

volumes. Let us further assume that we have an unsegmented brain MR image volume from another subject, where we want to localize the cited structures.

The approach is illustrated in Figure 4.1 and consists on the following steps:

1. Register the MR brain images of the manually segmented subjects as well as of the unsegmented one with the atlas. Apply the same transformation to the manual segmentations of the caudate nuclei of the segmented subjects, bringing all them to a normalized space.

2. Construct an Active Shape Model for the shape variations of the caudate nuclei on each slice, using the process described in Section 3.1 over the manual segmentations of the structures in the normalized space, using an anatomical atlas mask as base mask M_0. There will be a different ASM for each slice and structure.

3. Perform the segmentation of the caudate nuclei structures in the normalized space, using the procedure described in Section 3.2 over the slices of the MR brain image volume of the unsegmented subject.

4. Apply the inverse registration transformation to the segmented volume found in the previous step to obtain the final segmentation in the original space.

These steps are explained in detail in the next sections.

Cage Active Contours is a technique originally designed to work with 2D images[9]. To be able to work with 3D brain structures, we simply extract

[9]There is a recent extension of Cage Active Contours to 3D in [40].

transversal image slices from the image volumes, perform all operations over the slices and construct back the volumes using the information from the slices.

4.1 Atlas-based registration

The first step consists on bringing the image volumes of all the subjects to the same coordinate space so we can statistically analyze the deviations in contour shape from the template later. From a Machine Learning perspective, what we are doing is ensuring that all instances share the same features (which are coordinates in this case) so we can learn patterns from them.

To do so, we warp the image volumes of both the segmented subjects and the unsegmented one to match the atlas. This process is known as image registration and is described in Section 2.5.2.

SPM 8 includes a function to perform image registration using a combined model [41]. This function is based in a probabilistic framework that combines tissue classification, bias correction and image registration into a single generative model based on Gaussian mixtures to match the white matter, gray matter and cerebro-spinal fluid of the subjects volumes being processed to defined probabilistic templates. Since we have no specific needs, we use the default parameters of SPM for the integrated tissue classification and normalization process.

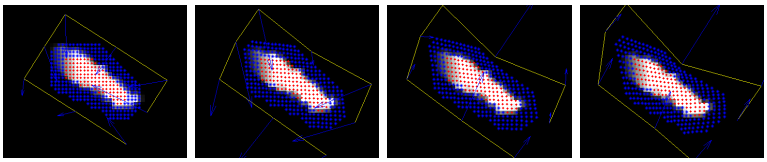
Apart from the normalized image volumes, the function produces by default, among other things, the parametric specification of the spatial normalization transformations for every subject (direct and inverse). These transformations can be used later to transform from the original space to the normalized one (direct) and from the normalized space to the original one (inverse). We use these parametrized transformations to bring all the manual segmentations to the normalized space of the atlas.

This pre-processing registration step can be seen as equivalent to the rotation and scaling step to minimize the sum of squared distances between landmark points in ASM [7]. Also, as noted in [30], a prior registration step can help to initialize the segmenting contour closer to the structure of interest. Without a proper initialization our method would probably fail to converge to the correct solutions in most cases, since it relies solely on local cues close to the segmentation region.

It is important to note that, while the original manual segmentations are given as a binary masks (either a voxel does belong to a given structure or it does not), the corresponding masks in the normalized space are no longer binary but gray-level, being more interpretable as probability maps. This is so because the registration transformation is not in general a one-to-one mapping between voxels, but one voxel in the original space can end up laying partially over several voxels and therefore the intensity values must be interpolated (here we use the linear interpolation methods from SPM and Matlab). For a more in-depth discussion about the issues that can arise during registration due to the discrete nature of the medical images, see [36].

4.2 Construction of an ASM for each slice and structure

To construct an ASM encoding the possible shape variations of the left and right caudate nuclei, we apply Algorithm 3 to the normalized manual segmentation



(a) (b) (c) (d)

Figure 4.2 – Evolution of the S_{in} and S_{out} regions while minimizing the SSD energy. The points belonging to S_{in} are shown in red and the ones to S_{out} in blue. The arrows indicate the directions in which the energy would grow. (a) Initialization, (b) after 4 iterations, (c) after 14 iterations and (d) after 28 iterations (convergence).

masks. We create an ASM for each slice and structure where there is at least a small portion of the caudate nuclei in the atlas mask, using the following inputs:

- Set of training masks M_i: manual segmentation mask for that slice and structure.
- Base mask M_0: anatomical atlas mask for that slice and structure.
- Initial configuration of cage control points $\mathbf{c}_0$: automatically determined from the base mask. We use a rectangular cage with 8 control points around the structure contour (see Figure 4.2 for an example).

The S_{in} and S_{out} band regions are extracted from the base mask using $d_{in} = 20$, since we want to consider all the whole interior of the caudate nuclei, which is narrower than 20 pixels and $d_{out} = 5$, because almost all the background of the slice images is black and therefore a narrow out band is enough (see Figure 4.2).

4.3 Segmentation using SCAC

We now follow the procedure described in Section 3.2 with a convex combination of edge and region energies especially designed to take into account the anatomical peculiarities of these caudate nuclei in the slice images. This energy combination constitute an unified energy (see Section 2.1.1):

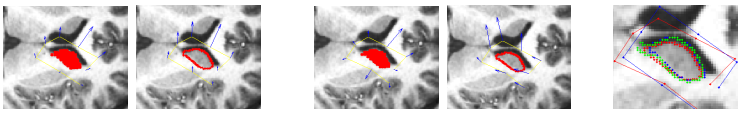
$$E_{image} = \alpha E_{edge} + (1 - \alpha) E_{gauss} \tag{4.1}$$

where α is a parameter between 0 and 1 that allows us to balance the importance of finding sharp edges and a region of uniform intensity. Figure 4.3 shows an example evolution with $\alpha = 0.7$. Again, we only apply the method to the slices where the caudate nucleus is present in the atlas mask image.

The inputs for Algorithm 4 for each structure and slice are:

- ASM: the ones found in the previous section for that slice and structure.
- Image I containing the structure to segment: slice of the MR image of the subject.
- Parameters m and s of E_{ASM}: tuned experimentally (see Section 5).

We now provide a detailed explanation of the energies used.



(a) (b) (c)

Figure 4.3 – Example of SCAC segmentation evolution with $\alpha = 0.7$. (a) Initialization, (b) first iteration and (c) convergence. The red areas of (a) and (b) show the interior region and the contour. (c) shows the contours for manual (green), atlas-based (blue) and SCAC (red) segmentation after convergence.

