2010

Analysis of Medical Images by Colonies of Prehending Entities

Rebecca Smith
Virginia Commonwealth University

School of Engineering

Virginia Commonwealth University

This is to certify that the thesis prepared by Rebecca Smith entitled ANALYSIS OF MEDICAL IMAGES BY COLONIES OF PREHENDING ENTITIES has been approved by her committee as satisfactory completion of the Thesis requirement for the degree of Master of Science

---

David Primeaux, Ph.D., Committee Chair, Department of Computer Science

---

Lorraine Parker, Ph.D., Dept. of Computer Science, School of Engineering

---

Ramana Pidaparti, Ph.D., Dept. of Mechanical Engineering, School of Engineering

---

Rosalyn S. Hobson, Associate Dean of Graduate Studies, School of Engineering

---

Russell Jamison, Ph.D., Dean, School of Engineering

F. Douglas Boudinot, Ph.D., Dean of the School of Graduate Studies

---

Date

ANALYSIS OF MEDICAL IMAGES BY COLONIES OF PREHENDING ENTITIES

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science at Virginia Commonwealth University.

by

REBECCA SMITH

B.Eng., Imperial College (London, United Kingdom), 2003

Director: DAVID PRIMEAUX

ASSOCIATE PROFESSOR, DEPARTMENT OF COMPUTER SCIENCE

Virginia Commonwealth University

Richmond, Virginia

May, 2010

Table of Contents

Page

List of Figures

Abstract

ANALYSIS OF MEDICAL IMAGES BY COLONIES OF PREHENDING ENTITIES

By Rebecca Smith

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science at Virginia Commonwealth University.

Virginia Commonwealth University, 2010

Major Director: David Primeaux
Associate Professor, Department of Computer Science

The concept of emergent behavior is difficult to define, but can be considered as higher-level activity created by the individual actions of a population of simple agents. A potential means to model such behavior has been previously developed using Alfred North Whitehead's concept of Actual Entities. In computational form, actual entities are agents which evolve over time in response to interactions with their environment via the process of prehension. This occurs within the context of a Colony of Prehending Entities, a framework for implementation of AE concepts. This thesis explores the practical application of this framework in analysis of medical images, with specific focus on Computed Tomography (CT) and Magnetic Resonance Imaging (MRI) scans. Specialized Slice COPEs are devel-

oped for analysis of individual image slices from these scans, focusing on the detection and segmentation of structures of interest (such as bone matter, ventricular tissue, and tumors). These structures exist in 3D and can be extracted across multiple consecutive scan slices. Therefore, a specialized Scan COPE is also proposed which aims to render the structure's volume via interpolation between previously analyzed slice images. The software developed for the specified application also provides visualization of a COPE's evolution toward its goal. This has additional value in general study of the COPE framework and the emergent behavior it generates.

CHAPTER 1 Theory of Actual Entities

1.1 Introduction

The philosophical concepts underpinning the theory of Actual Entities are far-reaching, and would require another thesis in themselves to explore in detail. This chapter therefore presents a brief overview of the aspects most relevant to this study, with further investigation left to the interested reader.

1.2 Process Philosophy

Process philosophy (or process theory) holds that existence is characterized by both processes and substance. This theory was first formulated in the twentieth century by Alfred North Whitehead. A mathematician and logician, who co-wrote *Principia Mathematica* with Bertrand Russell and developed a theory of relativity as an alternative to Einstein's, Whitehead suggested that reality consists entirely of atomic events of experiences, which he named Actual Entities. What we perceive as static objects are actually sequences of these actual entities (AEs); the objects are in constant flux, and only appear to be continuous in time. AEs can be grouped together. For example, a human being can be considered as a grouping of billions of AEs, all functioning independently, and yet also as an entity in itself. Process philosophy is also known as the ontology of becoming; each AE is in the process of becoming another. Whitehead first published his theory in 1929 as *Process and*

Reality [16], a work based on the Gifford Lectures he gave at the University of Edinburgh in 1927-1928.

1.3 Actual Entities

Whitehead intended to construct "an atomic theory of actuality" - and in their most basic sense, Actual Entities form the atomic units of this theory. An actual entity (AE) cannot be broken down into anything smaller than itself; in Whitehead's words, AEs are "the final real things of which the world is made up" [16]. An AE is self-determining and experiences the world around it, an act which alters its internal constitution. In this way, an AE is characterized by its history - i.e. its past experiences - which has lead to its current state. This ties in with Whitehead's principle of relativity, which states that the essence of every entity is determined by its relation to every other entity. As Whitehead states, "to be is to be a potential for every subsequent becoming" [16]. In other words, an AE's state is not only the result of its entire history of experiences, this history determines the way it affects other AEs in the future.

1.4 Prehension

Prehension can be considered the method by which one AE interacts with another object and incorporates its observations of this object into its experience (i.e. its internal constitution). In prehensions relevant to this research, one AE takes the role of the prehending entity (Pr) and the observed object the role of the prehended entity (Pd).

Whitehead describes prehensions as consisting of three components [16]:

1. The prehending subject

2. The object being prehended

3. A subjective form specifying *how* the subject prehends the object

By prehending other objects, an AE changes its internal state in response to its environment. In other words, it is accumulating experience - or learning. Note that by Whitehead's guidelines, only Pr may be changed by prehension; Pd is simply observed. (Despite the description of prehensions relying on observations, it should be remembered that these observations are not simply equivalent to sensory input in humans; they pertain to the whole environment of the prehending AE rather than to only those characteristics humans are able to perceive.)

Prehensions can be positive or negative. A positive prehension is a prehension which causes Pr to change itself, in order to incorporate new experience gained. A negative prehension is one in which no change occurs, meaning Pd does not contribute to the experience of Pr. How is it decided whether a prehension is positive or negative - or rather, by what criteria does an AE decide to incorporate or discard its observations of other objects? This is not an easy question to answer. Philosophically, this can loosely be described as depending on the AE's subjective aim - i.e. what it is in the process of becoming, or its ideal goal. In a computational sense, the answer is somewhat simpler. The exact nature of prehension and the conditions under which it will cause Pr to change are defined by a prehension function, as specified by the designer according to the system's purpose. The possible methods of implementing these functions in a computational simulation of AE behavior are explored in the next chapter.

CHAPTER 2 Computational Actual Entities

2.1 Role in Artificial Intelligence

Though influential on theologians and philosophers later in the 20th century, Whitehead's work has received relatively little attention outside of those fields. However, their self-determining nature and constant evolution based on their environment make Actual Entities highly suited to use in computational intelligence.Several prior studies have investigated this possibility. Henry conducted a systematic analysis of Whiteheadian process theory via programs developed the Prolog programming language [7]. Though the focus was on analyzing Whitehead's philosophy rather than applying it to a practical task, this was nonetheless one of the first implementations of his ideas in computational form. A later study by Panteli and Dibben [10] explored a potential real-world use of Whitehead's ideas - namely, a theory of virtuality in which the emergence and evolution of virtual organizations is viewed in terms of processes.

Whitehead himself attempted a mathematical formalism of his theories known as mereotopology, which employs mereological and topological concepts to express the relationships between parts and wholes. Though his explanation was incomplete, more recent studies have formalized his ideas and encouraged the use of mereotopology as a tool in ontological analysis and a means to express the notions of boundaries and interiors. One analogous concept in computer science is that of objects in programming. A programmer defines an object as having particular attributes and a set of rules controlling potential actions.

These objects can be specific versions of generalized ideas, forming a hierarchy where each level becomes progressively more narrow in focus (e.g. 'Dog' as a particular example of 'Mammal'). However, there is no simple or standard way to allow one object to evolve into another; any such change is treated as binary and must be defined as occurring at a particular point in the object's existence. This is reflective of the way a computer operates. In order to model the evolution of one object into another - or more generally, any change it undergoes over the course of its existence - we must develop techniques of modeling emergent behavior. When accumulated into a population, AEs are well suited to this, as they evolve over time; moreover, they do so in response to their environment, allowing modeling of complex real-world phenomena where multiple internal and external factors influence the state of some entity at any given time. The question then becomes how to simulate the behavior of a population of AEs within a computational context.

2.2 Colonies of Prehending Entities

Colonies of Prehending Entities, or COPEs, were proposed by Primeaux [12] as a method of studying the behavior of AEs within a closed environment. A COPE consists of a population of AEs which can be homogeneous or heterogeneous in their characteristics and prehension functions. By definition, a COPE has an ultimate goal which is passed on to its population of AEs; examples include modeling a real world phenomenon or accomplishing a specific task (such as the image analysis explored in this thesis). The AEs strive to achieve this goal via their individual and group behavior, altering the organization of the COPE as the population evolves.

Note that the standard computational implementation of a COPE does not fully obey the

principles of Actual Entity interaction. Ideally, all AEs in the COPE should be prehending simultaneously with no pause in their activities. Due to computational limitations, this parallel behavior is not easily implemented. It would require distributed processing across a cluster of machines; furthermore, the computational cost would increase with the COPE's size and complexity, demanding additional hardware for effective parallel processing. It would also require creation of a distributed virtual COPE encompassing all the COPES across the individual computers in the cluster.

A simple workaround is to have prehensions occur sequentially, but at rapid speed. This requires an artificial implementation of time. The most intuitive form would be a series of 'ticks', during which a round of prehension occurs. On a modern, reasonably powerful computer with a COPE of relatively low complexity, this is sufficient to give the illusion of parallel behavior. The standard approach is to select an AE at random from the COPE, have it prehend every other member, then combine the results of all its prehensions in a vector to be stored inside the entity. This process is repeated until all the AEs in the COPE have prehended every other AE, at which point each changes its state to that represented by the stored vector. This constitutes a single time step.

The implementation in this study deviates slightly from this approach, as a time step constitutes the prehension activity of only one AE rather than that of the entire population. This is done for multiple reasons, one of which is to enable better dynamic visualization of COPE behavior; shorter time steps (i.e. those involving less activity) allow for smoother updating of a 3-D display. Furthermore, COPEs generated from images are densely populated, and the single-AE approach deals with this by limiting the number of redundant prehensions while still preserving a high degree of randomness.

2.2.1 Object Representation

Arguably the most intuitive representation for an Actual Entity is as a vector of parameters. In other words, an AE is defined by its position in n-dimensional space, where n is the number of parameters it possesses. An AE $\mathbf{P}$ can thus be represented as:

$$\mathbf{P} =< p_1, p_2, \ldots, p_n > \tag{1}$$

where the parameters $p1, \ldots, p_n$ depend entirely on the specification of the individual AE and the behavior being modeled. Representing an AE as a vector is not only computationally simple, but also enables the use of masks in constructing prehension functions and controlling thresholds. This will be explored in more detail in later chapters.

