Shells and Spheres: A Framework for Variable Scale Statistical Image Analysis

C. Aaron Cois Ken Rockot John Galeotti
Robert Tamburo George Stetten

CMU-RI-TR-04-19

April 2006

Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

Abstract

We have developed a framework for analyzing images, called *Shells and Spheres*, based on a set of spheres with adjustable radii, with exactly one sphere centered at each image pixel. This set of spheres, known as a *sphere map*, is considered optimized when each sphere reaches, but does not cross, the nearest boundary. Calculations denoted as *Variable-Scale Statistics* (VSS) are performed on populations of pixels within spheres, as well as populations of adjacent and overlapping spheres, in order to deduce the proper radius of each sphere. Spheres grow or shrink by adding or deleting an outer shell one pixel thick. Unlike conventional fixed-scale kernels, our spherical operators consider as many pixels as possible to differentiate between objects and accurately delineate boundaries. We use the word "sphere" here for brevity, though the approach is not limited to 3D and is valid in n-dimensions. We illustrate our approach on synthetic images containing objects with uniform intensity. We then describe a particular algorithm using Shells and Spheres and demonstrate it by segmenting the aortic arch in a contrast-enhanced CT scan, both in 2D and 3D.

Contents

1 Introduction

The framework of Shells and Spheres described in this paper is based on a set of spheres called a *sphere map*. A sphere map consists of exactly one sphere centered at each image pixel, whose radius can be adjusted. Calculations denoted as *Variable-Scale Statistics* (VSS) are performed on populations of pixels within spheres, as well as populations of adjacent and overlapping spheres. Memory and computational requirements are kept reasonable by storing only a relatively small, fixed number of VSS at every pixel, many of which can be updated incrementally when growing or shrinking spheres. The ultimate goal of adjusting radii is to produce a sphere map in which each sphere is as large as possible without crossing a boundary. The spheres' radii are thus equivalent to what is commonly known as a *distance map* [1]. Though the task is trivial in binary images, where definitive boundaries are known, it presents a challenge when boundaries are difficult to determine due to noise and tissue inhomogeneity. Our approach is well suited to this challenge, with the caveat that the correctness of segmentation of real images is subjective.

Many conventional methods for image processing consider a region of fixed size and shape, usually referred to as a *kernel*, especially when used for convolution. Other common approaches define dynamic regions adjoining boundaries using deformable contours [2] or level sets [3]. Our approach, instead, uses a set of spheres whose individual radii are optimized using VSS operators to achieve maximum discrimination between image regions. Not only do such spheres provide highly representative populations for boundary detection, but those spheres that touch at least two boundaries are also *medial*, as classically defined by Blum [4], providing a basis for medial feature extraction. Unlike Gaussian blurring, commonly used in multi-scale analysis [5], Shells and Spheres preserves sharp boundaries with increasing scale.

This paper presents the notation and basic operators of Shells and Spheres for computing VSS. Using this framework, a wide variety of algorithms for sphere map optimization are possible, and we present one such algorithm here. We then extend this algorithm to include methods to identify boundary and medial locations, followed by an application of our algorithm to image segmentation.

2 Methodology

We begin by defining our notation. As previously stated, Shells and Spheres is inherently n-dimensional. For brevity, we use the term *sphere* instead of *circle* or *hypersphere*, regardless of image dimension. Figures are presented in 2D for ease of illustration.

2.1 Shells and Spheres

Since the framework of Shells and Spheres is used to gather statistics on dynamic collections of pixels, we adopt a hybrid form of notation derived from standard set theory and statistics. We denote vectors by lowercase bold-faced letters ($\mathbf{x}$), scalars by lowercase italic letters (r), and sets by uppercase letters (S). We use $\mathbb{Z}$ to denote the set of all integers, and $\Omega \subset \mathbb{Z}^n$ to denote the set of all pixel locations in a sampled n-dimensional image.

Given an n-dimensional image with intensities $f(\mathbf{x})$ for $\mathbf{x} \in \Omega$, we define a *sphere map*, which assigns the radius $r(\mathbf{x})$ to the sphere centered at each pixel $\mathbf{x}$.

We define a sphere to be an n-dimensional neighborhood of pixels that lie within a radius r of a center point. We use an integer value for r, such that a sphere of radius r centered at a pixel $\mathbf{x}$ is given by

$$S_r(\mathbf{x}) = \{\mathbf{y} : \text{round}(|\mathbf{y} - \mathbf{x}|) \leq r,\, \mathbf{y} \in \Omega\}\,. \tag{1}$$

Note the shorthand notation for the subscript r, meaning $r(\mathbf{x})$, the radius of the particular sphere at $\mathbf{x}$ as given by the sphere map. In some instances, the reader will encounter an example with a different subscript, such as $S_1(\mathbf{x})$, meaning a sphere of radius 1, irrespective of $r(\mathbf{x})$. By definition, $\mathbf{x} \in S_r(\mathbf{x})$ for all $\mathbf{x}$, even when $r(\mathbf{x}) = 0$, and hence $S_r(\mathbf{x})$ is always non-empty.