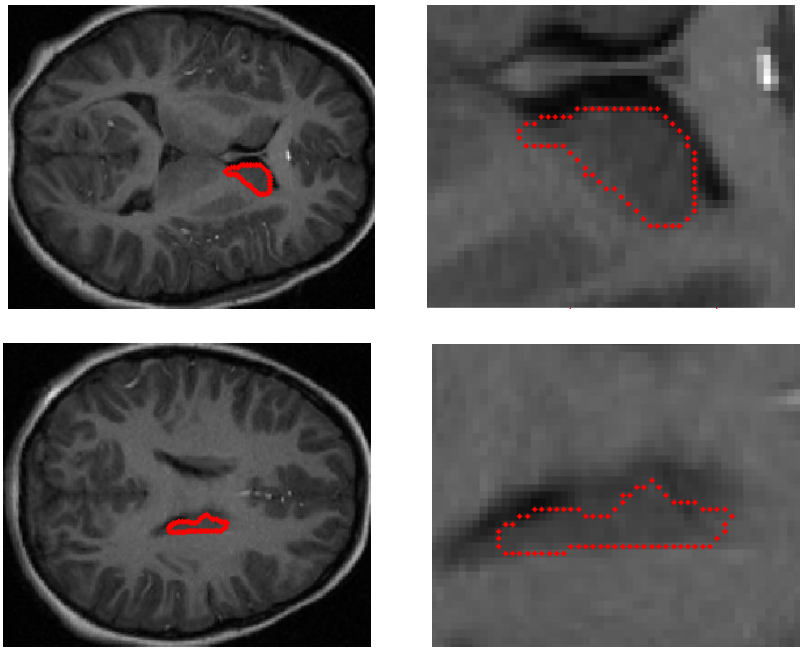


Figure 4.4 – Two examples of MRI slices (left: overview and right: detail). The red contour indicates the manual segmentation of the left caudate nucleus.

4.3.1 Region-based energies

As Figure 4.4 shows, the caudate nuclei are areas of gray matter surrounded by white matter in the exterior part, and a hollow area (of black color) in the interior part. As Figure 4.4 shows, the caudate nuclei are gray matter areas surrounded by two types of tissues: white matter (light gray color in the image) and cerebrospinal fluid (black color in the image). Since the caudate nuclei area exhibits a quite similar and homogeneous gray color, we include a region-based energy that enforces the intensity of this area to be uniform and close to a certain mean.

However, the difference in intensities of the exterior of the caudates inhibits us from including a term in the energy to enforce a similar intensity in the exterior. For this reason, we set $d_{in} = 20$ to include the whole caudate nuclei in the interior region, and $d_{out} = 0$ so there is no exterior segmentation region.

For the interior region we can use the mean-based energy from Eq. (2.8) or

the Gaussian energy from Eq. (2.7). If we estimate the value of σ_{in}^2 using Eq. (2.10), the squared term of Eq. (2.7) vanishes and the Gaussian energy is just the logarithm of the mean energy:

$$E_{gauss} = \log \sigma_{in} = \log E_{mean} \tag{4.2}$$

To use the gradient descent method in the parametric snakes framework we first need to compute the gradients with respect to the control points. The derivative of the mean energy with respect to the movement of the cage control point $\mathbf{v}_j$ is:

$$\begin{aligned}
\frac{\partial E_{mean}}{\partial \mathbf{v}_j} &= \frac{\partial E_{mean}}{\partial \Phi(\mathbf{p})} \frac{\partial \Phi(\mathbf{p})}{\partial \mathbf{v}_j} = \\
&= \frac{2}{|S_{in}|} \sum_{\mathbf{p} \in S_{in}} \left[(I(\Phi(\mathbf{p})) - \mu_{in}) \left(\phi_j(\mathbf{p}) \nabla I(\Phi(\mathbf{p})) - \frac{1}{|S_{in}|} \sum_{\mathbf{p}' \in S_{in}} \nabla I(\Phi(\mathbf{p}')) \phi_j(\mathbf{p}') \right) \right] = \\
&= \frac{2}{|S_{in}|} \sum_{\mathbf{p} \in S_{in}} (I(\Phi(\mathbf{p})) \phi_j(\mathbf{p}) \nabla I(\Phi(\mathbf{p})) - \mu_{in} \phi_j(\mathbf{p}) \nabla I(\Phi(\mathbf{p}))) - \\
&- \frac{2}{|S_{in}|} \left(\mu_{in} \sum_{\mathbf{p}' \in S_{in}} \nabla I(\Phi(\mathbf{p}')) \phi_j(\mathbf{p}') - \mu_{in} \sum_{\mathbf{p}' \in S_{in}} \nabla I(\Phi(\mathbf{p}')) \phi_j(\mathbf{p}') \right) = \\
&= \frac{2}{|S_{in}|} \sum_{\mathbf{p} \in S_{in}} (I(\Phi(\mathbf{p})) \phi_j(\mathbf{p}) \nabla I(\Phi(\mathbf{p})) - \mu_{in} \phi_j(\mathbf{p}) \nabla I(\Phi(\mathbf{p}))) = \\
&= \frac{2}{|S_{in}|} \sum_{\mathbf{p} \in S_{in}} (I(\Phi(\mathbf{p})) - \mu_{in}) \phi_j(\mathbf{p}) \nabla I(\Phi(\mathbf{p}))
\end{aligned} \tag{4.3}$$

where μ_{in} is defined at Eq. (2.9). The derivatives of the Gaussian energy take a very similar form:

$$\begin{aligned}
\frac{\partial E_{gauss}}{\partial \mathbf{v}_j} = \frac{\partial E_{gauss}}{\partial \Phi(\mathbf{p})} \frac{\partial \Phi(\mathbf{p})}{\partial \mathbf{v}_j} &= \frac{1}{2\sigma_{in}^2} \frac{\partial E_{mean}}{\partial \mathbf{v}_j} = \\
&= \frac{1}{\sigma_{in}^2} \frac{1}{|S_{in}|} \sum_{\mathbf{p} \in S_{in}} (I(\Phi(\mathbf{p})) - \mu_{in}) \phi_j(\mathbf{p}) \nabla I(\Phi(\mathbf{p}))
\end{aligned} \tag{4.4}$$

As we can see both energy functionals are essentially the same, with the only difference of that the gradient of the Gaussian energy functional is normalized by the intensity variance. For this reason, we make use of the Gaussian energy in our experiments and disregard the mean-based one.

To make the method more robust, we can take advantage of the fact that the caudate nuclei always show a similar gray-level intensity and specify the mean value μ_{in} manually, which ranges from 1 (white) to 0 (black). In our tests we use $\mu_{in} = 0.65$. To normalize the intensity of the caudate nuclei over all slices we apply an intensity equalization preprocessing to the slice images prior to segmentation.