2.3 Prehension Functions

Though Whitehead did not formalize the notion of prehension into a mathematical process, the description and conditions outlined in Section 1.4 lend themselves to computational implementation in the form of functions. Though possible implementations vary, an intuitive representation has previously been developed by Primeaux [14] which relies on the concept of distance between the prehending and prehended objects, computed as a vector $\mathbf{d} = \mathbf{o^t} - \mathbf{i}$, where $\mathbf{o^t}$ is the state of the prehender at time t, and $\mathbf{i}$ is the data for prehension (or, the state of the object being prehended). Primeaux proposes the use of the n-dimensional Euclidean distance, where n is the number of attributes included in the prehension. Positive and negative prehension are decided by means of thresholds on these distances; if the distance exceeds the specified threshold, $\mathbf{i}$ is not prehended. This results in the following threshold function:

$$w_j = \begin{cases} 1 - |d_j| & \text{if } \sqrt{d_0^2 + d_1^2 + \cdots + d_n^2} \leq \epsilon \text{ is even} \\ 0 & \text{otherwise} \end{cases}$$

where

$$d_j = o_j^t - i_j \tag{2}$$

and j is the dimension (attribute) under consideration, ϵ is the specified threshold and the components d_i are as specified in the preceding paragraph. The components w_j form a vector $\mathbf{w}$ across all n dimensions; if the prehension is negative, $\mathbf{w}$ is the zero vector. The vector $\mathbf{w}$ can be used to calculate a vector containing the scaled differences between the two entities as follows:

$$s = \frac{\sum_j w_j}{n} \tag{3}$$

This is simply the average amount by which the (scaled) differences exceed 0, and is used as a weighting factor to control the movement of AEs. However, a second use is also proposed, where $\mathbf{w}$ performs weighting of individual attributes in the overall impact of the prehension on the prehending AE. The scaled values are then used to calculate the subjective form $\mathbf{f}$ of the prehension - i.e. the impact of prehension upon the prehender - as:

$$f_j = s w_j d_j \tag{4}$$

This is a reformulation of an earlier approach which incorporated the idea of an AE's

disposition to trust prehensions at a given time t, represented by a variable g^t [13].

Finally, the prehender's new state (i.e. position across the dimensions of its attributes) is calculated as:

$$o_j^{t+1} = o_j^t - f_j \tag{5}$$

Intuitively, this formulation of a prehension function provides a measure of similarity between the prehender and the prehended object. This study adopts a similar form, albeit with a few changes that will be explained in later chapters.

2.4 Practical Applications

Various applications of Actual Entities and COPEs have already been proposed or implemented, many of them focusing on modeling complex real-world behavior. For example, Pidaparti, Primeaux and Saunders used the COPE framework to model molecular self-assembly. Specifically, AEs were used to simulate the hierarchical manner in which microtubules assemble themselves from tubulin monomers of protein monomers into filaments, rings, and other structures [11]. Self-assembly is an ideal concept to be modeled by AEs, which act individually within a larger population in order to achieve a specified goal. More specifically, the actions of an AE depend fully on the characteristics (and by extension, activities) of other objects in its COPE environment. This particular study involved a COPE consisting of a random population of AEs and a single Eternal Object (EO) representing the desired shape or object. Eternal Objects are a complex concept in Whiteheadian theory, but can be sufficiently explained for the present purpose as distinguished AEs that never prehend, but rather represent some ideal state for the AEs in the COPE. The AEs prehend only the EO, and the prehension function used collapses

the randomly dispersed AEs into the shape the EO describes. For example, if the desired outcome is a circle, then the AEs collapse on one dimension (representing a single plane). The prehension thresholds are set based on the circle radius specified by the AE.

The use of COPEs in clustering data has also been explored [12]. Although the principles of emergent behavior have been applied to this task in the past via other approaches, an AE is able to hold the data it is attempting to cluster by means of additional parameters. The process of actual clustering is simply a more specific form of prehension function. Desirable features of the data can be grouped together and used to calculate metrics for the purpose of clustering. Primeaux proposes the use of Euclidean distance, as described in Section 2.3, to generate a simple attraction function that causes AEs with similar characteristics to converge into clusters.

Two other proposed applications include modeling of political affiliations and the spread of ideas and modeling corrosion of different materials in various weather conditions. Both are examples of complex phenomena involving many factors and considerable randomness. COPEs have also been used in the generation of electronic music, an example of how emergent behavior of AEs may result in interesting patterns; a variation of this would be the use of COPEs containing AEs with location and color parameters to create images. In fact, AEs have much potential in image-based applications, one of which is explored in the next section.

2.5 COPEs in Image Analysis

This study focuses on the applications of AEs in medical image segmentation, a specialized form of image analysis where the aim is to detect a specified object and extract it from

the background. AEs are well-suited to this due to their propensity to group themselves based on similarities, as described in Section 2.3. By extension, they are also suitable for edge detection. A COPE can be initialized with AEs that assume the positions and color values of pixels in the image; with appropriate thresholds and attribute use, a prehension function can detect sudden discontinuities in the color values of the image that are likely to represent edges. Similar to the aim of segmentation is the task of identifying a sub-image within a larger parent image. This sub-image need not represent any particular object; the process of identification is based purely on attraction. Both the sub-image and the parent are used to initialize a COPE, the former contributing a set of non-prehending objects and the latter a population of standard AEs. The objects created from the sub-image are initialized in a plane some specified distance from the AEs. The AEs then prehend these objects using a simple attraction function in which positive prehensions causes them to move toward the desired sub-image. These AEs record their original positions in $x - y$ dimensions. When convergence is complete, those AEs which have reached the sub-image plane contain the $x - y$ coordinates of the corresponding pixels in the parent image.

This concept of a single axis of movement - which can be visualized as a third axis raising AEs up from the initial flat array - is put to use in this study as a means of drawing desired structures out from the image. In the case of segmentation, there is no sub-image; the shape of a particular organ or bone structure varies from person to person and scan to scan, particularly in pathological cases. However, non-prehending AEs can be chosen that best represent the structure, then used to define the subjective goal of the standard AEs by representing some ideal state. This concept is revisited in Chapters 4 and 5.

2.6 Similar Approaches

2.6.1 Swarms

A swarm can be considered as a grouping of simple agents which interact with their local environment and other agents in their group. Each agent acts independently, but when aggregated, their actions and interactions give rise to emergent behavior within their group. The term was first defined by Beni and Wang [1], in reference to the "group intelligence" displayed in nature by ant colonies, schools of fish, and flocks of birds. The parallels with Actual Entities are clear. Both share the basic tenets of decentralization and self-organization; i.e. agents follow simple internal behavior rules, as opposed to operating under a centralized command structure inside or outside of their group. Effectively the group of agents governs itself, forming its own patterns and altering its internal organization. As with AEs, the behavior of a computerized swarm is highly dependent on the implementation method, and the parameters of both the agents and the swarm itself. However, while AEs represent both data and intelligent agents, swarms typically consist of agents acting on external data.

2.6.2 Cellular Automata

A cellular automaton (CA) consists of a grid of cells, each of which is permitted to be in one of a predefined finite range of states. This grid evolves over time depending on each cell's interaction with its neighbors. Consider a CA containing n cells, each of which is assigned to some initial state s_j $(s = 0, \ldots, m)$ at time $t = 0$. When the grid advances to an arbitrary time-step t, every cell considers its current state and determines whether to change to a different one. More specifically, a cell c_i $(c = 0, \ldots, n)$ applies a state-transition

function to the collective states of all cells in its surrounding neighborhood, the output of which is c_i's state at time $(t+1)$. Both the function and the size of the cell neighborhood are defined in advance for the entire CA and remain fixed throughout its evolution. These changes occur synchronously, creating a new generation at each increment in t. Thus, the CA exhibits emergent behavior; in fact, probably the most famous implementation of a CA is the *Game of Life* computer program created in 1970. This implements a CA as an infinite 2-D grid populated by cells with two states: alive or dead (visually, black or white). Each cell changes state according to four simple rules applied to its 8-neighborhood. The user interacts with the program only in deciding the CA's initial configuration; all subsequent evolution is decided entirely by this and the state-transition functions.

There are strong parallels between the cells of a CA and the AEs within a COPE. The cells are clearly more restricted in their evolution; they do not move in the dimensions of their parameters as AEs do, but simply transition between a range of states. In fact, its state is the only parameter a cell has available to use in the grid's transition function. Nonetheless, AEs and CAs produce emergent behavior in a similar way: as self-contained machines that change in response to other objects. A single cell and a single AE serve no purpose - their evolution is dependent on their interactions with their environment. Emergent behavior is only made possible when the cells or AEs are assembled into the larger population of a CA or COPE. Attempting to formalize these similarities and define a relationship between the two approaches raises an interesting possibility, which will be explored in the following section.

2.7 The COPE Framework

In their use of a single state-transition function which relies strictly on a cell's immediate neighborhood - and the concept of a fixed range of states - a CA could be considered as a specialized case of an COPE of AEs. Generalizations of the CA concept have already been proposed: for example, via the incorporation of probabilistic transition functions or the use of multiple functions within a single grid. AEs allow for both these extensions and more, along with a means to restrict their behavior to that of a CA cell. Furthermore, discussions between Primeaux and the author have examined whether a swarm can be considered a specialized implementation of a COPE. This relationship has not been formally proven, but it seems intuitive given their shared base concept as a group of agents reacting to their environment. If a swarm is viewed as a collection of agents which change in response to the data provided by their environment and other agents in the vicinity, this can be replicated by a COPE; AEs contain the data they act on, meaning that swarm-like behavior simply depends on the choice of prehension function. If a COPE can be used to implement a swarm, it can also implement sub-types such as ant colonies and flocks, again by altering the method of prehension. In fact, it is the use of prehension that affords AEs such great flexibility in the data and phenomena that they model. Other intelligent agents do not possess an analogous function and are therefore more restricted in their capabilities. This presents the interesting possibility that COPEs can provide a general framework for modeling emergent behavior of all kinds, with specialized methods such as swarms being implemented via specific variations in the prehension function. Further study would be needed to prove this formally; the outcome could have considerable impact on the role of

Actual Entities in computational intelligence.

CHAPTER 3 Medical Image Segmentation

3.1 Background and Motivation

Medical imaging technology has made major advancements in the course of the last fifty years. Two of the major modalities took hold in the 1970s, which saw the construction of the first Computed Tomography (CT) scanner and whole-body Magnetic Resonance Imaging (MRI) machine. More recent developments such as functional MRI (fMRI) and the combined Positron Emission Tomography/Computed Tomography (PET/CT) scanner, both of which generate 3-D images of the body's functional processes. These various imaging modalities enable physicians to make non-invasive diagnoses of disease and injury, perform screening prior to additional procedures, plan surgery, and monitor a patient's recovery. In fact, medical personnel may now be faced with the problem of too much data rather than too little. While the technology may have improved, the means of analysis are still mostly manual and remain time-consuming even for experts in the field. This discrepancy has generated a high level of interest in computerized processing of medical images via both automated and semi-automated methods. Though the individual applications vary widely, the essential aim is to combine the advantages of machine analysis - such as the ability to rapidly process huge volumes of data and identify patterns, and the fact that most modalities return their output in digital form - with the knowledge and flexibility of human experts [5].

A particularly important subfield, often required prior to any other analysis, is the seg-

mentation and detection of known anatomical structures from medical images. In some applications this is still done manually by physicians, using basic metrics or simply by sight; tumor volumes are calculated in this manner when deciding the best means of treatment [6]. This process can be time-consuming and prone to inaccuracies, leading to recent interest in systems that automate or semi-automate the task. Accurate delineation of anatomical structures and tissue types can not only provide a framework for further quantitative analysis, but also facilitate visualization of the detected structures in a manner that provides the most information to physicians and surgeons.

3.2 Modalities

A wide variety of modalities exist for capturing images of the human body. Examples include X-ray radiography, positron emission tomography (PET), ultrasound and mammography. This thesis will focus on two modalities in particular: computed tomography (CT) and magnetic resonance imaging (MRI). Both capture sections of the body as a series of 2-D images ("slices") around a single axis of rotation.