A *shell* is a set of all pixels whose distance to the center rounds to a given radius, defined for radius r as

$$H_r(\mathbf{x}) = \{\mathbf{y} : \text{round}(|\mathbf{y} - \mathbf{x}|) = r,\, \mathbf{y} \in \Omega\}\,. \tag{2}$$

Shells are non-overlapping such that for concentric shells,

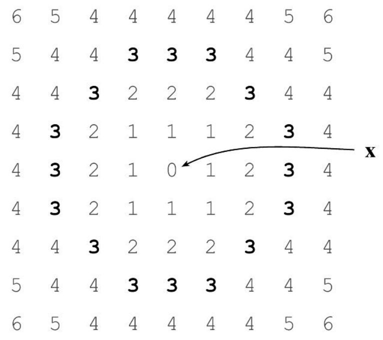
$$H_p(\mathbf{x}) \cap H_q(\mathbf{x}) = \emptyset,\;\; p \neq q\,. \tag{3}$$

Additionally, shells are space-filling, and thus a sphere of radius r may be formed from a union of shells,

$$S_r(\mathbf{x}) = \bigcup_{k=0}^{r} H_k(\mathbf{x})\,. \tag{4}$$

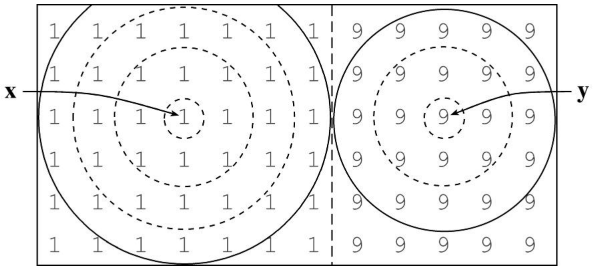
Figure 1 illustrates the distribution of pixels in a series of concentric shells surrounding a central pixel in a 2D image. Each pixel is labeled with its offset from the central pixel (labeled "x"). Fig. 2 shows an image containing two noiseless objects with pixel intensities of 1 and 9 respectively. Note that pixels in this case are represented by their intensity. The boundary between the objects is identified by a straight dashed line. Pixel $\mathbf{x}$ is surrounded by a concentric set of four shells $H_0(\mathbf{x})$, $H_1(\mathbf{x})$, $H_2(\mathbf{x})$, and $H_3(\mathbf{x})$, shown separated by dashed circles. Shell $H_3(\mathbf{x})$ is truncated by the edge of the image. The union of all four shells is $S_r(\mathbf{x})$, shown enclosed by a solid circle, with a radius governed by the value of $r(\mathbf{x}) = 3$ in the sphere map. Similarly, on the other side of the boundary, pixel $\mathbf{y}$ with a value of 9 has three shells whose union $S_r(\mathbf{y})$ has a radius $r(\mathbf{y}) = 2$. Both $S_r(\mathbf{x})$ and $S_r(\mathbf{y})$ touch but do not cross the boundary, and are therefore correctly optimized.

The correctly optimized sphere map of the image in Fig. 2 is shown in Fig. 3, with each pixel represented by the radius of the sphere centered at that pixel. Note the linear increase in sphere radius with distance from the boundary and the fact that the radius equals zero adjacent to the boundary.



6	5	4	4	4	4	4	5	6
5	4	4	**3**	**3**	**3**	4	4	5
4	4	**3**	2	2	2	**3**	4	4
4	**3**	2	1	1	1	2	**3**	4
4	**3**	2	1	0	1	2	**3**	4
4	**3**	2	1	1	1	2	**3**	4
4	4	**3**	2	2	2	**3**	4	4
5	4	4	**3**	**3**	**3**	4	4	5
6	5	4	4	4	4	4	5	6

Figure 1: Each pixel is shown as a number indicating its integer distance from the central pixel. If we denote the central pixel as $\mathbf{x}$, then pixels labeled n are members of the set $H_n(\mathbf{x})$. For example, the pixels labeled “3” (shown in bold) comprise the set $H_3(\mathbf{x})$.



1	1	1	1	1	1	1	9	9	9	9	9
1	1	1	1	1	1	1	9	9	9	9	9
1	1	1	1	1	1	1	9	9	9	9	9
1	1	1	1	1	1	1	9	9	9	9	9
1	1	1	1	1	1	1	9	9	9	9	9
1	1	1	1	1	1	1	9	9	9	9	9

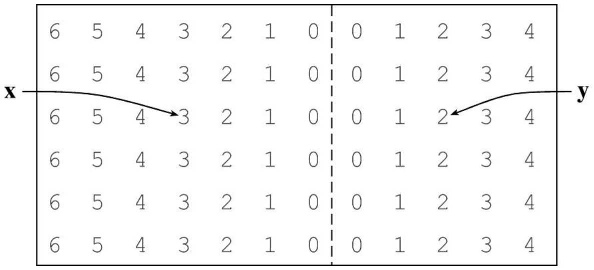
Figure 2: Noiseless image with boundary between two objects. Correctly scaled spheres $S_r(\mathbf{x})$ with $r(\mathbf{x}) = 3$ and $S_r(\mathbf{y})$ with $r(\mathbf{y}) = 2$ touch, but do not cross, the boundary. Numbers indicate pixel intensity.

2.2 Variable Scale Statistics

We derive a number of statistics at each pixel $\mathbf{x}$, calculated on the intensities of pixels within spheres. Since these statistics depend on the radii of the spheres, we call them Variable Scale Statistics (VSS). We denote as *primary statistics* those VSS at $\mathbf{x}$ calculated using only the population of pixels within $S_r(\mathbf{x})$. Higher-order *secondary statistics* are VSS derived from multiple spheres.

2.2.1 Primary Statistics

The primary statistics at pixel $\mathbf{x}$ concern only the population of pixels within the sphere $S_r(\mathbf{x})$. Thus the *mean* at pixel $\mathbf{x}$ is the mean intensity of all pixels within the population



6	5	4	3	2	1	0	0	1	2	3	4
6	5	4	3	2	1	0	0	1	2	3	4
6	5	4	3	2	1	0	0	1	2	3	4
6	5	4	3	2	1	0	0	1	2	3	4
6	5	4	3	2	1	0	0	1	2	3	4
6	5	4	3	2	1	0	0	1	2	3	4

Figure 3: Correctly optimized sphere map of the image in Fig. 2. Numbers indicate the integer radius of the sphere at each pixel. Pixels $\mathbf{x}$ and $\mathbf{y}$ are labeled as before.