4.3.2 Edge energy

If region energies were used alone, the parametric snakes would evolve trying to make the segmentation region smaller, since this way it is easier for all the pixels inside to have a similar intensity. That is, the region energies alone do not characterize the areas we want to segment, since a minimum in these energies do not necessarily correspond to a good segmentation.

To solve this issue, we include an edge-based energy, based on Eq. (2.4), to the previous region-based one:

$$E_{edge} = -\frac{1}{|S_{in}|}\sum_{\mathbf{p}\in C} ||\nabla I(\Phi(\mathbf{p}))||^2 \tag{4.5}$$

where C is the contour of the segmentation region. This contour C is obtained using the Matlab implementation of the Canny method [38].

By using the edge-based energy, the contour is driven towards the regions of the image with a larger variation in the intensity, that often correspond to the boundary between brain structures. In our case, the boundary between the caudate nuclei and the hollow area closer to the center of the brain is reasonably clear and can therefore can detected with that gradient-based edge energy, but the frontier with the white matter is not as clear (see Figure 4.4).

The derivative of the previous equation with respect to the x coordinate of the cage control point $\mathbf{v}_j$ takes the form:

$$\begin{aligned}\frac{\partial E_{edge}}{\partial v_j^x} &= -\frac{1}{|S_{in}|}\sum_{\mathbf{p}\in C}\frac{\left(\partial I(\Phi(\mathbf{p}))/\partial x\right)^2 + \left(\partial I(\Phi(\mathbf{p}))/\partial y\right)^2}{\partial v_j^x} = \\ &= -\frac{2}{|S_{in}|}\sum_{\mathbf{p}\in C}\left(\frac{\partial I(\Phi(\mathbf{p}))}{\partial x}\right)\left(\frac{\partial^2 I(\Phi(\mathbf{p}))}{\partial x^2}\right)\phi_j(\mathbf{p}) + \\ &\quad + \left(\frac{\partial I(\Phi(\mathbf{p}))}{\partial y}\right)\left(\frac{\partial^2 I(\Phi(\mathbf{p}))}{\partial x\,\partial y}\right)\phi_j(\mathbf{p})\end{aligned} \tag{4.6}$$

and similarly for y:

$$\begin{aligned}\frac{\partial E_{edge}}{\partial v_j^y} &= -\frac{2}{|S_{in}|}\sum_{\mathbf{p}\in C}\left(\frac{\partial I(\Phi(\mathbf{p}))}{\partial x}\right)\left(\frac{\partial^2 I(\Phi(\mathbf{p}))}{\partial y\,\partial x}\right)\phi_j(\mathbf{p}) + \\ &\quad + \left(\frac{\partial I(\Phi(\mathbf{p}))}{\partial y}\right)\left(\frac{\partial^2 I(\Phi(\mathbf{p}))}{\partial y^2}\right)\phi_j(\mathbf{p})\end{aligned} \tag{4.7}$$

4.4 Inverse mapping

To be able to evaluate the quality of the SCAC segmentation we warp the resulting mask back to the original space. To do so, we make use of the parametrized transformation registration found in the first step of the process (see Section 4.1).

As happens with the first transformation step, it is important to note the differences in the resulting masks in the original space. On one hand, the manual segmentation masks are created manually labelling the structures at each slice

in a binary way: a voxel either belongs to a structure or it does not. On the other hand, the anatomical atlas and the SCAC masks are not binary but have 255 levels of gray[10], representing the probability of a pixel/voxel to be part of the selected structure.

4.5 Technical aspects

An important component of the work done in the context of the Master Thesis was to write the Matlab code to perform all the operations described in Sections 3 and 4. The code of Smart Cage Active Contours is a completely new version based on the code from [8], although a some stubs where taken from the work from [40] and the functions used for atlas-guided registration were taken from the code from [42].

Since this document is not intended as a complete guide to modify and extend the SCAC code, the code listings will not be explained here (although they are heavily commented and hopefully easy to follow along). However, in the next subsections we comment on certain features of our implementation.

4.5.1 Platform and language

Although the code for the original Cage Active Contours framework was written in C using the OpenCV and MegaWave libraries, we decided to switch to Matlab for several reasons:

- Unlike C, Matlab uses an interpreted language, which allows faster prototyping and testing.
- Basic operations like fast matrix multiplication, vector manipulation, plotting, etc. are easily accessible within Matlab.
- Matlab includes an image processing toolbox with useful functions similar to those of OpenCV, like edge detection, image erosion and dilation, filtering, etc.
- Scalability constraints were not severe. Most segmentation processes run in less than a second, being the calculation of the mean value coordinates one of the main bottlenecks.
- SPM runs on Matlab, therefore it is much easier to inter-operate with that library if our code is also written for Matlab.

The code was tested to work on Matlab 2011 running on a Linux platform.

4.5.2 Modularity

Although the code was developed with the application to brain image segmentation in mind, there has been an effort to keep it modular and reusable. To this end, the code included with this work has been split in two parts:

[10] In all cases, the image intensities are considered to be normalized and take values between 0 and 1, not between 0 and 255.

- Smart Cage Active Contours. The method that combines Cage Active Contours with Active Shape Models, which is independent of the medical imaging application and can therefore be used to segment other kind of images.
- Brain image segmentation using SCAC. Uses SCAC as a library and takes care of manipulating the image volumes, transforming them to the normalized space and back, and evaluating the quality measures, among other MRI manipulation related tasks.

The code for brain image segmentation can be found in the `Code` folder of the project, while the SCAC library is placed into the `SCAC` sub-folder.

4.5.3 Configuration parameters

To improve re-usability and understandability, all configuration parameters that are expected to be specified by the user are placed in two files:

- The first file, named `brain_parameters.m`, is in the root of the project code. It contains the parameters related to the application of the SCAC method to segment the caudate nuclei in magnetic resonance images:
  - Numerical ids of the subjects to consider. We can use the whole database (40 subjects) or select a subset of them for all the process.
  - Slices of the normalized space to work on using SCAC. The rest of them will be considered mask background not containing any part of the structure to segment.
  - Accumulated variance limit for PCA e_{var}. The system picks the first principal components until this value of accumulated variance is explained.
  - Parameters s and m of the E_{ASM} energy defined in Eq. (3.12).
  - Number of cross-validation folds $nfolds$ to use for performance validation.
  - Values of d_{in}^{ASM} and d_{out}^{ASM} for the determination of the inside and outside regions when constructing the ASM from the masks.
  - Values of d_{in}^{SCAC} and d_{out}^{SCAC} for the determination of the inside and outside regions when using SCAC.
  - Value of the parameter α of the unified energy of Eq. (2.11). The closer this value is to 1, the more importance the edge energy acquires.
  - Mean value intensity μ_{in} of the interior of the segmentation region. A value of 1 is white, while 0 is black. If it is not specified, it is estimated at each step using Eq. (2.9).
  - Paths to the input, temporary and output folders for the placement of the files used during the process.
- The second file, named `defaultParameters.m`, can be found in the SCAC sub-folder. It controls the parameters specific to the SCAC method and not related to the medical application:

- Algorithm used to minimize the energy functional. Gradient descent and L-BFGS are available. L-BFGS behaved strangely in our tests so we stuck to basic gradient descent.
- Relative energy difference e_{tol} in an step to stop optimization. If $(E(\mathbf{x}_{t+1}) - E(\mathbf{x}_t))/E(\mathbf{x}_t) < e_{tol}$, the algorithm considers that a local minimum has been reached.
- Maximum number of steps $maxIter$ to stop optimization. If no local minimum has been found after $maxIter$ steps, the algorithm stops and returns the last position $\mathbf{x}$ tested.
- Maximum position jump $maxMove$ when using the gradient descent optimization algorithm. When performing line search, consider only displacements that satisfy $max(\mathbf{x}_{t+1} - \mathbf{x}_t) \leq maxMove$. This allows to be conservative and avoid too large jumps that might affect the robustness of the optimization.
- Padding π and complexity χ of the initial set of cage control points. The initial cage will be rectangular, with sides at a minimal distance of π pixels from the contour and $2^{\chi+1}$ control points.

4.5.4 Verbosity and debug modes

To ease research, three levels of verbosity or "debug modes" for the SCAC optimization can be set as a global variable during execution. From most verbose to least verbose:

- **DEBUGMODE = 2**. At each iteration, show the evaluation value of each of the energies and the gradient in the control point positions they induce as vectors on the image.
- **DEBUGMODE = 1**. Show only the segmentation results after the contour evolution finishes, which can be due to convergence or because the maximum number of steps was exceeded. For the brain segmentation application, the atlas-guided segmentation, the manual segmentation and the SCAC segmentation, as well as the initial and final energies of this last method, are also shown for comparison.
- **DEBUGMODE = 0**. No information about the energy values evolution or figures are shown.

For the two most verbose debug modes, the execution pauses after showing the information, having the user to hit any key to continue. Otherwise, the amount of data output to the screen would be too much to be analyzed properly.

4.5.5 Unused implemented features

There are two features implemented in the SCAC code but that are not used in the experiments in this work:

- Multiple region control using a single cage. The user can define several regions (named "areas" in the code) to be controlled by a single cage enclosing both regions. Each region can contribute to the cage control points evolution with a custom energy function, chosen by the user. This

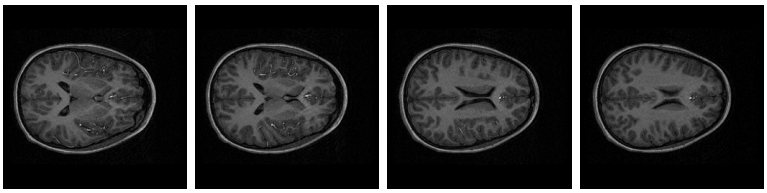


Figure 5.1 – Example images of the transversal slices number 28, 30, 34 and 36 of a random subject from the dataset.

opens the possibility to combining local information from multiple regions to achieve a more informed deformation. Also, it can be used to deform one region using the information from another region.

- Multiple resolution depth. Instead of defining a fixed set of cage control points, start with a cage of n points and evolve until convergence. Then refine the cage placing one extra control point between each two points (resulting in n^2 points at this level) and evolve again. Repeat this as many times as desired.

5 Experiments

Before presenting the results of our experiments, we describe the material and methods of comparison, as well as the measures used to assess the quality of the segmentation and the parameter adjustment and validation protocol followed.

5.1 Material

The magnetic resonance volume images of the subjects were taken from the control set of the URNC Database. This is the database used in [42], which includes 40 control subjects recruited from the community, compiled by the Unit of Child Psychiatry at the Vall d'Hebron Hospital in Barcelona, Spain, and coordinated by the Unit of Research in Cognitive Neuroscience (URNC) at the IMIM Foundation. The mean age of the group was 11.7 ($\sigma = 3$). The resolution of the scans is $256 \times 256 \times 60$ pixels with 2-mm thick slices. Expert segmentations of the 80 individual caudate nuclei (left plus right) was obtained, using the MRIcro software[11] for volume labelling and manipulation.

Figure 5.1 shows four examples of transversal projection slices from the dataset, and Figure 5.2 shows examples of manual segmentations.

The atlas tissue probability maps (corresponding to gray matter, white matter and cerebro-spinal fluid) we used for image registration are the standard ones shipped with SPM, which are a slightly modified version of the ICBM Tissue Probabilistic Atlas[12]. The atlas masks for the left and right caudate in the normalized space were extracted from the Anatomical Automated Labelling (AAL) atlas from [43].

[11] Available at http://www.mccauslandcenter.sc.edu/mricro/.

[12] Available at http://www.loni.ucla.edu/ICBM/ICBM_Probabilistic.html.

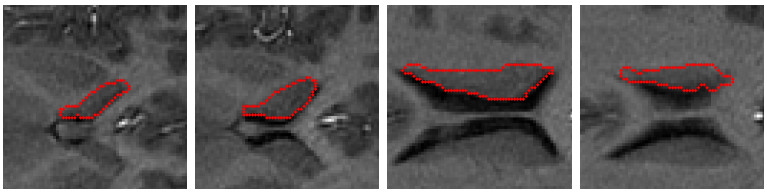


Figure 5.2 – Transversal slices number 28, 30, 34 and 36 showing the manual segmentation of the left caudate nucleus of a random subject from the dataset.

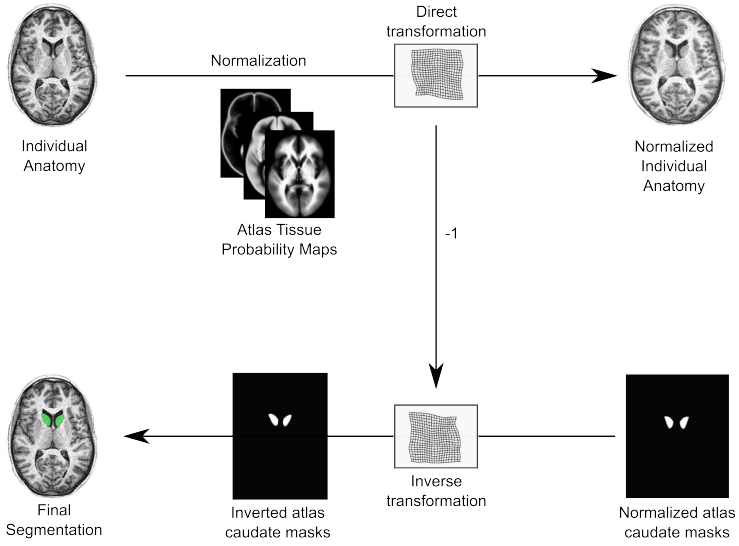


Figure 5.3 – Flowchart of the atlas-guided segmentation approach used in this work. Note that in this case the inverse transformation is applied is directly to the atlas anatomical caudate masks, instead of to their deformed versions resulting from the SCAC energy minimization process.