3.2.1 Computed Tomography

Tomography is the method of imaging a structure by sections, producing slices along a plane that form a tomogram. In the case of CT, a large number of 2-D X-ray images are taken around a single specified axis, forming a CT scan. Typical planes used include the transverse (orthogonal to the vertical axis of the upright body), sagittal (orthogonal to the left-to-right horizontal axis), and coronal (orthogonal to the front-to-back horizontal axis). Figure 3.1 presents an example CT slice taken from an axial scan of the lungs.

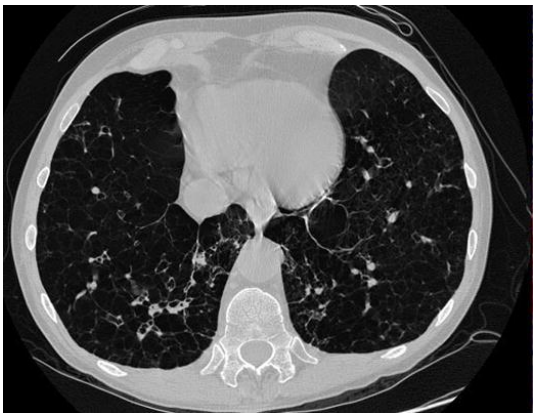


Figure 3.1: A slice from an axial CT scan of the lungs.

3.2.2 Magnetic Resonance Imaging

Magnetic Resonance Imaging (MRI) is a newer imaging modality that offers far greater contrast between soft tissues than CT, and consequently a higher level of detail. It also does not require exposure to ionizing radiation. Instead, the method relies on the use of magnets to align hydrogen atoms in water molecules found in human tissue. By repeatedly altering the alignment of the atoms using a radio frequency (RF) electromagnetic field, a signal is generated which can ultimately be used to construct a series of image slices of the chosen body region. When combined in sequence, these slices form a full MRI scan. Figure 3.2 presents an example slice taken from an axial view of an MRI scan of the brain.

Despite the higher image quality of MRI scans and the risks posed by CT in terms of radiation exposure, the latter is still in widespread use. CT imaging is both cheaper and quicker; furthermore, it causes less disruption to patients in critical state and can be performed using portable scanners in emergency rooms. It also offers particularly good delineation of bone and blood matter. However, soft tissue contrast is poor, since the X-

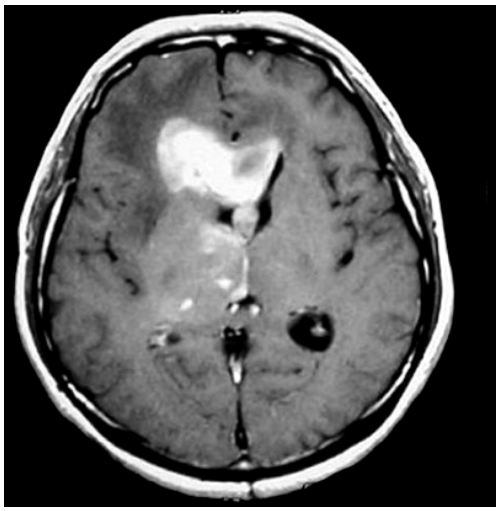


Figure 3.2: A slice from an MRI scan of the brain (axial view).

rays must be blocked by dense tissue when creating each CT slice. Meanwhile, since soft tissues typically contain water molecules, MRI imaging offers excellent soft-tissue contrast. Its primary medical use is therefore in distinguishing pathological tissue, particularly in the case of tumors. In both modalities, the scan slices can be viewed individually, in sequence, or even used to generate volumetric 3-D representations of anatomical structures, as shown in Figure 3.3. However, one disadvantage of 3D volumetric renderings is that they require a reduced slice thickness for smooth reconstruction. Consequently, a much large number of images must be generated and processed per scan, which also subjects the patient to a higher dose of radiation.

3.3 Other Approaches

Unlike generic segmentation techniques, modern medical image segmentation techniques are typically application specific and take advantage of existing clinical and anatomical knowledge. For example, it is known that in transverse scans of the human brain, the

Figure 3.3: 3D representation created from chest CT scan.

lateral ventricles should remain roughly in the centre of the skull; a large deviation from this indicates tissue deformation and underlying pathology.

The literature concerning image segmentation is vast in scope, encompassing many different techniques, many geared toward specific modalities or even individual tissue types within a modality. Example approaches include statistical modeling, thresholding, image registration, edge detection, and mixture models. The review presented here will focus specifically on those based on emergent behavior and intelligent agents, in keeping with the aim of applying Actual Entities to the segmentation task.

Liu and Tang have explored the use of intelligent agents in segmenting greyscale images [8]. This approach treats an image as a 2-D cellular environment, populated by agents attempting to segment homogeneous regions. These agents operate directly on individual pixels, altering their behavior in response to the characteristics of neighboring regions (such as relative contrast and mean). The method has shown reasonable success in detecting

tumors from MRI brain scan slices; however, though the agents successfully delineate the tumorous region from the surrounding soft tissue, the final segmentation is limited in accuracy and smoothness. White et. al. have developed an color image segmentation algorithm which used swarm intelligence to segment images based on similarities in color intensity [4]. The swarm consists of two types of specialized agents: scouts to find and label pixels similar in color, and workers to join these pixels into connected regions. Another swarm-based approach involves multiple populations competing to occupy a landscape of cells covering an image; labels were assigned to corresponding pixels depending on the member occupying the cell after the final generation [2]. Several further studies have examined the use of ant-based algorithms [15, 9, 3].

3.4 Application of Actual Entities

Chapter 2 briefly described the possible application of Actual Entities in general image analysis. A natural narrowing of focus would be their use in the delineation of anatomical structures in medical images. In such cases, an AE's "goal" might be to detect a specific tissue type or structure. The use of anatomical knowledge described in Section 3.3 could to a certain extent be incorporated into prehension functions.

As mentioned earlier in the chapter, the CT and MRI modalities generate scans in the form of slice sequences. These slices are 2-D images, and a 3-D volumetric representation can potentially be reconstructed from the full scan sequence. From there, it follows that there are two basic types of COPE to be considered.

3.4.1 Slice COPEs

A slice is represented in the form of a COPE with one AE per image pixel. For ease of reference, these will be referred to as Pixel AEs (PAEs). However, it important to note that a Pixel AE is not the same as a pixel; it is a separate unit, which takes on parameters inherited from the pixel at the time the COPE is first initialized. These parameters include location and gray-level intensity value, as well as a 'weight' parameter which increases or decreases as the Pixel AE prehends other objects in the COPE.

3.4.2 Scan COPEs

Since it consists as a series of slices, for the purpose of analysis a scan can be considered a series of Slice COPES structured as described in the previous subsection. However, within the context of a Scan COPE, a Slice COPE can itself also be considered as an AE. A Scan COPE can thus be considered both as a population of Slice COPEs, and as a population of the Pixel AEs that form the populations of these Slice COPE. In practice, the two are combined. This will be explained further in Chapter 5.

Note that although a Scan COPE aims to construct a 3-D volume using the structures detected in a series of Slice COPEs, it can be considered 3-D only in terms of visualization and physical location within a stack of slices. Each Scan AE has more than 3 parameters and therefore exists in a COPE of a higher dimension. The same is true for the Slice COPEs, which are considered 2-D only in the sense that their Pixel AEs have an x and y mapping to pixels in the original image.

CHAPTER 4 Slice COPEs

4.1 Structure

We now limit discussion of AEs and COPEs to the specific structures relevant to the present research, beginning with Slice COPEs. A Slice COPE can be visualized as a two-dimensional grid, containing as many rows and columns as the slice image under consideration. Each cell of the grid contains a single Pixel AE, as illustrated in figure 4.1. The COPE thus has the same number of PAEs as there are pixels in the slice image. As stated previously, it should be noted that while the PAEs are initialized using the pixel values at the corresponding positions in the image, they are not equivalent to the pixels themselves.

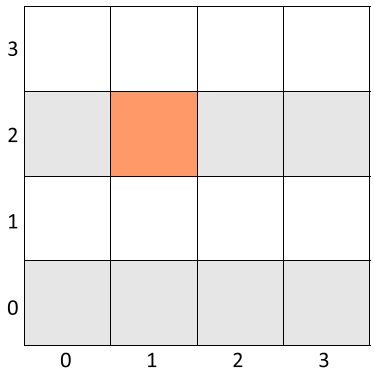


Figure 4.1: A view of a Slice COPE as a 2-D grid. The orange (or darker) cell is populated by the AE prehending the pixel at co-ordinates (1,2) in the slice image.

A Pixel AE can take one of two structures in a Slice COPE, depending on the color depth

of the image. If the image is 8-bit greyscale, an AE **P** is represented as follows:

$$\mathbf{P} =< x, y, g, w > \tag{6}$$

where x and y are the location of the PAE's corresponding pixel, g is the pixel's grey-level value, and w is a weight parameter. If the image is in color, **P** is represented as:

$$\mathbf{P} =< x, y, rgb, w > \tag{7}$$

where x, y, and w are as above, and rgb contains the color information for the initializing pixel. The choice of representation for RGB values could be handled as a set of integer components (red, green, blue), a color object, or some other form; the exact implementation does not significantly affect the Slice COPE's behavior.

Although all of a PAE's attributes can influence whether or not it prehends another PAE, only w is altered during prehension. A PAE has freedom of movement in no dimension other than weight; throughout its existence, it maintains the $< x, y >$ coordinates and the grey-level or color value of the pixel used to initialize it. Depending on the prehension function governing a PAE, it may only be able to increase in weight, or it may increase and decrease. The precise interpretation of w depends on the type of COPE, as will be explained later in this chapter.

A Pixel AE initialized using a greyscale image thus has four dimensions, and only moves in one. This is difficult to visualize, so Figure 4.2 restricts itself to the $x-$, $y-$, and $w-$ dimensions to offer an illustration of how the PAEs evolve. Assume the orange and blue boxes are PAEs in the same Slice COPE. Their positions on x- and y-axes map to the x and y coordinates of the pixels used to initialize them; these do not change during

prehension. Only their positions on the w-axis are altered, in response to the changes in their w (weight) parameters caused by positive prehensions. The orange PAE in fig. 4.2 has a higher weight value than the blue PAE. Also shown is the Euclidean distance between them on the x/y grid, which is typically used when measuring distance thresholds. However, treating w as a third axis allows the use of the 3-D Euclidean distance, which incorporates the difference in weights when deciding if a prehension is positive.

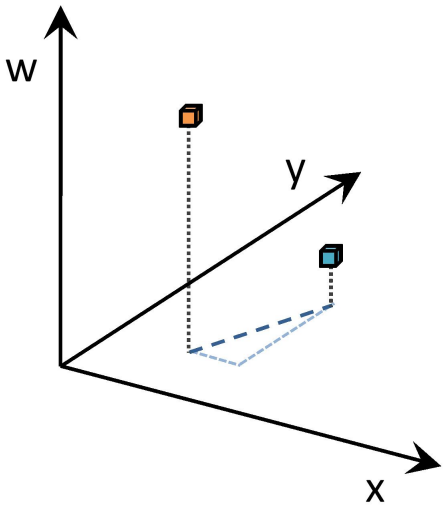


Figure 4.2: A 3-D graph view of two Slice COPE AEs. The w-axis refers to an AE's weight, which increases and decreases in response to its prehensions.

4.2 Prehension Masks

The use of thresholds in the prehension decision has been described in Section 2.3, along with a vector representation of attributes. Ideally, the Slice COPEs should offer flexibility in deciding which dimensions (parameters) of the AEs to consider and the resulting behavior, along with the thresholds used and potential weighting of differences. This is accomplished via the use of prehension masks.