$S_r(\mathbf{x})$, defined as

$$\mu(\mathbf{x}) = \frac{1}{|S_r(\mathbf{x})|} \sum_{\mathbf{y} \in S_r(\mathbf{x})} f(\mathbf{y}) \,, \tag{5}$$

where $|S_r(\mathbf{x})|$ is the number of pixels in $S_r(\mathbf{x})$ and $f(\mathbf{y})$ is the image intensity at $\mathbf{y}$. The *variance* at pixel $\mathbf{x}$ is defined as

$$\sigma^2(\mathbf{x}) = \frac{1}{|S_r(\mathbf{x})| - 1} \sum_{\mathbf{y} \in S_r(\mathbf{x})} [f(\mathbf{y}) - \mu(\mathbf{x})]^2 \,. \tag{6}$$

where $|S_r(\mathbf{x})| > 1$. The standard deviation $\sigma(\mathbf{x})$ is simply the square root of the variance.

The *first-order moment* of intensities within $S_r(\mathbf{x})$ is given by

$$\mathbf{m}(\mathbf{x}) = \sum_{\mathbf{y} \in S_r(\mathbf{x})} (\mathbf{y} - \mathbf{x}) f(\mathbf{y}) \,. \tag{7}$$

Due to the finite extent of an image's domain Ω, a sphere may be truncated by one or more edges of the image (for example, $S_3(\mathbf{x})$ in Fig. 2). Unlike conventional kernels, which usually require pixel values to be arbitrarily defined outside the image, our spherical sets simply exclude such locations from all calculations. Thus, truncation will not adversely affect $\mu(\mathbf{x})$ or $\sigma(\mathbf{x})$. However, the first-order moment will suffer a bias due to asymmetrical pixel distribution within the truncated sphere. We compensate for this in defining a measure that shows no edge effect, which we call *VSS gradient*. Given the center of mass of pixel locations in sphere $S_r(\mathbf{x})$,

$$\mathbf{c}(\mathbf{x}) = \frac{1}{|S_r(\mathbf{x})|} \sum_{\mathbf{y} \in S_r(\mathbf{x})} \mathbf{y} \,, \tag{8}$$

the VSS gradient at $\mathbf{x}$ is

$$\nabla f(\mathbf{x}) = \frac{1}{|S_r(\mathbf{x})|} [\mathbf{m}(\mathbf{x}) - \mu(\mathbf{x})(\mathbf{c}(\mathbf{x}) - \mathbf{x})] \,. \tag{9}$$

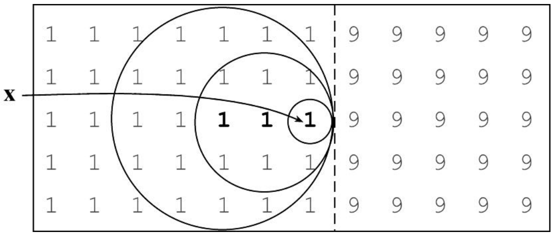


Figure 4: The $S^{-1}(\mathbf{x})$ set of spheres that contain pixel $\mathbf{x}$, adjacent to the boundary between two noiseless objects with respective intensities of 1 and 9.

The VSS gradient does not suffer from the usual edge effects of convolution kernels (see Fig. 6C). Note that for non-truncated spheres $\mathbf{c}(\mathbf{x}) = \mathbf{x}$ and VSS gradient is equivalent to the moment vector normalized to the number of pixels, $\mathbf{m}(\mathbf{x}) \,/\, |S_r(\mathbf{x})|$.

All of the above statistics can be computed incrementally as shells are added to, or removed from, a given sphere, significantly reducing computational load during sphere map optimization.

2.2.2 Secondary Statistics

We define secondary statistics as higher order VSS derived by combining multiple spheres to form more complex neighborhoods. One such neighborhood, $S^{-1}(\mathbf{x})$, is defined as

$$S^{-1}(\mathbf{x}) = \{\mathbf{y} : \mathbf{x} \in S_r(\mathbf{y})\}\,, \tag{10}$$

the set of all pixels whose spheres contain $\mathbf{x}$. The -1 supercsript is used to impart the flavor of an inverse function. Note that since it is always true that $\mathbf{x} \in S_r(\mathbf{x})$ it must likewise always be true that $\mathbf{x} \in S^{-1}(\mathbf{x})$.

Given an optimized sphere map, $S^{-1}(\mathbf{x})$ will consist entirely of pixels within the same object as pixel $\mathbf{x}$. Fig. 4 shows members of one such $S^{-1}(\mathbf{x})$ set, consisting of three pixels (bold), whose spheres contain $\mathbf{x}$. Notice that all three spheres touch but do not cross the boundary, so this particular $S^{-1}(\mathbf{x})$ set is correctly optimized.

Secondary statistics are derived from populations of spheres such as $S^{-1}(\mathbf{x})$. Thus, the *mean of means* is defined as

$$\mu_\mu(\mathbf{x}) = \frac{1}{|S^{-1}(\mathbf{x})|} \sum_{\mathbf{y} \in S^{-1}(\mathbf{x})} \mu(\mathbf{y})\,, \tag{11}$$

or the mean of the mean intensities for all the spheres in $S^{-1}(\mathbf{x})$. In a noiseless image containing distinct homogeneous regions, the $\mu_\mu(\mathbf{x})$ yielded by a correct sphere map will be identical to the original image.