5.2 Methods

As mentioned in the Introduction, we compared the SCAC method against a current state-of-the-art method: classical atlas-guided segmentation. The SCAC method applied to brain image segmentation is described in Section 4. The atlas-guided segmentation process for a subject is illustrated in Figure 5.3 and consists on the following steps:

1. The brain image of the subject is elastically registered from the original to the normalized space of the atlas using the unified segmentation and normalization process implemented in SPM. From this calculation an inverse deformation transformation can be extracted, that maps the normalized space back onto the original one.

2. The inverse deformation is applied to the atlas masks of the caudate nuclei to obtain their segmentation in the original space.

5.3 Quality quantitative measures

We use two measures to quantitatively assess the quality of the resulting segmentation volume masks, always comparing with the manual segmentation, taken as ground truth:

- Volumetric union overlap (VO). We first convert all images to binary using a probability threshold of 0.5: all voxels with a probability of 0.5 or more to be part of the structure are considered part of this structure (and given a value of 1), and the rest are discarded (with a value 0). Using R to denote the estimated segmentation, G to denote the manual segmentation and $|\cdot|$ to denote the cardinality of a set:

$$\text{VO} = \left| \frac{R \cap G}{R \cup G} \right|$$

  A value of 1 indicates a prefect segmentation, while a value of 0 means no overlap between the estimated segmentation and the manual one.

- Sum of squared intensity differences (SSD). We compare how far are the estimated segmentations from the manual ones, without binarizing:

$$SSD = \frac{1}{|R \cup G|} \sum_{\mathbf{p} \in |R \cup G|} (I_R(\mathbf{p}) - I_G(\mathbf{p}))$$

  where R represents here the voxels of the image volume that are not 0 and $I_{R/G}(\mathbf{p})$ the intensity value of the corresponding image volume at the point $\mathbf{p}$. In this case, a perfect segmentation would be represented by a value of 0, while a totally wrong segmentation correspond to the value 1.

5.4 Parameter adjustment and validation protocol

As is widely known in the Machine Learning community, estimating the accuracy of a model over the same data that was used to create the model (the training data) is prohibited. This is so because when we construct a model we want to encode some patterns but also to have generalization capabilities. The model must be able to work with data that is similar to the training data (similar in the sense that it follows a common pattern) but not necessarily identical.

If we checked the performance of a model over the training data we would be probably obtain "too good" results. To avoid this problem we use a cross-validation strategy and split randomly our subjects dataset in two:

- A learning dataset, with 75% of the subjects (30), used to create and tune the model. We further split this set in 5 to adjust the parameters using 5-fold cross-validation. For each cross-validation iteration, we denominate "training" to the set used to construct the ASM (see Section 3.1) and "validation" to the set used to perform the SCAC segmentation and quality assessment (see Sections 3.2 and 5.3).

- A test (also called hold-out) dataset with the resting 25% of them (10 subjects), used to check the accuracy of the final "winner" model. After

constructing the ASM with the whole learning dataset, we perform segmentation over the subjects of the test dataset and evaluate the quality of the results.

The protocol followed is described in Algorithm 5. For all experiments the parameters were set to $m = 5$ (to strongly restrict the allowable ranges of the ASM parameters b_i, see Figure 3.3) $d_{in}^{ASM} = d_{in}^{SCAC} = 20$, $d_{out}^{ASM} = 5$, $d_{out}^{SCAC} = 0$ (see Sections 4.2 and4.3), accumulated variance $e_{Var} = 0.95$ (because it seems enough to encode the main variations), slices from 22 to 39 (the ones containing a portion of the caudate nucli in the anatomical atlas), $nfolds = 5$, padding $\pi = 5$ and cage complexity $\chi = 2$ (8 control points). These last two parameters were assigned rather arbitrarily, and the development of a methodology to optimize them, and the position of the control points in general, has been proposed as futured work.

Algorithm 5 Parameter adjustment and validation protocol

Let TR(i) and VA(i) be the training and validation set of subjects, respectively, of the i-th cross-validation fold, and TE the test set of subjects.

1. For every combination of parameters:
   (a) For every fold i:
      i. Construct the ASMs using TR(i).
         A. Perform SCAC and atlas-guided segmentation over VA(i).
         B. Evaluate the quality measures over VA(i).
   (b) Compute the mean of the quality measures over all the folds.
2. Take the combination of parameters that provides better results over VA(i).
3. Construct an ASM using all subjects from $\bigcup_i \text{TR}(i) \cup \text{VA}(i)$.
4. Perform the SCAC and atlas-guided segmentation over TE.
5. Evaluate the quality of this segmentation over TE.

The region-based energy used was the Gaussian one. The CAC optimizations were performed using the gradient descent method until the relative energy variation is less than 0.1% (what means that we reached a local minimum, or a maximum number of 150 iteration steps is exceeded).

The following parameters were tuned using cross-validation:

- Standard deviations allowed $s = 1, 2, 3$ (cf. Eq. (3.12)).
- Relative weight between edge and region-based energy in the unified energy $\alpha = 0.3, 0.4, 0.5, 0.6, 0.7$ (cf. Eq. (4.1)).
- Whether to fix the mean intensity at the interior of the segmentation region $\mu_{in} = 0.65$ or estimate it at each step. The value of $\mu_{in} = 0.65$ was estimated empirically by analyzing the intensity levels of the caudate nuclei regions in the MR brain images of some of the subjects.

See Section 4.5.3 for a description of all the mentioned parameters.

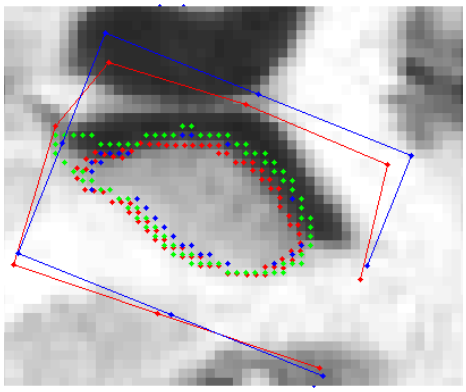


Figure 5.4 – Example segmentation of the caudate nuclei. The manual segmentation is marked in green, the blue corresponds to the atlas mask and the red one is the result of the SCAC segmentation with $\alpha = 1$ and $s = 1$. Note the lack of contrast between the caudate nucleus and the white area under it.