Before progressing, several terms must be defined. Let T_i, T_o, T_g, and T_w respectively represent the minimum and maximum thresholds on distance and the thresholds on grey-level and weight differences. Although the idea of an inner distance threshold was not mentioned in 2.3, it can prove useful in a number of applications. For example, an inner threshold can be used to make a prehension negative if two AEs are too close in some relevant dimension(s). This can be used in data clustering to prevent a cluster of AEs from converging to a single point.

Considering a prehender Pr and a prehended object Pd, let D_d, D_g, and D_w represent the $x-y$ Euclidean distance, the grey-level difference and the weight difference between Pr and Pd respectively. These are calculated during each prehension. Finally, let p represent pixel units, used to express the $x-y$ distance between two Pixel AEs (a reference to how the distance would be calculated in the original image).

As stated, the natural formulation of an AE is as a vector of parameters. This offers the benefits of simple vector arithmetic. Consider the prehender $Pr =< d_i, g_i, w_i >$ prehending the object $Pd =< d_j, g_j, w_j >$ within the COPE environment described in Section 4.1. During prehension, a *current difference* vector CD representing the distance between Pr and Pd is constructed:

$$\mathbf{CD} = \begin{bmatrix} \sqrt{D_d} \\ \mathrm{D_g} \\ \mathrm{D_w} \end{bmatrix} = \begin{bmatrix} \sqrt{(x_i - x_j)^2 + (y_i - y_j)} \\ |\mathrm{g_i} - \mathrm{g_j}| \\ |\mathrm{w_i} - \mathrm{w_j}| \end{bmatrix} \tag{8}$$

Now consider an *attribute mask* **A** that specifies which parameters to incorporate into the prehension decision. In the proposed application, this is formulated as:

$$\mathbf{A} = \begin{bmatrix} \text{usedistance} \\ \text{usegrey} - \text{level} \\ \text{useweight} \end{bmatrix} \tag{9}$$

where each component acts as a binary 'switch' (0 = false, 1 = true). We would also like to introduce a *threshold mask* **T**, containing the threshold values in use for each parameter. This can be formulated as:

$$\mathbf{T} = \begin{bmatrix} -\mathrm{T_i} \\ \mathrm{T_o} \\ \mathrm{T_g} \\ \mathrm{T_w} \end{bmatrix} \tag{10}$$

where the components are as previously defined. Again, the reason for inverting the sign of T_i will be explained later. Note that T_i and T_o are both calculated only in the x- and y-dimensions. This does not strictly follow Primeaux's formulation proposed in Section 2.3, in that Euclidean distance is not calculated across all four dimensions. Instead it is aimed at the specific application of image analysis and mimics the approach of basic spatial segmentation. The threshold mask **T** can be multiplied with the attribute mask **A** to generate an *active threshold mask* **AT** for the current prehension. This requires including the distance switch twice in **A** to account for the separate T_i and T_o thresholds, but offers a flexible means to define the prehension function. Likewise, the distance must also be included twice in the **CD** vector, which is now formulated as $\mathbf{CD} =< -D_d, D_d, D_g, D_w >$. Again, the reason for inverting the sign of the first component will be explained later.

Consider the situation where $T_o = 60p$, $T_i = 5p$, $T_g = 17$, and $T_w = 30$, and the user

wishes the prehension function to incorporate only the outer distance and the grey-level difference. This creates the following active threshold mask **AT**:

$$\mathbf{AT} = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \end{bmatrix} \times \begin{bmatrix} 5 \\ 60 \\ 17 \\ 30 \end{bmatrix} = \begin{bmatrix} 0 \\ 60 \\ 17 \\ 0 \end{bmatrix} \tag{11}$$

The prehending PAE Pr must now examine whether its interaction with Pd meets the necessary thresholds for prehension. Suppose for our current Pr, $D_d = 50p$, $D_g = 13$, and $D_w = 55$. Pr calculates a *prehension test* mask **PT** as follows:

$$\mathbf{PT} = \mathbf{AT} - \mathbf{CD} * \mathbf{A} = \begin{bmatrix} 0 \\ 60 \\ 17 \\ 0 \end{bmatrix} - \begin{bmatrix} -50 \\ 50 \\ 13 \\ 55 \end{bmatrix} \times \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 10 \\ 4 \\ 0 \end{bmatrix} \tag{12}$$

A simple test is then applied to **PT**: if all components have values ≥ 0, then all thresholds are met and prehension can occur. This explains the need to invert the sign for T_i and to repeat the distance component in D_d. Unlike the other thresholds, all of which specify a maximum possible value, T_i specifies a minimum value. To standardize the test applied to the components, the following rule is used: if the threshold T_α is a minimum threshold then invert the sign of both T_α and the difference value D_α. As an example, consider $T_i = 5$ and $D_d = 25$. The corresponding value in **PT** is $(-5 - (-25)) = 20$, and thus the threshold is met and the prehension is positive. However, if $T_i = 2$ then the value in **PT**

is $(-5-(-2)) = -3$; in this case, the threshold is not met and the prehension is negative.

This explains the attribute and threshold masks. An optional extension is a *bias mask* **B**, which involves a fourth type of threshold T_B (the bias threshold). **B** exists independently of any other threshold. It provides a method to combine the differences between PAE attributes in various dimensions and test them under one threshold, while simultaneously controlling the impact of each attribute on prehension. Where n attributes are under consideration, the prehension test is performed as follows:

$$\sum_{i=1}^{n}(B[i] * CD[i]) \leq T_B \tag{13}$$

In other words, when Pr prehends Pd, the bias mask is multiplied with the current difference mask and the resulting elements are summed. If this is below the specified bias threshold, the resulting prehension is positive. In this study, **B** contains three components B_d, B_g and B_w, representing the distance bias, grey-level bias and weight bias respectively.

Note that T_B can be implemented at the same time as the other thresholds, or separately. The other thresholds offer hard limits on specific differences between the PAEs, whereas the use of T_B offers more flexibility in the precise contributions of each attribute.

4.3 Movement after Prehension

If a prehension is positive, then Pr's w-value can increase, decrease, or move toward Pd's w value, depending on the particular COPE type. The degree of this movement can be specified in two ways. It can be a fixed 'step' M_s, implemented as either a constant number or a percentage of the maximum possible value of w (100 in this study). It can

also be a scaled version of this fixed step, where a scaling factor M_{sf} is applied to the sum of normalized differences of the attributes considered during prehension. The second option opens up the possibility for non-linear movement, for example by replacing the sum operator with exponentiation.

4.3.1 Penalties

The discussion of Pixel AE movement leads to the idea of penalties. These are defined as movements which cause an PAE's weight w to decrease and/or cause it to move away from Pd. Three types of penalty can be imposed, all of which are designed for this specific application. Penalty $P1$ applies when the distance between Pr and Pd exceeds the outer threshold T_o. Penalty $P2$ applies to PAEs that fall within T_o but exceed the weight and/or grey-level thresholds T_w and T_g. Finally, penalty $P3$ relates specifically to the T_B threshold and is thus applied independently of $P1$ and $P2$. The penalties are implemented as weights applied to the two methods of movement outlined in Section 4.3. As an example, if $P1 = 0.5$ and a fixed step M_f is used, then a penalized AE will move $0.5 * M_f$ units along the w-axis.

Penalties remove the possibility of negative prehension from a COPE. Remember that positive prehension refers only to an AE altering itself in response, and does not specify the nature of this alteration. Negative prehensions reflect the absence of change. While a penalty can be seen as involving a negative outcome for the prehending PAE - in the sense that it moves away from the prehended object and/or reduces its weight - it is in fact simply a positive prehension that results in repulsion rather than attraction. Negative prehensions therefore only occur in Slice COPEs which do not impose penalties.

4.4 Basic Greyscale COPE

This thesis focuses on segmentation of greyscale images, since these are standard in both the MRI and CT modalities. Three types of COPE are proposed, each of which offers different possibilities for segmentation and causes its population of Pixel AEs to exhibit different patterns of emergent behavior. The standard form is the Basic Greyscale COPE (referred to as BGC). A BGC consists only of standard PAEs, all with equal prominence. There is no method for a user to give preference to any particular PAE. It is therefore anticipated that this COPE will not perform well in delineating structures, but it forms the base framework for the other types of COPE and may demonstrate some interesting behavior. As described in Section 4.1, the COPE consists of a grid of PAEs initialized using pixels in the corresponding slice image. Their weights (w parameter) can be initialized at zero, or randomized. In the latter case, penalties must be imposed to obtain meaningful behavior.

4.4.1 Selection Functions

As stated in chapter 2, a COPE requires a selection function in order to choose a prehender - and, in some situations, the range of objects that can be prehended. (Note that the prehended object is not directly specified, since this precludes emergent behavior). The BGC offers three selection functions:

1. **Every other at random**: A random PAE Pr is selected from the COPE and randomly prehends every other PAE in the COPE (each being prehended once only). The PAE's response is presented to the COPE between prehensions. This follows

the standard selection approach of Chapter 2.

2. **Every other in order**: A random PAE Pr is selected from the COPE and randomly prehends every other PAE except itself, starting from position $[0, 0]$ in the 2-D AE array. The PAE response is presented to the COPE between prehensions.

3. **Single at random**: Two AEs Pr and Pd are selected at random from the COPE and one prehends the other.

These functions are replicated in the Seeded and Distinguished Greyscale COPEs which will be defined later in this chapter.

By immediately returning Pr's prehension response to the rest of the COPE, prehension p_i is performed using Pr's state after the prehension p_{i-1}. Another approach would be for Pr to change states only after prehending all other AEs, with the change being calculated based on the combined prehensions. However, this would create issues with 3-D visualization and is thus not pursued in this study.

In a COPE of reasonable size, the 'Every other at random' selection function performs slowly on a typical desktop computer. The scan images under consideration in this study have been resized to (256*256) pixels, creating Slice COPEs populated by 65536 AE objects. Even if a tally is kept of which PAEs in the COPE Pr has already prehended, the process still involves repeated generation of random $x - y$ positions. This was addressed previously in Section 2.2 and illustrates the benefits of switching to a selection function that has the randomly selected PAE prehend only one other entity.

4.5 Distinguished Greyscale COPE

The Distinguished Greyscale COPE (DGC) introduces a second type of object: the Distinguished Actual Entity (DAE). A DAE is identical in structure to a PAE, with the exception that it does not prehend any other object and it is assigned maximum starting weight on the w-axis. It acts as a fixed reference point to which similar PAEs are attracted, where similarity is measured based on some combination of attributes and thresholds. In other words, it can be considered an 'ideal' or goal to which a standard AE aspires. This echoes the sub-image recognition application described in Section 2.5. In the application of image segmentation, a DAE acts as a means to denote the structure of interest in the image. A user can place multiple DAEs within this structure in locations that best reflect its visual characteristics. These DAEs will exert strong influence over the evolution of the COPE, which in theory will successfully delineate the desired structure from its background.

4.5.1 Selection Functions

Along with the standard selection functions of the BGC, the DGC offers two additional choices:

1. **All objects**: A random Pixel AE Pr is selected from the COPE and prehends some random object in the COPE (alternating between prehension of another PAE or a DAE)

2. **DAEs only**: A random PAE Pr is selected from the COPE and prehends a random member of a set of DAEs (which was previously specified by the user).