Likewise, the *standard deviation of the means* is defined as

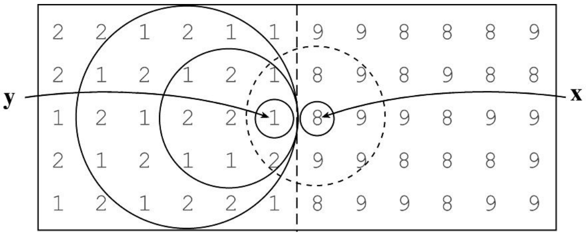


Figure 5: Image with noise. Pixel $\mathbf{x}$ is deterred from extending its sphere across the boundary because its mean is an outlier in the population $S^{-1}(\mathbf{y})$.

$$\sigma_\mu(\mathbf{x}) = \left[\frac{1}{|S^{-1}(\mathbf{x})| - 1} \sum_{\mathbf{y} \in S^{-1}(\mathbf{x})} [\mu(\mathbf{y}) - \mu_\mu(\mathbf{x})]^2 \right]^{\frac{1}{2}} . \tag{12}$$

Note that the above definition of $\sigma_\mu(\mathbf{x})$ is given only for $|S^{-1}(\mathbf{x})| > 1$. For noiseless images, a correct sphere map will yield $\sigma_\mu(\mathbf{x})$ values of 0 for all pixels.

We compute a *z-value* to provide a measure of how well $S_r(\mathbf{x})$ fits into the current $S^{-1}(\mathbf{y})$ set,

$$z_\mu(\mathbf{x}|\mathbf{y}) = \frac{|\mu(\mathbf{x}) - \mu_\mu(\mathbf{y})|}{\sigma_\mu(\mathbf{y})} . \tag{13}$$

The justification is that, in an optimized sphere map, if $S_r(\mathbf{x})$ were to contain pixel $\mathbf{y}$, then $\mu(\mathbf{x})$ should fall well within the distribution of means for all spheres that already contain $\mathbf{y}$. This concept is illustrated in Fig. 5, which shows pixel $\mathbf{x}$ attempting to extend its sphere across the boundary to include pixel $\mathbf{y}$. We have included noise in the image to demonstrate that a high z-value could be used to stop the growth of $S_r(\mathbf{x})$ at the boundary, even in the presence of noise. It should be noted that the utility of this statistic is dependent on a reasonable initialization of the sphere map, such that the percentage of spheres not crossing boundaries is high enough to lend statistical validity to $\mu_\mu(\mathbf{y})$ and $\sigma_\mu(\mathbf{y})$.

2.3 Demonstration on a Noiseless Image

We demonstrate the optimization of a sphere map using our system on a noiseless synthetic image containing a number of homogeneous objects as shown in Fig. 6A. Optimization on such an image is trivial; spheres are initialized to zero radius and allowed to grow until adding the next shell would yield a non-zero variance $\sigma^2(x)$. Figure 6B shows results of optimizing the sphere map on the synthetic image in Fig. 6A. Figure 6B shows negative ridges (dark) along boundaries, and positive ridges (bright) along medial manifolds. Figures 6C and 6D show, respectively, the y component of the VSS gradient and the variance calculated using spheres of radius 5 at all pixels in the image. Note that neither the gradient nor the variance suffer from edge effects. Note also that

Figs. 6C and 6D use a non-optimized sphere map, which includes spheres that cross boundaries.

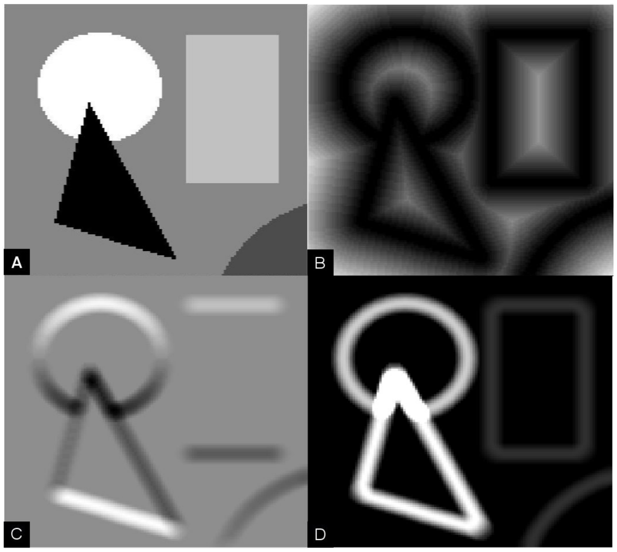


Figure 6: A: A noiseless synthetic image. B: The optimal radius image. C: The VSS gradient in the y direction, calculated using spheres of radius 5 pixels at all image points. D: The variance image, also calculated with all spheres set to a radius of 5.

3 Algorithm For Segmenting Real Images

The Shells and Spheres framework presented thus far provides basic tools that can be used in many ways. We have explored a number of these, and in the remainder of this paper we will present one effective algorithm for optimization of the sphere map, followed by segmentation of the aortic arch in a contrast-enhanced CT scan.

The algorithm presented here consists of six steps. Steps 1-4 optimize the sphere map. Step 5 identifies medial pixels, i.e. those whose spheres touch at least two boundaries. Given a medial seed point, step 6 connects neighboring medial pixels and combines their corresponding spheres to produce a segmentation. The following sections describe each step in more detail.