5.5 Results and Discussion

Table 1 shows the results for the cross-validation tuning of the parameters s and α. As can be observed, larger values of s provide a significantly poorer segmentation results for all values of α. That is, giving more freedom to the CAC model to deform in ways less typical in the training set was counter-productive. A possible explanation for that result is that the mask initialization to the mean shape in the normalized frame of reference constitutes already a good approximation to many cases and most individual shapes resemble those of the training set.

A perhaps more surprisingly result is that the unified energy that provides best results is the one that only uses the edge-based energy ($\alpha = 1$), completely ignoring the region-based Gaussian one. This means that trying to minimize the intensity difference between the pixels inside the segmentation region did not help segmenting the caudate nuclei. We believe there are two possible reasons for this. First, if the mask initialization to the mean shape is already close enough to the solution, it is more important to make the contour follow strong edges. As explained in Section 2.1.1, region-based energies have a larger basin of attraction, but fail at localizing explicit edges precisely when there is a lack of contrast (cf. Figure 2.2 and Figure 5.4). And second, the manual segmentations of the structures often include a small band of the darker pixels next to the caudate nuclei in the interior of the segmentation region. The Gaussian energy penalizes the inclusion of this dark region, being more conservative and resulting in a smaller overlap (cf. Figure 5.5).

The cross-validation results obtained for atlas-guided segmentation (in all cases, since they are not influenced by the tuning of parameters) are $VO_{AG} = 0.6494$ and $SSD_{AG} = 0.1907$, which are clearly worse than those obtained using the SCAC methodology and best combination of parameters.

Table 2 shows the quality results over the test set for both the atlas-guided segmentation and SCAC with $\alpha = 1$ $s = 1$ and $\mu_{in} = 0.65$, together with the standard deviations of the results over the segmentation of the 20 caudate nuclei (10 subjects, left and right caudate). A t-test over these values results in the

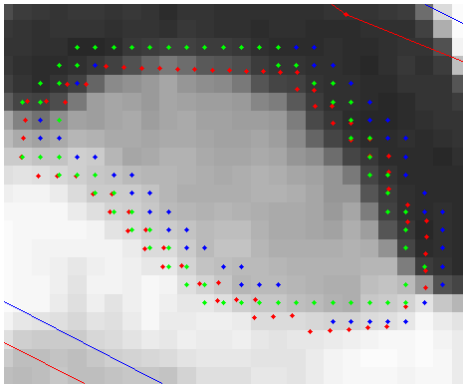


Figure 5.5 – Detail of a segmentation example. The green points indicate the manual segmentation contour. Note how considering this green contour includes a band of dark pixels inside the segmentation region.

		$\mu_{in} = 0.65$	$\mu_{in} = 0.65$	$\mu_{in} =$ (2.9)	$\mu_{in} =$ (2.9)
s	α	VO_{SCAC}	SSD_{SCAC}	VO_{SCAC}	SSD_{SCAC}
1	0.3	0.6799	0.1330	0.6760	0.1361
1	0.4	0.6826	0.1312	0.6788	0.1343
1	0.5	0.6859	0.1295	0.6810	0.1325
1	0.6	0.6886	0.1277	0.6843	0.1302
1	0.7	0.6909	0.1261	0.6879	0.1280
1	0.8	0.6925	0.1248	0.6906	0.1260
1	0.9	0.6934	0.1238	0.6924	0.1243
1	**1**	**0.6934**	**0.1232**	0.6934	0.1232
2	0.3	0.6342	0.1627	0.6167	0.1753
2	0.4	0.6421	0.1570	0.6250	0.1695
2	0.5	0.6523	0.1507	0.6362	0.1622
2	0.6	0.6637	0.1435	0.6497	0.1530
2	0.7	0.6747	0.1365	0.6641	0.1435
2	0.8	0.6721	0.1315	0.6771	0.1347
2	0.9	0.6879	0.1267	0.6853	0.1286
2	1	0.6869	0.1259	0.6869	0.1259
3	0.3	0.5690	0.2083	0.5318	0.2409
3	0.4	0.5824	0.1993	0.5476	0.2281
3	0.5	0.6012	0.1845	0.5701	0.2092
3	0.6	0.6252	0.1678	0.5980	0.1875
3	0.7	0.6471	0.1529	0.6297	0.1647
3	0.8	0.6651	0.1403	0.6588	0.1445
3	0.9	0.6735	0.1330	0.6689	0.1357
3	1	0.6721	0.1315	0.6721	0.1315

Table 1 – Mean VO and SSD quality measures for 5-fold cross-validation. Best results are marked in bold.

Method	VO	SSD
Atlas-guided	0.6335 ± 0.0799	0.1975 ± 0.0632
SCAC	0.6975 ± 0.0475	0.1186 ± 0.0267
t-test (99%)	SCAC > AG	SCAC < AG

Table 2 – Quantitative quality measures for atlas-guided and SCAC segmentations over the test set (with 10 subjects).

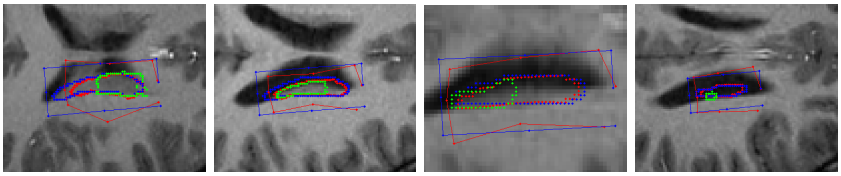


Figure 5.6 – Example segmentation contours of the last slices of the left caudate nucleus (from 36 to 39) in the normalized space. The manual segmentation is shown in green, the atlas-guided one in blue and the SCAC one in red.

acceptance of the hypothesis of that the SCAC segmentation quality measures are significantly better than those of the atlas-guided segmentation, with a 99% of confidence. Therefore, we can safely affirm that the SCAC segmentation method provides results closer to the manual segmentation than those from the atlas-guided method.

Looking into the results more in detail, we can find out that the slices where SCAC segmentation provides worse results correspond to the last ones, at the tail of the caudate nuclei. Figure 5.6 shows some examples of these wrong segmentations. As can be observed, all them are cases where there are very few visual cues to drive the algorithm, like poor contrast or the absence of reasonably strong edges around the whole structure. Moreover, the tail and head are the areas of the caudate nuclei more difficult to capture in an atlas, due to the variability of manual segmentations on them. Thus, atlas contains not correct information about the caudate tail and the error is propagated to the result of the atlas-guided method and the initialization of the SCAC.