The first option alternates between DAE prehension and PAE prehension - in other words, whenever a prehending PAE Pr attempts to prehend another object, the COPE records this activity. If Pr's most recent prehension (positive or negative) was of a DAE, it will next prehend another standard PAE, and vice versa. This allows more possibility for interesting behavior. However, depending on the thresholds used and the size of the original image, the COPE may evolve slowly; this is due to the increased potential of prehensions that do not involve PAEs related to the structure of interest. Note that the balance of DAE to PAE prehensions could be altered to place stronger or weaker emphasis on the specified DAE set.

The second option subverts the principle of random selection but can still offer interesting behavior with a sufficiently large number of DAEs. It can also be extended to the type of COPE explained in the next section.

4.6 Seeded Greyscale COPE

Consider a DGC using the 'DAEs only' selection function - with the difference that, after an PAE Pr positively prehends a DAE D, Pr also becomes prehendable. If another PAE then positively prehends Pr, it too becomes prehendable. This approach offers a middle ground between the two DGC-specific selection functions. The COPE is not restricted to a fixed set of prehendable entities, but prehensions between PAEs which are not related to the structure are less likely to occur. Growth effectively spirals out from the starting list of DAEs, since only those PAEs sharing a high degree of similarity become prehendable objects. This leads to the concept of a Seeded Greyscale COPE (SGC)

The initialization of an SGC is similar to that of a DGC. The user chooses a number of

seed pixels in the slice image, which are used to create Seed AEs (SAEs). Only these SAEs can be prehended at the start of COPE evolution, but over time this grows to include all PAEs sharing a high degree of similarity. The influence of a SAE thus decreases as the COPE evolves, as opposed to the fixed 'ideal' of a DAE. The other crucial difference to a DGC is that, unlike a DAE, a SAE can prehend other SAEs and PAEs and thus alter its own weight.

4.6.1 Selection Functions

The SGC has the standard BGC selection functions, but provides an additional function **'Random prehendable'** which incorporates the idea of a starting Seed AE list. The COPE's initialization creates a list *prehendables* containing those objects which can be prehended. This list is initialized with the user-selected SAEs.

An prehending PAE (or SAE) Pr is selected at random from the COPE, and selects a random member S of the prehendables list for prehension. If the resulting prehension is positive - i.e. Pr is sufficiently similar to S - then Pr is added to the list of prehendable AEs. If the prehension is negative, then the list is unchanged. This process repeats, gradually increasing the length of the prehendables list. The SAEs thus exert influence over the COPE primarily in its early stages, since they are the only objects which can be prehended. As an example, consider that the chosen prehension function uses a distance threshold to decide whether a prehension is positive. In the period immediately following the COPE's initialization, only those entities which are within this specified distance of a SAE will achieve positive prehension, but over time this region will grow. (This necessitates the use of the T_g or T_B thresholds to achieve meaningful behavior.)

CHAPTER 5 Scan COPEs

5.1 Structure

Along with segmentation of tissues and structures from individual 2-D slices, this study is also interested in volumetric reconstruction of 3-D objects identified across the entire scan. As stated, the means to do this already exists at the time of imaging, but requires dense scans with a high number of images. This chapter will explore an approach that circumvents this issue through inter-slice interpolation, achieved via the emergent behavior of AEs.

A Scan COPE actually consists of a number of Slice COPEs, each of which has the option to prehend each other entirely. In this manner, the Slice COPEs function as AEs in themselves. However, the Pixel AEs contained within each Slice COPE can prehend those in other slices. Therefore, for computational simplicity, the Scan COPE is represented as a 3-D array of objects in the $x-$, $y-$, and $z-$ dimensions (z being the vertical dimension associated with stacking the slices one on top of another). However, each Pixel AE remains tied to its slice, and the incorporation of multiple prehension functions allows variations such as one entire Slice COPE prehending another, and a Pixel AE within one Slice COPE prehending all those in another Slice COPE.

Based on this, the most intuitive form of a Scan COPE AE (SCAE) P might be as follows:

$$P =< x, y, z, a, w, g > \tag{14}$$

where x, y and z represent its fixed position in the 3-D 'stack' of slices, w and g are the weight and grey-level parameters used in the Slice COPE, and a is an *activation* parameter. P consequently has two weight parameters: w, which measures its 'nearness' (or estimated degree of fit) to the structure of interest in an individual slice, and a, which measures its 'nearness' to the desired 3-D representation.

However, in the proposed implementation, an SCAE P takes the following form:

$$P =< x, y, z, a, > \tag{15}$$

In this case, the parallels between the a attribute of a SCAE and the w attribute of a PAE are even stronger, in that they represent the only dimension of movement for their respective entities. This formulation depends on prior separate processing of each scan slice - i.e. that a Slice COPE is formed for each individual slice image to extract the structure of interest. This COPE is then stored in a file as a 2-D array containing the final w values for all Pixel AEs in its population. A Scan COPE is generated by reading a series of such files and generating a Slice Object from each. Although this implementation is simplified and comes at the cost of a wider range of SCAE movement within the Scan COPE, it is effective and computationally efficient. The intuitive representation given in Eqn. 14 would result in a COPE with high computational complexity. For a scan taken at a resolution higher than the (256×256) used in this study and containing a reasonable number of slices, it is likely to prove impractical without introducing parallel computation. An interesting area for future study would be the construction of Scan COPEs in a distributed computational environment with the processing power required by the additional dimensions of movement.

The chosen implementation using the AE structure proposed in Eqn. 15 relies strongly on the Slice Objects mentioned above. These are generated from the stored output of Slice COPEs initialized using actual slice images. There is still the issue of interpolation between these in order to create a 3-D volume of the structure of interest. This leads to the definition of 'Real' and 'Virtual' Slice Objects provided in the next section.

5.2 Real and Virtual Slice Objects

A Real Slice Object (RSO) is defined as the output of Slice COPE analysis applied to an actual slice image from the CT or MRI scan set - i.e. the 2-D array of weights stored in a file and read during creation of the Scan COPE. However, a RSO is not the same as a Slice COPE. First, not all x and y positions in a RSO contain AEs (whereas a Slice COPE maps an AE to each of the pixels in the image under analysis). The population is decided when the stored file is first read, by applying a threshold T_w to the weight values in the 2-D array. Only those positions with weight $w > T_w$ will be populated, as these are likely to correspond to the structure of interest. All other positions in the RSO are left empty. The second and more crucial difference is that a RSO is populated entirely by Distinguished AEs which do not prehend any other object and therefore do not change. This causes the RSO to become a DAE in itself, in that it can be prehended by other Slice Objects and non-distinguished AEs but cannot prehend. Given that the aim is to interpolate between existing slices, this is a reasonable choice of implementation.

A Virtual Slice Object (VSO) has no correspondence with any slice image. It should not be considered an empty slice, since it contains initialized AEs; however, all of these AEs have an activation level of $a = 0$. Interpolation occurs as the AEs prehend other

objects and alter their a parameter in response. Initially, AEs in a VSO can only prehend the DAEs in a RSO (or the RSO itself). However, when their activation level exceeds a threshold T_a, the AEs are considered activated, and can then be prehended by other AEs in the Scan COPE. A hard threshold is chosen in order to preserve firm edges of the 3-D volume, but if the method is refined further then use of a fuzzy boundary may be more useful in achieving a smoother final structure.

A VSO can be implemented in two forms. The first mimics that of a Slice COPE - a 2-D grid with the same number of rows and columns as the slice image, containing an AE in each cell. In theory this allows for a wider range of behavior by virtue of having a larger population. However, in practice it is arguably redundant. The aim is to interpolate a 3-D representation of a specified structure identified in the separate slice images; therefore, the region of interest in a VSO is focused on the approximate location of the structure in the slices corresponding to the nearby RSOs. This typically results in more rapid COPE evolution by avoiding prehensions that involve an AE far away from the structure's location.

As stated, a Scan COPE is constructed as a 3-D array, which can also be considered a 2-D stack of slices. The user can specify how many VSOs should be placed between each RSO; more VSOs leads to a higher level of computational complexity but allows a finer degree of interpolation between slices and has the potential to create smoother volumes. Figure 5.1 illustrates how the real and virtual slices are stacked during creation of a 3-D Scan COPE.

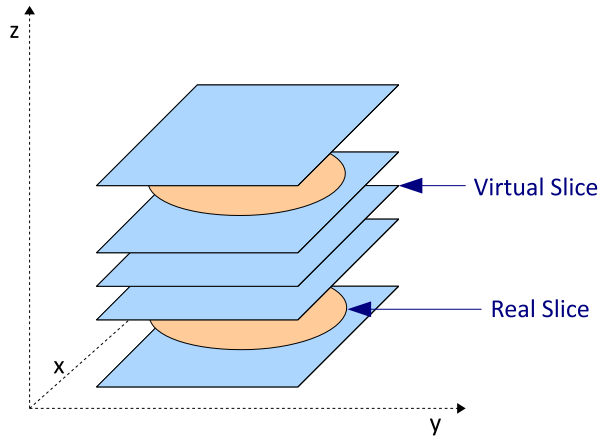


Figure 5.1: A stack of real (orange) and virtual (blue) scan slices.

5.3 Prehension

Prehension occurs primarily based on x, y, and z positions. Although the use of 3-D Euclidean distance is an option, a more successful method may be to consider the $x-y$ and z dimensions according to separate thresholds, or apply a bias that allows a wider range of movement along the $z-$ axis (i.e. interpolation across the stack). The process of attribute selection and thresholding is similar to that outlined for the Slice COPEs in Chapter 4.

5.4 Selection Functions

The SCAEs in a Scan COPE focus their prehensions on the DAEs contained in the RSOs, as these outline the structure of the desired object and provide an interpolation frame. This gives rise to a number of possible selection functions:

1. **DAEs only:** A random SCAE Pr is selected from a random VSO and prehends a random DAE D in a random RSO.

2. **Nearest DAEs only:** A random SCAE Pr is selected from a random VSO and prehends a random DAE D in the nearest RSO (calculated in terms of distance along the z-axis).

3. **Random RSO:** A random SCAE Pr is selected from a random VSO and prehends a random RSO (by prehending every DAE contained within).

4. **Random DAE or AE:** A random SCAE Pr is selected from a random VSO and prehends a random object from a random RSO or VSO.

5. **Prehendables only:** A random SCAE Pr is selected from a random VSO and prehends a random member of the prehendables list

'Prehendables only' is a variation on the selection function used by the SGC, as described in Section 4.6.1. A list **prehendables** is created when the Scan COPE is initialized and populated with the DAEs in the RSOs. As the SCAEs in the VSOs positively prehend the DAEs, their activation levels increase. When a VSO AE becomes activated, it is added to the list of prehendables. This encourages prehension between VSOs, which is particularly useful when the number of VSOs heavily outweighs the number of RSOs. It also circumvents the slowness of the 'Random DAE or AE' selection function by avoiding redundant prehensions involving VSO SCAEs which are a long distance (in $x - y$ space) from the structure of interest.

CHAPTER 6 User Interface and Imaging

This chapter will briefly examine the user interface to the COPE image analysis program. The program itself is not the focus of this thesis, but the features it offers for observing COPE behavior are worth mentioning. The most obvious benefit is the ability to view the emergent behavior of AEs in dynamic 3-D form, using the x-, y- and w- axes previously described. There is also a standard static analysis procedure, the output of which can be viewed on the 3-D axes or in 2-D image form.