3.1 Step 1: VSS Gradient-Based Radius Initialization

For the noiseless synthetic image shown earlier (Fig. 6), it was trivial to optimize the sphere map by growing the spheres until any non-zero variance was detected. When analyzing real images, however, this approach will fail, because intensity variation due to noise may be indistinguishable from an object boundary, especially within small spheres. To yield viable statistical populations, at least some spheres must be correctly initialized. To accomplish this, we first set all spheres to $r(x) = 0$ and then allow spheres to grow until a persistent increase in VSS gradient magnitude is detected over a series of scales. Unlike conventional gradient measured with a fixed-scale kernel, the VSS gradient depends locally on $r(\mathbf{x})$, and is based on the first-order moment of intensity normalized by the number of pixels in the sphere. Thus we can expect VSS gradient to increase monotonically as a sphere grows past a boundary, since the first order moment favors the outer pixels. We look for a persistent increase in VSS gradient for g consecutive steps, at which point $r(\mathbf{x})$ is reset to the scale just before the increase began. Empirically, we have found that a value of $g = 5$ works well in our particular application to achieve a reasonable first approximation of the optimized sphere map.

Step 1 is effective at growing spheres past tissue inhomogeneity and noise. However, spheres may not stop exactly at the boundary, because of the inherent effect of noise on the detection of gradient increase. Using the gradient to govern sphere growth can also fail completely for a sphere that encounters two opposing boundaries simultaneously, as their contributions to the gradient may cancel.

3.2 Step 2: Variance-Constrained Radius Reduction

After Step 1, three possible states exist for each sphere: The sphere can be too large, too small, or the correct size (i.e. it touches the nearest boundary but does not cross it). Empirically, the most glaring error in the sphere map after Step 1 is the presence of large-scale spheres that have incorrectly grown past boundaries. As mentioned above, this typically occurs when a growing sphere contacts multiple boundaries at once, which is to say, when the sphere lies on the medial manifold. In such a case, the contributions to the gradient from multiple boundaries may cancel, allowing the sphere to grow past its correct scale. Such spheres will, however, have a larger variance $\sigma^2(\mathbf{x})$ than spheres that correctly remain within the object boundaries.

To address this case, we shrink all spheres with a variance above a certain threshold α, decrementing $r(\mathbf{x})$ for each sphere $S_r(\mathbf{x})$ until $\sigma^2(\mathbf{x}) < \alpha$. The threshold α is set relative to μ_{σ^2} and σ_{σ^2}, the mean and standard deviation, respectively, of the variance of all the spheres in the current sphere map, as defined by

$$\alpha = \mu_{\sigma^2} + \beta\sigma_{\sigma^2}. \tag{14}$$

The positive constant β represents the number of standard deviations above the mean permitted for a sphere's variance without the sphere being required to shrink. We have found that this parameter is not particularly sensitive, as the difference between spheres with erroneously high variance and all others is approximately two orders of magnitude. For the results presented, we used a value of β = 0.2.

It should be noted that the threshold depends on the current $r(\mathbf{x})$, which is not yet fully optimized. We address this by returning to variance in Step 4, once a more accurate $r(\mathbf{x})$ is available.

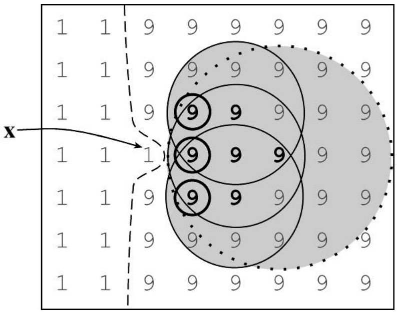


Figure 7: Illustration of $K^{-1}(\mathbf{x})$ containing 7 pixels (bold), each of whose sphere would place its reflector across the boundary at $\mathbf{x}$.

3.3 Step 3: Outpost Selection and Exclusion

A primary use of VSS is to differentiate regions on opposite sides of the boundary. To do this, however, most spheres must already not be crossing boundaries, as spheres that do cross boundaries will corrupt the statistics. Following Step 2, we find that enough spheres are approximately the right size that many face each other across boundaries, producing high values for $z_\mu(\mathbf{x}|\mathbf{y})$. We can identify a likely boundary candidate for a given sphere by finding a pixel in its $H_{r+1}(\mathbf{x})$ shell with a high value for $z_\mu(\mathbf{x}|\mathbf{y})$. The sphere at $\mathbf{x}$ is said to place a *reflector* at such a location, denoting a vote by the sphere for the pixel $\mathbf{y}$ across the nearest boundary. Thus, referring to Fig. 5, $S_r(\mathbf{x})$ could place a reflector at pixel $\mathbf{y}$. The set of reflectors placed by a given sphere $S_r(\mathbf{x})$ is denoted $K(\mathbf{x})$. In the present algorithm we limit $|K(\mathbf{x})| = 1$, that is, each sphere places only one reflector for reasons discussed below. This limit enables us to define

$$K(\mathbf{x}) = \{\mathbf{y} : \mathbf{y} = \underset{\mathbf{y}\in H_{r+1}(\mathbf{x})}{\operatorname{argmax}}\ z_\mu(\mathbf{x}|\mathbf{y})\}. \tag{15}$$

Each pixel may contain reflectors placed by a number of spheres, and we define the set of spheres that have placed reflectors at $\mathbf{x}$ as

$$K^{-1}(\mathbf{x}) = \{\mathbf{y} : \mathbf{x} \in K(\mathbf{y})\} \tag{16}$$

using the inverse notation as we did for $S^{-1}(\mathbf{x})$ in Eq. 10.

We refer to $|K^{-1}(\mathbf{x})|$ as the *reflector count*, i.e. the number of reflectors that have been placed at location $\mathbf{x}$. For example, in Figure 7, the reflector count $|K^{-1}(\mathbf{x})| = 7$.