This is probably an intrinsic limitation of the method coming from the fact of that it works with transversal slices. When working with tridimensional images, it might be necessary to take into account multiple complementary points of view to determine the limits of an structure. Therefore, to overcome this limitation it would be interesting to extend SCAC to 3D structures.

We can see that the segmentation provided by the atlas mask for this slices is also bad in most cases. Since SCAC uses only local information, it is highly dependent on a good initialization. Therefore, if the mean segmentation shape for a slice[13] is very different from the manual segmentation contour, SCAC segmentation will probably provide poor results.

Finally, obtaining a good initialization in the normalized space largely depends on a correct image registration. A more accurate initialization might be obtained by tweaking the registration parameters or choosing an atlas that matches better

[13]Remember that the mean segmentation shape is just the anatomical atlas mask deformed using the mean position of the control points over the set used to construct the ASM.

the population under study.

6 Conclusions

In this work, we present a new segmentation method, Smart Cage Active Contours (SCAC), that combines a parametric active contours framework, Cage Active Contours (CAC) with Active Shape Models (ASM). The evolution is driven by a reduced set of control points, that modify the shape of the segmentation contour via an affine transformation. The Active Shape Model is constructed using these control points, avoiding the difficult and tedious work of placing the landmarks manually over the training shapes.

We apply our method to the segmentation of the right and left caudate nuclei in the normalized magnetic resonance brain images of a set of 40 subjects. We use 30 of them to tune the parameters of the model and construct an ASM, and test on the rest. The segmentation is initialized to the mean shape and evolved using an edge-based energy functional that looks for strong edges, while deforming the shapes only in ways consistent with the ASM.

The results show that the SCAC method clearly outperforms the segmentation of the same structures using a classical atlas-guided strategy, as evaluated by using two different quantitative quality measures. The worse results for the SCAC method correspond to the tail of the caudate nuclei, an area where the atlas-guided strategy also provided bad quality segmentations, indicating that SCAC depends strongly on a good initialization.

Also, using only edge-based energies provides better results than using only region-based ones, or a convex combination of both. This suggests that, for our particular application, region-based energies do not provide useful information to finely adjust the segmentation contour to the region of interest due to the lack of contrast, being strong edges a much more informative cue.

7 Future work

Our method approaches the segmentation of volumetric structures by working on them slice by slice and combining the results to create a segmented volume. However, the boundaries of these 3D structures are likely to be more easily identifiable with a real 3D segmentation model. The use of a 3D model would allow the evaluation of relevant information like the image intensity gradients perpendicular to the model surface. Moreover, it would ensure the smoothness of the segmentation volume itself by fitting all slices simultaneously. This factor is completely overlooked by our method now, where the slices are treated independently. The extension of Active Shape Models and Active Appearance Models techniques to 3D images is already has already been considered in [26, 27, 28]. Also, the generalization of mean value coordinates to $\mathbb{R}^3$ is presented in [44], and the use of the Cage Active Contours framework to 3D is explored in [40]. Therefore, we think that it should be easy and interesting to extend Smart Cage Active Contours to deal with 3D models.

We have also observed that the power of generic energy functionals, like the ones we used, to segment medical images, is somewhat limited. Very often, the structures on these images do not exhibit strong edges or high contrast variations.

A much more sensible approach would be to construct a combined model of the gray-level (especially at the boundaries) and shape of the structures to segment, i.e. to use Active Appearance Models instead of Active Shape Models in our framework. This is indeed the strategy followed by some of the most promising medical imaging model-based methods [29, 30, 31, 32]. Actually, some of these works represent the deformation of the brain structures using a tetrahedral mesh of control points covering the region of interest and affine interpolation.

Another possible research direction is the simultaneous search of multiple segmentation regions in SCAC using the same set of control points. Instead of constructing a model for just one structure, we can create a composite model involving more than one and adjust that composite model to an input image. This is the approach followed by [30], where the adjustment of the composite model is used to initialize an individual model of the left and right caudate nuclei.

An area that deserves special attention that is not related directly to SCAC but to our application to brain image segmentation is image registration. In this work we use image registration as a way to align all images to be able to construct and use the Active Shape Models, as well as to provide a good contour initialization. The optimality of the registration process therefore affects the quality of the segmentation. Here we just use SPM with the default parameters, but a tuning of these parameters or the selection of the atlas (including multi-atlas approaches) could be used to improve the results.

The choice of the base mask that deforms to adapt to the individual shapes of the structures from the training subjects is also very important. In our application, we use the anatomical atlas of each caudate nuclei as base masks, but we have noticed bad results in the cases where this atlas was very different from the manual segmentation. Therefore, it would be more sensible to derive the base mask from the manual segmentations themselves. A groupwise strategy to register images without the need of manually establishing a landmark correspondence that can be used to find a good-quality base mask is described in [31].

Another interesting research direction would be to experiment with hierarchical approaches, where a set of control points controls another set of control points (extended up to any hierarchy depth). This would allow for an initial coarse segmentation by moving the higher level of control points that could be later refined by adjusting the position of the control points at lower levels. Also, a composite model for a complex structure could be coarsely adjusted, passing the control later to smaller "cages" to obtain a finer segmentation of every substructure.

Last, some work could be done in investigating the best placement of the control points relative to segmentation contour. In our application we consider a rectangular cage with 8 control points around the contour, but some other configurations of cage shapes, number of control points and relative positioning could be explored.

References

[1] D. L. Pham, C. Xu, and J. L. Prince, "Current methods in medical image segmentation," *Annual review of biomedical engineering*, vol. 2, pp. 315–37, Jan. 2000.

[2] Z. Ma, J. a. M. R. S. Tavares, R. N. Jorge, and T. Mascarenhas, "A review of algorithms for medical image segmentation and their applications to the female pelvic cavity.," *Computer methods in biomechanics and biomedical engineering*, vol. 13, pp. 235–46, Jan. 2010.

[3] N. Pal and S. Pal, "A review on image segmentation techniques," *Pattern recognition*, 1993.

[4] R. Haralick and L. Shapiro, "Image segmentation techniques," *Computer vision, graphics, and image processing*, vol. 29, no. 1, pp. 100–132, 1985.

[5] T. McInerney and D. Terzopoulos, "Deformable models in medical image analysis: a survey," *Medical Image Analysis*, vol. 1, pp. 91–108, 1996.

[6] T. Cootes, A. Hill, C. Taylor, and J. Haslam, "Use of active shape models for locating structures in medical images," *Image and Vision Computing*, vol. 12, pp. 355–365, July 1994.