The user first chooses from a menu the type of COPE he or she wants to analyze: the Basic, Seeded, or Distinguished Slice COPE or the Scan COPE. To provide an example, assume the user selects the DGC option. This takes them to the main analysis panel shown in figure 6.1.

Here, they can choose to set DGC's parameters, perform static analysis without visualization, view a static 2-D or 3-D snapshot of the its status, perform dynamic 3-D analysis, or export its current status to an output text file ready to be used in creating a Scan COPE. The image under analysis appears at the top of the panel. The two green dots represent Distinguished AEs assigned by the user.

Figure 6.2 shows the parameters dialog box for a DGC cope. Much of this is self-explanatory, but one aspect worth noting is that the program allows changes to be made in this panel while the dynamic 3-D visualization of the COPE analysis is running. This enables on-the-fly alteration of parameters, which proves useful in exploring new types of

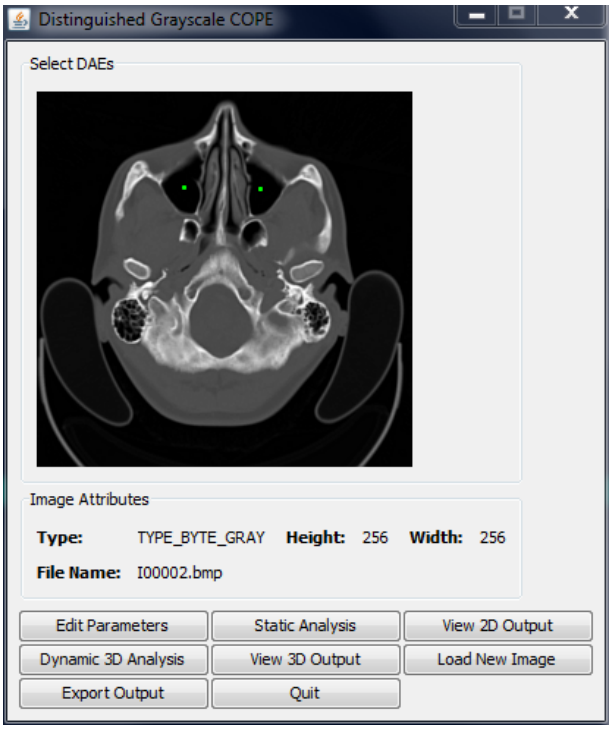


Figure 6.1: The main analysis box for a DGC COPE.

behavior. For example, the user could alter the prehension thresholds toward the later stages of a Slice COPE's evolution, or switch to a different selection function to influence the populating AEs in a different direction.

A 2-D view of the Slice COPE under analysis can be called up at any point, including during dynamic 3-D visualization. This acts as a snapshot of its current status. It is created by duplicating the image under analysis, but altering the pixel colors to reflect the current weight of each AE (PAE or SAE). This use of colors is common throughout all visualization options, and can be calculated in several ways. First, the user can specify a number of 'levels' of color, dividing the w-axis into multiple strata. For instance, a w-axis with a maximum value of 100 might contain 10 levels; in terms of visualization, an AE

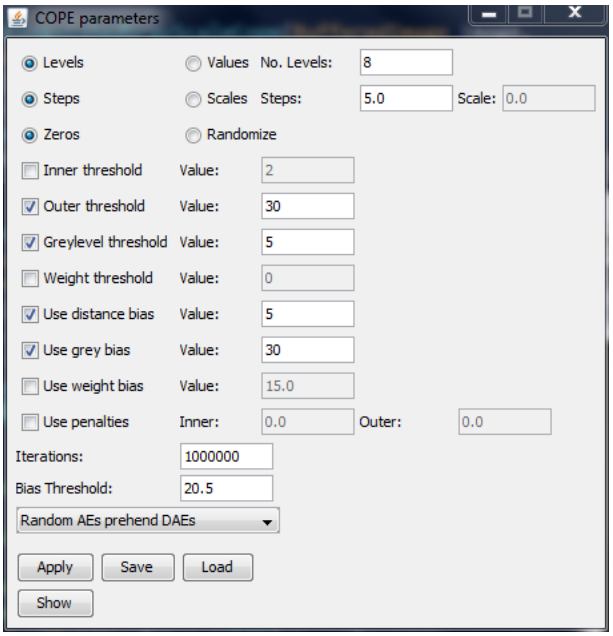


Figure 6.2: The parameters dialog box for a DGC COPE.

with weight $w = 2$ would appear the same as an AE with weight $w = 8$, until either one crossed the boundary to the next level. When using levels, the colors are generated at random (with the exception of the default option of 8 levels, which has a fixed palette). The second option is to base an AE's color purely on its weight, without the use of levels. This is done by scaling.

Figure 6.3 shows an example snapshot. This was taken during analysis of the image shown in figure 6.1, where the user has marked the left and right nasal cavities as the areas they wish to segment.

Finally, figure 6.4 shows the 3-D visualization option. This is implemented in OpenGL and the user has control over the viewing angle; they can zoom in and out, rotate the axes and move the camera. The display updates at regular intervals defined by the number of prehensions (positive or negative) that have occurred, and the user can output a snapshot

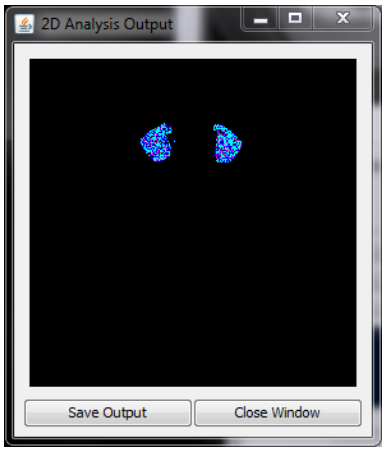


Figure 6.3: A 2-D snapshot of a DGC COPE.

of the 3-D view to an image file at any point. The use of colors is identical to that in the 2-D snapshot creation method described above; they represent the current weight (w parameter) of a given AE.

The benefit of this visualization option is that it enables the user to both observe the behavior that emerges at each stage of the Slice COPE's evolution, and also alter parameters to influence that evolution. Static analysis does not demand as much computational power and is therefore much faster - depending on the prehension function, five hundred thousand prehensions can be completed in 10-30 seconds on a standard desktop computer. However, dynamic visualization is necessary in order to fully characterize the behavior of the COPE.

After performing COPE analysis on a series of slices, the user can assemble these into a Scan COPE. The front end is shown in figure 6.5, including the parameters panel.

Scan analysis can be performed invisibly ("Static analysis"), and the results viewed in 3-D form. Alternatively, the evolution of the Scan COPE can be visualized in 3-D as it

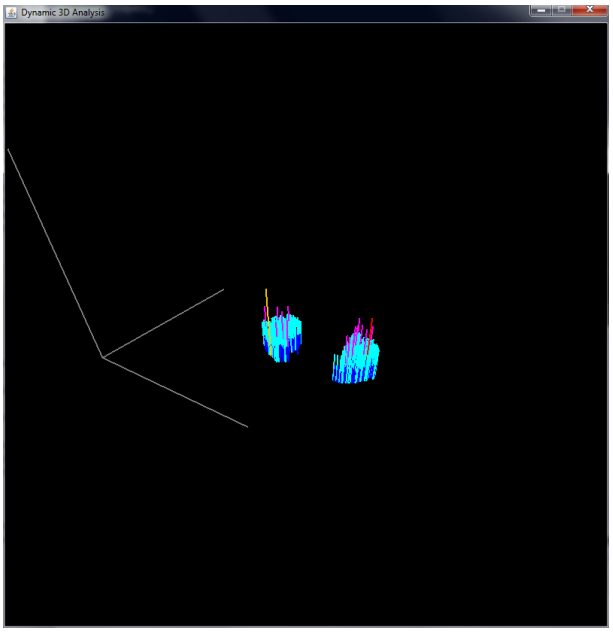


Figure 6.4: 3-D visualization of a DGC COPE.

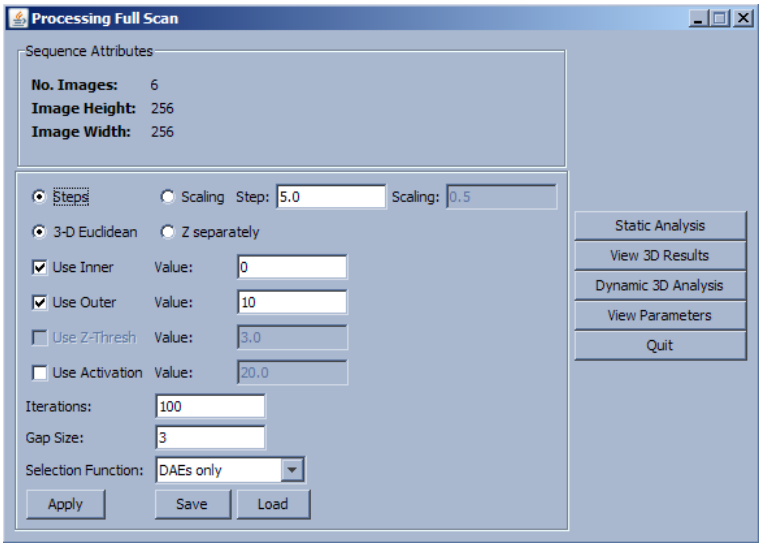
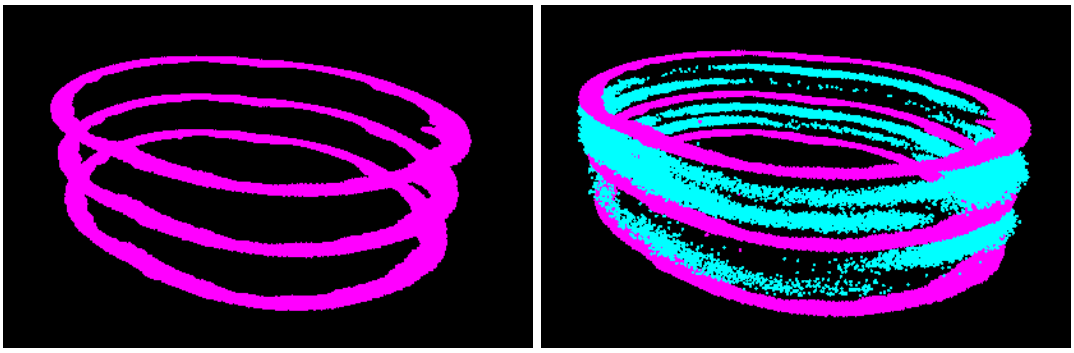


Figure 6.5: The main analysis box for a Scan COPE.



(a) COPE at initialization (b) COPE during evolution

Figure 6.6: 3-D visualization of a simple Scan COPE

occurs. Figure 6.6 provides a simple example demonstrating the slice stack described in Chapter 5. The pink pixels correspond to DAEs in the Real Slice Objects, and the blue pixels to activated AEs in the Virtual Slice Objects. This is a simple example; in practice, the arrangement of DAEs in the real slices will differ according to change in appearance of the desired structure(s) through the image series of the MRI or CT scan. Figure 6(a) and 6(b) respectively show the COPE at initialization and as it interpolates the virtual slices.

CHAPTER 7 Experimental Results and Observations

7.1 Slice Processing

7.1.1 Experiment 1: BGC analysis

The Basic Greyscale Cope (BGC) gave some unexpected results. The initial suspicion was that it would behave similarly to a clustering technique such as k-means. However, as figures 7.1 and 7.2 illustrate, this was not the case.