The reflector placed by $\mathbf{y}$ at $\mathbf{x}$ has an inherent direction governed by the vector $(\mathbf{y} - \mathbf{x})$. We denote the vector sum of the directions of all of the reflectors at $\mathbf{x}$ as the *reflectance* $\mathbf{k}(\mathbf{x})$, defined as

$$\mathbf{k}(\mathbf{x}) = \frac{1}{|K^{-1}(\mathbf{x})|} \sum_{\mathbf{y}\in K^{-1}(\mathbf{x})} \frac{\mathbf{y} - \mathbf{x}}{|\mathbf{y} - \mathbf{x}|}. \tag{17}$$

Since we have decided that each sphere will contribute exactly one reflector, reflector density and reflectance are normalized over the image. We can therefore use reflector count to differentiate between significant collections of reflectors at boundaries and sparse distributions of reflectors in the interior of objects. To denote pixels containing a significant number of reflectors, we adopt the term *outpost*, since such pixels serve as border markers, and in fact, face each other across boundaries much the same way that military outposts of opposing armies face each other across the battle line.

The set of all pixels in an image chosen to be outposts is denoted by P, and in the present algorithm we find this set in two steps. First, we find the set of *primary outposts* P', which contains all pixels with zero radius and at least κ reflectors,

$$P' = \{\mathbf{x} : |K^{-1}(\mathbf{x})| \geq \kappa, r(\mathbf{x}) = 0\}. \tag{18}$$

For the results presented in this paper, $\kappa = 4$. To increase the density of outposts along the boundaries, we find a set of *secondary outposts* P'' containing all pixels with zero radius that adjoin an outpost in P' and have at least λ reflectors, where $\lambda < \kappa$,

$$P'' = \{\mathbf{x} : |K^{-1}(\mathbf{x})| \geq \lambda, H_1(\mathbf{x}) \cap P' \neq \emptyset, r(\mathbf{x}) = 0\}. \tag{19}$$

For the results presented, $\lambda = 2$. Combining the sets of primary and secondary outposts, we say P is the set of all outposts,

$$P = P' \cup P''. \tag{20}$$

Each outpost $\mathbf{y} \in P$ has a non-zero reflectance $\mathbf{k}(\mathbf{y})$. A sphere at $\mathbf{x}$ can distinguish whether a given outpost is on its side of the boundary, constituting a *friendly outpost*, or the other side of the boundary, being an *enemy outpost*, based on the direction of the outpost's reflectance. The set of enemy outposts (those with reflectance facing $\mathbf{x}$) within the sphere of radius $r(\mathbf{x})$, is defined as

$$E_r(\mathbf{x}) = \{\mathbf{y} : \mathbf{y} \in P \cap S_r(\mathbf{x}), \mathbf{k}(\mathbf{y}) \cdot (\mathbf{y} - \mathbf{x}) < 0\}, \tag{21}$$

where the sign of the dot product determines the direction of $\mathbf{k}(\mathbf{y})$ relative to $\mathbf{x}$, differentiating friendly from enemy outposts.

In governing the growth of a sphere, enemy outposts are to be avoided, while friendly outposts can be included. Step 3 uses the number of enemy outposts to adjust the sphere size as follows: If the pixel contains no enemy outposts in $S_{r+1}(\mathbf{x})$, the sphere grows until it does. That is

$$\text{If } |E_{r+1}(\mathbf{x})| = 0, \text{increase } r(\mathbf{x}) \text{ until } |E_{r+1}(\mathbf{x})| > 0.$$

If the number of enemy outposts in $S_r(\mathbf{x})$ is greater than γ, the radius is decreased until this is no longer true, i.e.

$$\text{If } |E_r(\mathbf{x})| > \gamma, \text{reduce } r(\mathbf{x}) \text{ until } |E_r(\mathbf{x})| \leq \gamma.$$

In the present implementation, $\gamma = 2$. This factor prevents lone pixels that have been improperly labeled as outposts from incorrectly causing spheres to shrink.

The effect of Step 3 is that significant densities of reflectors along boundaries correctly govern the size of spheres, sweeping incorrect reflectors from the centers of objects to the boundaries. Because most spheres are large relative to the spacing of outposts along the boundary, their growth will be stopped and they will not "leak" or "bleed" across boundaries.

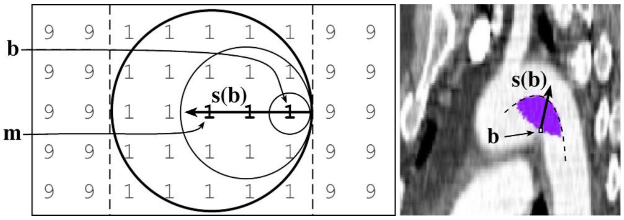


Figure 8: Diagram on left of an object with intensity 1 between two regions of intensity 9, showing set $S^{-1}(\mathbf{b})$ of pixels (bold) whose spheres contain pixel $\mathbf{b}$. This set produces an $\mathbf{s}(\mathbf{b})$ vector along which the furthest bold pixel $\mathbf{m}$ is the center of a medial sphere (circle in bold) touching both boundaries (dashed lines). Image on the right of a 2D slice through a CT scan of the aorta with contrast shows an actual $S^{-1}(\mathbf{b})$ set (purple/grey), the resulting $\mathbf{s}(\mathbf{b})$ vector, and the medial manifold on which the furthest sphere along $\mathbf{s}(\mathbf{b})$ must lie.