[7] T. Cootes, C. Taylor, D. Cooper, and J. Graham, "Active Shape Models-Their Training and Application," *Computer Vision and Image Understanding*, vol. 61, pp. 38–59, Jan. 1995.

[8] A. Berenguel, "Image Segmentation with Cage Active Contours," 2011.

[9] M. S. Floater, "Mean value coordinates," *Computer Aided Geometric Design*, vol. 20, pp. 19–27, Mar. 2003.

[10] D. L. Collins, A. P. Zijdenbos, W. F. C. Baaré, and A. C. Evans, "Animal+insect: Improved cortical structure segmentation," in *IPMI*, pp. 210–223, Springer, 1999.

[11] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active contour models," *International journal of computer vision*, vol. 331, pp. 321–331, 1988.

[12] M. Jacob, T. Blu, and M. Unser, "Efficient energies and algorithms for parametric snakes," *IEEE transactions on image processing*, vol. 13, pp. 1231–44, Sept. 2004.

[13] T. Schoepflin, "Active contour model with gradient directional information: directional snake," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, no. 2, pp. 252–256, 2001.

[14] T. F. Chan and L. a. Vese, "Active contours without edges," *IEEE transactions on image processing*, vol. 10, pp. 266–77, Jan. 2001.

[15] A. Chakraborty, L. H. Staib, and J. S. Duncan, "Deformable boundary finding in medical images by integrating gradient and region information," *IEEE transactions on medical imaging*, vol. 15, pp. 859–70, Jan. 1996.

[16] M. Rousson and R. Deriche, "A variational framework for active and adaptative segmentation of vector valued images," in *Workshop on Motion and Video Computing, 2002. Proceedings.*, pp. 56–61, IEEE Comput. Soc, 2002.

[17] H. Gouraud, "Continuous shading of curved surfaces," *Computers, IEEE Transactions on*, 1971.

[18] B. Phong, "Illumination for computer generated pictures," *Communications of the ACM*, 1975.

[19] K. G. Kobayashi and K. Ootsubo, "t-FFD," in *Proceedings of the eighth ACM symposium on Solid modeling and applications - SM '03*, (New York, New York, USA), p. 226, ACM Press, 2003.

[20] S. Coquillart, "Extended free-form deformation: a sculpturing tool for 3D geometric modeling," 1990.

[21] M. Floater and C. Gotsman, "How to morph tilings injectively," *Journal of Computational and Applied Mathematics*, vol. 101, no. 1-2, pp. 117–129, 1999.

[22] C. Gotsman and V. Surazhsky, "Guaranteed intersection-free polygon morphing," *Computers & Graphics*, 2001.

[23] K. Hormann and M. S. Floater, "Mean value coordinates for arbitrary planar polygons," *ACM Transactions on Graphics*, vol. 25, pp. 1424–1441, Oct. 2006.

[24] H. Hotelling, "Analysis of a complex of statistical variables into principal components.," *Journal of educational psychology*, 1933.

[25] T. Cootes, G. Edwards, and C. Taylor, "Active appearance models," *Computer Vision - ECCV'98*, vol. 1407, pp. 484–498, 1998.

[26] B. Lelieveldt, "Time continuous segmentation of cardiac MR images using Active Appearance Motion Models," *International Congress Series*, vol. 1230, pp. 961–966, 2001.

[27] A. Hill, A. Thornham, and C. Taylor, "Model-Based Interpretation of 3D Medical Images.," *BMVC*, 1993.

[28] S. C. Mitchell, J. G. Bosch, B. P. F. Lelieveldt, R. J. van der Geest, J. H. C. Reiber, and M. Sonka, "3-D active appearance models: segmentation of cardiac MR and ultrasound images," *IEEE transactions on medical imaging*, vol. 21, no. 9, pp. 1167–78, 2002.

[29] T. Cootes and C. Beeston, "A unified framework for atlas matching using active appearance models," *Information Processing in Medical Imaging*, vol. 1613, pp. 322–333, 1999.

[30] K. Babalola, V. Petrovic, T. Cootes, C. Taylor, C. Twining, T. Williams, and A. Mills, "Automatic segmentation of the caudate nuclei using active appearance models," 2007.

[31] T. Cootes, "Groupwise construction of appearance models using piece-wise affine deformations," *British Machine Vision*, vol. 2, no. Mdl, pp. 879–888, 2005.

[32] K. O. Babalola, T. F. Cootes, C. J. Twining, V. Petrovic, and C. J. Taylor, "3D brain segmentation using active appearance models and local regressors," *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2008*, vol. 5241, pp. 401–408, Jan. 2008.

[33] B. Zitova and J. Flusser, "Image registration methods: a survey," *Image and vision computing*, 2003.

[34] M. Cabezas, A. Oliver, X. Lladó, J. Freixenet, and M. B. Cuadra, "A review of atlas-based segmentation for magnetic resonance brain images," *Computer methods and programs in biomedicine*, vol. 104, pp. e158–77, Dec. 2011.

[35] J. Talairach and P. Tournoux, "Co-planar stereotaxic atlas of the human brain. 3-Dimensional proportional system: an approach to cerebral imaging," 1988.

[36] D. Hill and P. Batchelor, "Medical image registration," *Physics in medicine and Biology*, 2001.

[37] J. Maintz and M. Viergever, "A survey of medical image registration," *Medical image analysis*, 1998.

[38] J. Canny, "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, pp. 679–698, 1986.

[39] K. Friston, J. Ashburner, and S. Kiebel, *Statistical Parametric Mapping: The Analysis of Functional Brain Images: The Analysis of Functional Brain Images*. 2003.

[40] Q. Xue, *Three dimensional parametric active contours*. Erasmus mundus master thesis, UB, 2012.

[41] J. Ashburner and K. Friston, "Unified segmentation," *Neuroimage*, 2005.

[42] L. Igual, J. C. Soliva, A. Hernández-Vela, S. Escalera, X. Jiménez, O. Vilarroya, and P. Radeva, "A fully-automatic caudate nucleus segmentation of brain MRI: application in volumetric analysis of pediatric attention-deficit/hyperactivity disorder.," *Biomedical engineering online*, vol. 10, p. 105, Jan. 2011.

[43] N. Tzourio-Mazoyer, B. Landeau, D. Papathanassiou, F. Crivello, O. Etard, N. Delcroix, B. Mazoyer, and M. Joliot, "Automated anatomical labeling of activations in SPM using a macroscopic anatomical parcellation of the MNI MRI single-subject brain.," *NeuroImage*, vol. 15, pp. 273–289, 2002.

[44] M. S. Floater, G. Kós, and M. Reimers, "Mean value coordinates in 3D," *Computer Aided Geometric Design*, vol. 22, pp. 623–631, Oct. 2005.