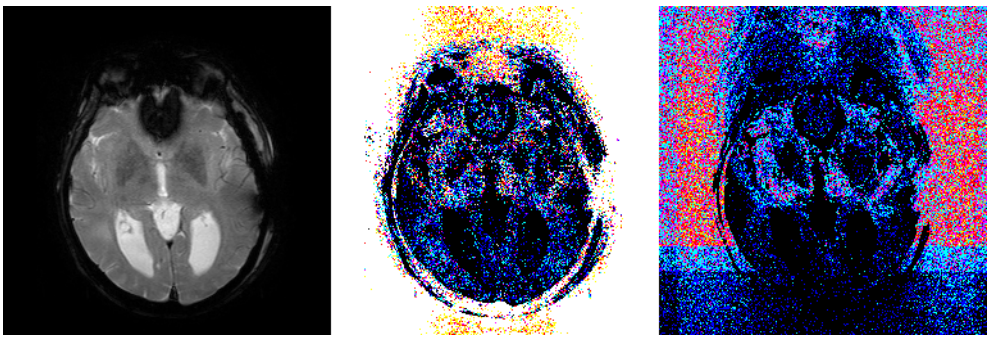


Figure 7.1: Unexpected results obtained using a BGC with grey-level and distance thresholds.

Further experimentation revealed that the BGC converges toward the dominant grey-level among its population. Figure 7.3 demonstrates this - the dominant color in the slice image is black, and therefore the PAEs with the lowest grey-level attribute values move up the w-axis. Altering the parameters does little to change this, with the exception of drastic penalties or extremely generous thresholds (which cause all PAEs to converge the minimum and maximum possible values of w respectively). With further development, the BGC may have potential use as a means of edge detection.

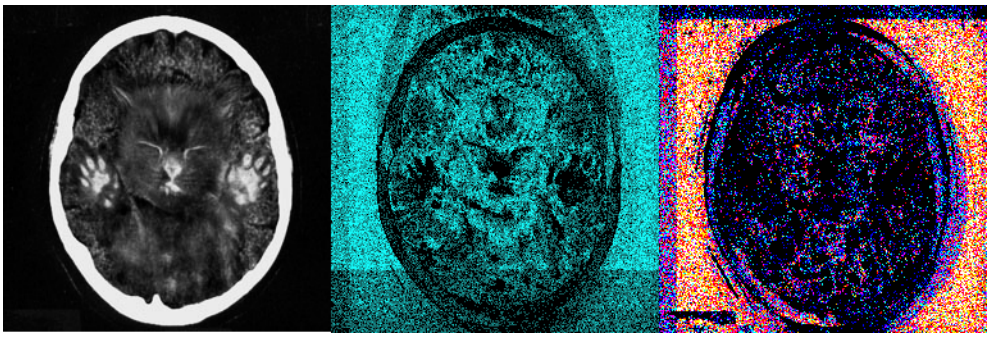


Figure 7.2: Unexpected results obtained using a BGC with a grey-level threshold and distance bias.

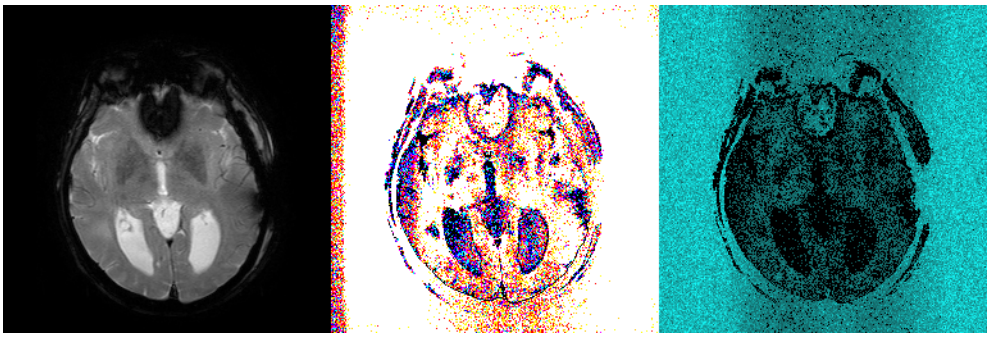


Figure 7.3: The AEs in a BGC converging to the dominant grey-level value

7.1.2 Experiment 2: Impact of SGC vs. DGC in skull segmentation

The important distinction between SGCs and DGCs is that seeds encourage growth from a starting point, after which any AE that positively prehends a seed effectively takes on the same role. No other special distinction is made between a seed AE and a regular PAE. By contrast, an PAE in a DGC cannot become distinguished; the list is specified by the user when the COPE is first initialized and remains static throughout evolution.

Experiments showed that SAEs proved particularly capable at extracting large objects from images. Figure 7.4 provides an example: the bone matter is extracted from a slice of a cranial MRI scan via the placement of four seeds. DGCs did not perform as well on this task, probably owing to the nature of their selection functions. The function which allows PAEs to prehend both DAEs and other PAEs evolved very slowly and led to considerable fragmentation of the detected structure, as opposed to the SGC which converged far more quickly and provides a clear delineation of the skull wall. The alternate DGC selection function, which allows PAEs to only prehend DAEs, evolved quickly but required a much higher number of starting DAEs to successfully detect large structures. This is due to the thresholds involved. Detecting a large structure using a small number of DAEs (e.g. 4 or 5) necessitates the application of a high outer distance threshold T_o. In the example slice image shown in Figure 7.4, setting T_o to a sufficiently high value caused the area in the center of the skull to also be segmented, as it possesses a similar grey-level value to bone matter. Bias measures could not solve the problem, as they prevented the far reaches of the skull wall from being detected.

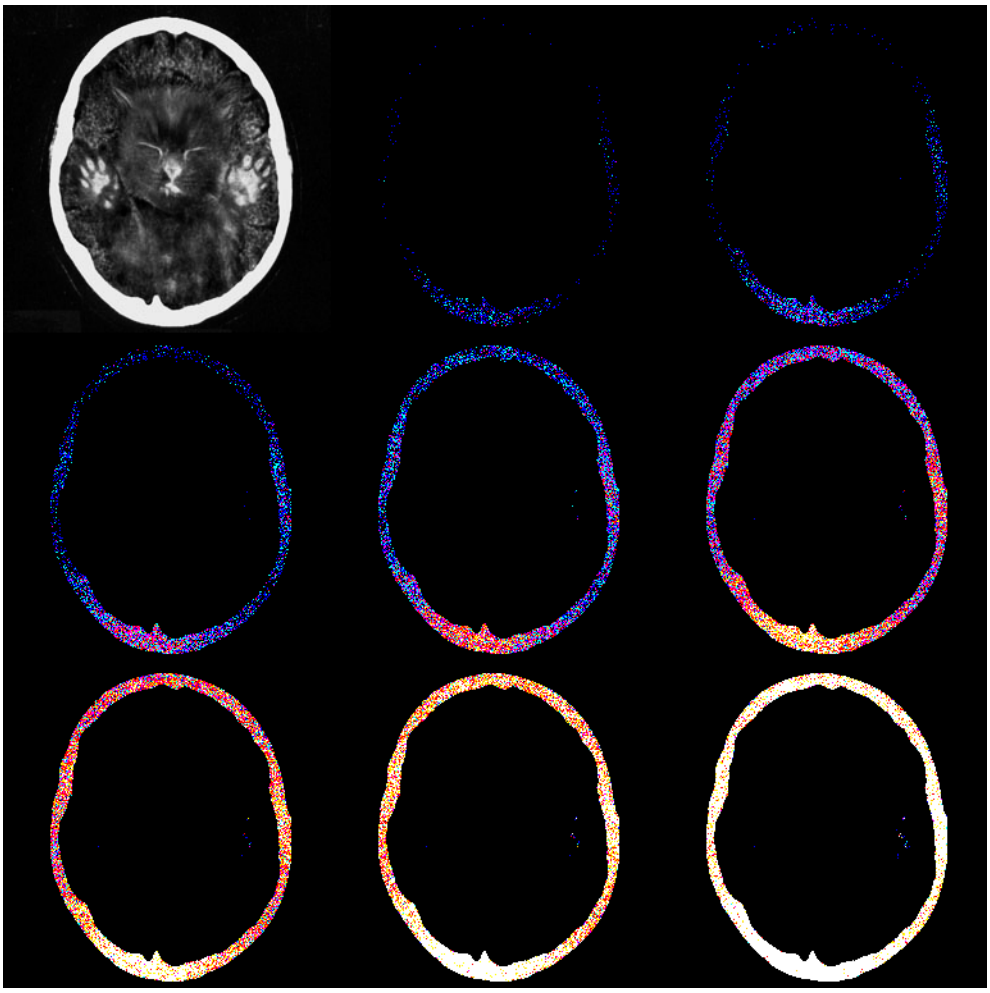


Figure 7.4: Extraction of bone matter from a brain MRI slice using a four-seed SGC.