3.4 Step 4: Variance-Constrained Radius Growth

Although Step 3 results in a reasonably accurate sphere map, some spheres may still not reach boundaries, because of pixels incorrectly labeled as outposts. We have found it useful to grow spheres, using variance again to detect the boundary as in Step 2. Global variance measures μ_{σ^2} and σ_{σ^2} are recalculated, and subsequently α is recalculated using the same value for β as in Step 2. Due to further optimization of the sphere map, α represents a more suitable variance threshold. We increment $r(\mathbf{x})$ for all spheres while $\sigma^2(\mathbf{x}) < \alpha$. The effect is to make the spheres more precisely match the objects in the image.

At this point we consider the sphere map to be optimized.

3.5 Step 5: Medial Pixel Identification

Given that we have established an optimized sphere map, we aim to extract medial pixels. To do this, we first define a dense set of boundary pixels B from the sphere map as those with radius 0 or 1,

$$B = \{\mathbf{x} : r(\mathbf{x}) \leq 1\}. \tag{22}$$

We use $S^{-1}(\mathbf{b})$ for $\mathbf{b} \in B$ to find pixels on the medial manifold, whose spheres are as big as possible while still lying completely within the object. Figure 8(left) shows such a medial pixel (labeled "m") on the medial manifold of an object of intensity 1, between two regions of intensity 9.

To find such medial pixels within $S^{-1}(\mathbf{b})$, we first define an orientation $\mathbf{s}(\mathbf{b})$ orthogonal to the boundary as the vector sum of the normalized offsets relative to $\mathbf{b}$ for pixels within $S^{-1}(\mathbf{b})$ as

$$\mathbf{s}(\mathbf{b}) = \frac{1}{|S^{-1}(\mathbf{b})|} \sum_{\mathbf{y} \in S^{-1}(\mathbf{b})} \frac{\mathbf{y} - \mathbf{b}}{|\mathbf{y} - \mathbf{b}|}. \tag{23}$$

For each boundary pixel $\mathbf{b} \in B$, we identify the pixel $\mathbf{m} \in S^{-1}(\mathbf{b})$ furthest from the boundary along $\mathbf{s}(\mathbf{b})$ as a medial pixel, as depicted in Fig. 8(left). The set of all medial pixels M is thus

$$M = \{\mathbf{m} : \mathbf{m} = \underset{\mathbf{y} \in S^{-1}(\mathbf{b})}{\operatorname{argmax}} \left((\mathbf{y} - \mathbf{b}) \cdot \mathbf{s}(\mathbf{b})\right), \mathbf{b} \in B\}. \tag{24}$$

Figure 8(right) shows an actual $S^{-1}(\mathbf{b})$ set for a pixel b on the boundary of the aorta in a CT scan with contrast.

Selecting a single medial pixel from each $S^{-1}(\mathbf{b})$ potentially overlooks a large number of additional medial pixels on the outer edge of each set. However, we find the set M derived in this manner to be a sparse but reliable set of pixels on the various medial manifolds within the image.

3.6 Step 6: Medial Flood-Fill Segmentation

To segment a particular object, we select a medial seed pixel $\mathbf{p} \in M$ on that object's medial manifold and use a flood fill operator to find a connected subset $C \subseteq M$ containing medial pixels that are connected to $\mathbf{p}$. Pixels belonging in C are found iteratively using a series of sets C_i starting with C_0 containing just the seed pixel $\mathbf{p}$. At each subsequent step $i + 1$, the set C_{i+1} is created by adding medial pixels within a radius ϕ of pixels already in set C_i. More precisely, C_i is defined inductively as

$$C_0 = \{\mathbf{p}\} \tag{25}$$

$$C_{i+1} = \{\mathbf{x} : \mathbf{x} \in M, S_\phi(\mathbf{x}) \cap C_i \neq \emptyset\}. \tag{26}$$

For the results presented, scale ϕ was dynamically set to $\phi(\mathbf{x}) = r(\mathbf{x})/2$, as this causes the algorithm to search half-way from the medial manifold to the boundary for new medial pixels to include, therefore staying within the designated object. When a final step f adds no new pixels, such that

$$C_f = C_{f-1}, \tag{27}$$

the flood-fill is complete and we say that the set of connected medial pixels within the object is

$$C = C_f. \tag{28}$$

The set of spheres centered at these medial pixels effectively segments the object.

3.7 Results

Tests were performed on a 3D contrast-enhanced CT scan containing the aortic arch. Results can be seen in Fig. 9. Figure 9A shows a 2D sagittal slice through the raw 3D data. Figure 9B is the optimized sphere map $r(\mathbf{x})$ of this 2D slice after step 4 showing dark ridges along the boundaries and bright ridges along the medial manifolds. The *mean of means* image $\mu_\mu(\mathbf{x})$, obtained using spheres from the optimized sphere map,

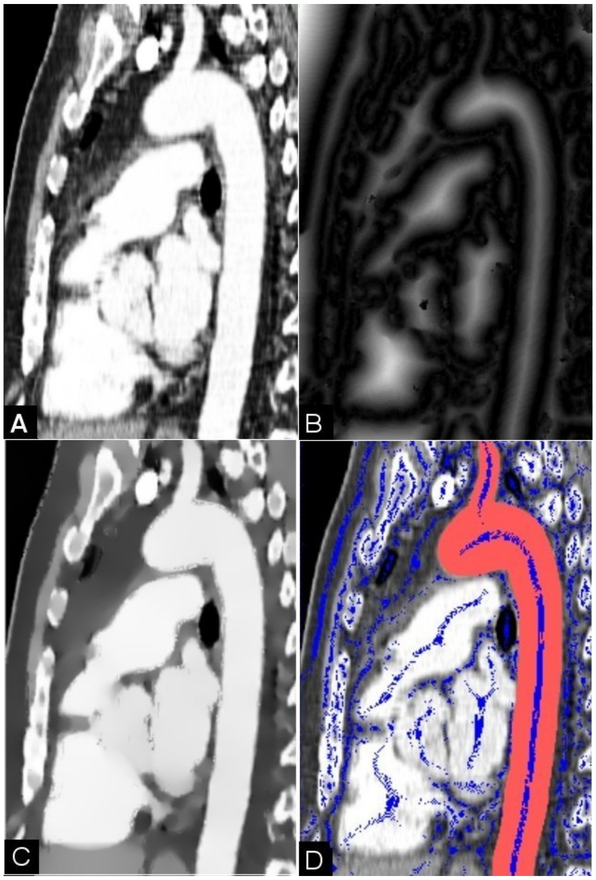


Figure 9: A: 2D CT image of the aortic arch. B: Optimized $r(\mathbf{x})$ image. C: Measured mean of means $\mu_\mu(\mathbf{x})$ image given optimized $r(\mathbf{x})$. D: Segmentation of aortic arch (pink/dark gray) and medial pixels within the image objects (blue/dark gray).

is shown in Fig. 9C. A reduction in noise within objects, without significant blurring of boundaries, is evident. Fig. 9D shows the corresponding medial pixels in set M as blue/dark spots and the segmentation from a seed pixel within the aorta as pink/dark shading.

Figure 10 shows the results of the algorithm on the full 3D CT data set. The surface model shown is simply the union of all spheres in the segmentation set C calculated from a medial seed pixel within the aortic arch. The lighting on the surface model was produced using OpenGL to render the spheres in set C.

Run time for the algorithm on 2D images (512x512) was typically 1-2 minutes on a 2.4 GHz Pentium 4-M laptop with 1 gigabyte of RAM. Run time for 3D images (155x90x75) was typically 3-4 hours using a single processor and 2 gigabytes of RAM on a 2 GHz 64-bit quad-Opteron workstation.

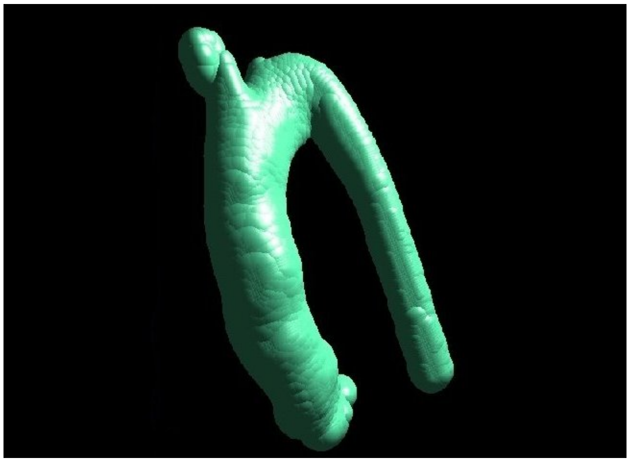


Figure 10: Surface model of the aortic arch produced by applying our algorithm to a 3D contrast-enhanced CT scan, using a single manually placed seed point.

4 Conclusions

Shells and Spheres provides an n-dimensional framework for computing Variable Scale Statistics (VSS), using local pixel populations whose size and shape conform to objects in a particular image. The populations are assembled from spheres whose radii vary from pixel to pixel, providing the benefits of rotational invariance without sacrificing the ability to analyze arbitrarily shaped regions. Separate regions span the entire extent of objects on both sides of the boundary, rather than being limited to a small neighborhood near the boundary.

Shells and Spheres differs from multi-scale approaches based on isotropic Gaussian filters in that our domains are not weighted toward zero at their extremities, thus remaining "sharp" at large scale. Our approach is similar to *anisotropic diffusion*[**?**], in that we seek to avoid blurring boundaries while considering large pixel populations.

The radius image $r(\mathbf{x})$ is analogous to what is often called the *distance map*, which has been well explored for pre-segmented images [1]. We aim to converge on an optimal $r(\mathbf{x})$ on unsegmented grayscale images. We determine a single optimal radius for each pixel such that each sphere touches but does not cross the nearest boundary. Troughs represent boundaries, where $r(\mathbf{x})$ is zero. Ridges represent the medial manifold, where $r(\mathbf{x})$ is the medial scale.

We have presented Shells and Spheres and VSS operators, along with one algorithm to optimize $r(\mathbf{x})$ and segment the aortic arch in 2D and 3D from a single seed point. We have shown that the framework is computationally tenable. Although we feel this first algorithm requires too many parameters, the underlying framework offers many possible avenues for further development.

References

[1] Per Erik Danielsson, “Euclidean distance mapping,” vol. 14, no. 3, pp. 227–248, Nov. 1980.

[2] M. Kass, A. Witkin, and D. Terzopoulos, “Snakes: Active contour models,” *International Journal of Computer Vision*, vol. 1, no. 4, pp. 321–331, 1988.

[3] James Albert Sethian, *Level set methods: Evolving interfaces in geometry, fluid mechanics, computer vision, and materials science*, Number 3 in Cambridge monographs on applied and computational mathematics. Cambridge University Press, Cambridge, U.K., 1996, 218 pages.

[4] H. Blum and R. N. Nagel, “Shape description using weighted symmetric axis features,” *Pattern Recognition, 10*, pp. 167–180, 1978.

[5] T. Lindeberg, “Scale-space theory in computer vision, kluwer, dordrecht,” *Monograph 1994*, 1994.