7.1.3 Experiment 3: Impact of SGC vs. DGC in lung tumor extraction

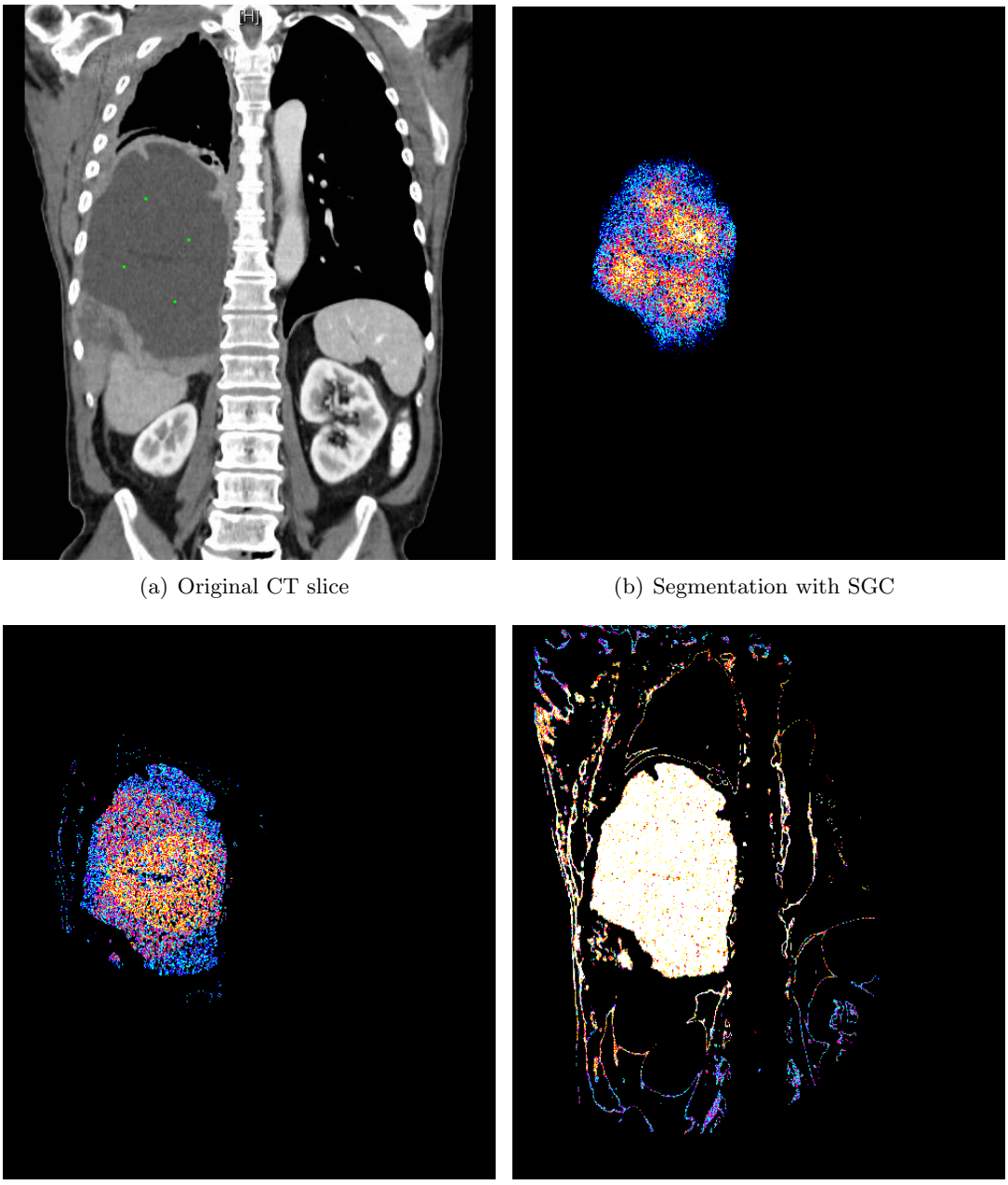
Figure 7.5 presents another example where an SGC arguably outperforms a DGC in segmentation. Again, the structure to be segmented (a lung mesothelioma) occupies a relatively large percentage of the image area. In both cases four starting points were specified as seeds or distinguished AEs; the green dots in Fig. 5(a) show how these points were initially placed. As can be seen in Fig. 5(b), SGC analysis offered more precise segmentation when applied with restrictive thresholds on distance and grey-level difference. While AE weights vary within the tumor region, the weights of those falling outside the region remain close to zero. However, using the same threshold parameters to initialize a DGC COPE resulted in four separate clusters of high-weight PAEs, where the size of each cluster matched the specified outer distance threshold. This behavior does not achieve the desired segmentation goal. Analysis was therefore repeated with the outer threshold value doubled. The results are presented in Fig. 5(c), and show a significant level of noise outside the structure of interest. This is due to the grey-level similarity between the tumor region and portions of the surrounding area. Incorporating bias measures and penalties did not significantly improve performance. However, the outer edge of the tumor is more clearly defined than in Fig. 5(b). DGC analysis was then conducted again with much weaker restrictions; the distance threshold was increased by 50%, and the grey-level threshold was doubled. This was offset by the inclusion of bias measures that increased the impact of grey-level difference on prehension. Despite this, the segmentation results were poor. As Fig. 5(d) shows, while the more generous thresholds provided clear definition of the tumor region, multiple other structures were detected. In particular, the behavior of the

PAEs in the DGC COPE appears to have produced edge detection. One explanation is that the low resolution of CT imaging causes blurred edges in the resulting scan image. In a transition between low and high greylevel values, there will be pixels possessing a similar greylevel value to that of the tumor region. The corresponding AEs will therefore positively prehend the DAEs associated with the tumor.

However, in this application SGC analysis has a major drawback - namely, processing time. Standard scan slices of 256×256 pixels can be processed relatively quickly using both DGC and SGC COPEs. However, the SGC COPE processing time increases significantly for larger images. For the lung mesothelioma detection task, the size of the original image is 496×557, creating a total of $276,272$ AEs in the COPE. The average analysis time using a four seed SGC and strict thresholds was over 10 minutes.

One possible cause is the large population of AEs in the COPE and the high percentage of negative prehensions that consequently occur. This is the case regardless of whether the 'prehenders only' or 'prehendables only' selection function is used. The only difference observed so far is that the prehension list grows slightly faster when using the 'prehenders only' function compared to 'prehendables only'; however, the AEs associated with the structure of interest take longer to reach the maximum permitted value of w. Neither behavior is desirable. A potential solution is to use more generous thresholds and so decrease the number of negative prehensions, but this can cause AEs outside the structure of interest to be included in the results of segmentation.

However, the DGC analysis tasks which produced results similar to those in Figs. 5(c) and 5(d) intuitively involve many more negative prehensions, as the list of prehendable



(a) Original CT slice

(b) Segmentation with SGC

(c) Strict segmentation with DGC

(d) Weak segmentation with DGC

Figure 7.5: Detection of lung mesothelioma in a coronal CAP CT slice.

objects remains fixed. It would therefore be expected that DGC analysis time would exceed that of SGC. Instead, segmentation of the tumor using strict thresholds had an average processing time of 22 seconds across five runs, while the same task using weaker thresholds took an average of 12 seconds. The flexibility of the SGC prehension list comes at additional computational cost - and it may be that a large population of AEs causes these lists to become unwieldy. The 'prehendables' and 'prehenders' lists must be continually updated and retained in memory. The structure of interest in Fig. 7.5 is relatively large compared to the image size, occupying approximately 9-10% of the total area. At the end of COPE evolution, the SGC list can be expected to contain around 25000 SAEs. Meanwhile, the execution time of a single prehension is too low to be easily recorded and can be assumed to be measured in milliseconds. This suggests that a high percentage of negative prehensions has much less effect on the COPE's performance than the costs of maintaining a prehension list.

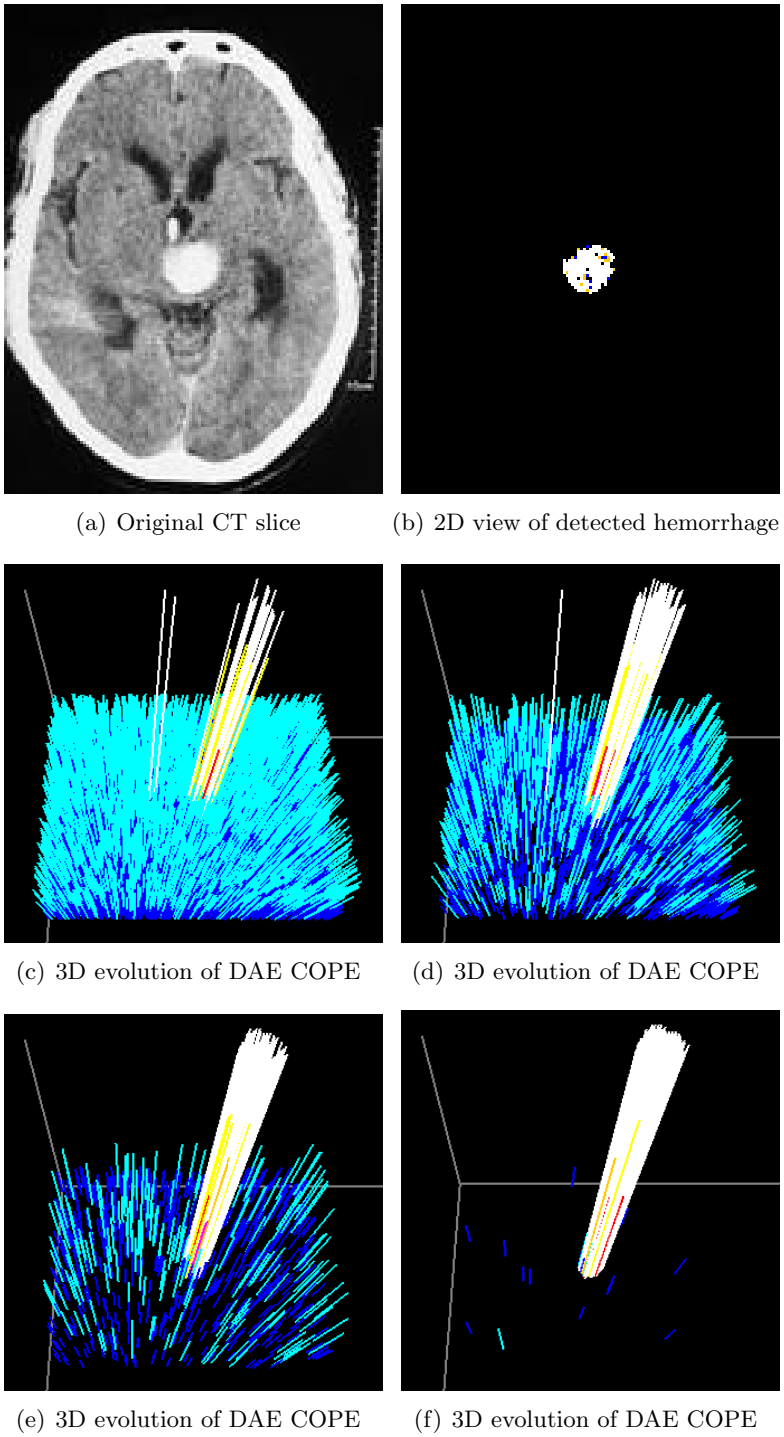
7.1.4 Experiment 4: DGC analysis in brain hemorrhage extraction

In keeping with the observations of Experiment 3, DGCs seem to be particularly suited to detection of small objects. An example is shown in Fig. 7.6, which illustrates the evolution of a DGC when detecting intra-cerebral hemorrhage in a slice taken from a CT scan of the brain. The original cropped slice image is relatively small (129×167 pixels) and the hemorrhage region occupies approximately 4.2% of the total area. The starting weights of the PAEs were randomized to allow better visualization of the COPE's rapid convergence in Figs. 6(c) through 6(f), with penalties being imposed on AEs that did not meet the distance and grey-level thresholds. These decreased w by $2x$ and x respectively, where x

represents the increase in w which occurred in an AE that met both thresholds. The images show that the PAEs corresponding to the hemorrhage region rapidly moved toward the maximum possible weight. The other PAEs reduced their weights to zero at a slower pace, but the processing time required to achieve the result shown in Fig. 6(b) was still only 15.6 seconds. These results along with other experiments suggest that DGCs are best suited to segmentation of small structures, particularly those with a relatively large difference in grey-level value compared to their surrounding area. This view is supported by the results presented in Figs. 5(c) and 5(d) which show the reverse situation - i.e., segmentation of a large structure with a similar average grey-level value to its surroundings, in which DGC analysis produces poor results.

7.1.5 Impact of Selection Functions

As stated, there are two selection functions specific to Seeded Greyscale COPEs. If penalties are imposed, 'prehendables only' offers a significant advantage. Any AE in the COPE is permitted to prehend any member of the 'prehendable' list - which at COPE initialization contains only the Seed AEs. This maintains the dominance of the SAEs by reducing the likelihood that they will be selected to prehend another AE and incur a penalty by failing to meet the required thresholds. The 'prehenders only' function reverses this situation; at initialization, only the Seed AEs are allowed to prehend. If an SAE prehends some dissimilar AE (in terms of either greylevel or distance), its w parameter will be decreased. This affects the entire evolution of the COPE. One way to circumvent this would be to make SAEs immune to penalties. However, it should be noted that this would decrease the impact of penalty measures as the 'prehenders' list increases in size.



(a) Original CT slice (b) 2D view of detected hemorrhage

(c) 3D evolution of DAE COPE (d) 3D evolution of DAE COPE

(e) 3D evolution of DAE COPE (f) 3D evolution of DAE COPE

Figure 7.6: Detection of hemorrhage region in axial brain CT slice.

7.1.6 Impact of Prehension Parameters

The impact of prehension function within each experiment has already been described, but a few general trends are clear. In particular, penalties have a significant effect on COPE behavior, and appear most beneficial when applied to DGCs with more than one assigned DAE. In this case, the dominance of the DAEs is pre-established, so penalties help in reducing positive prehensions that do not help in accomplishing the COPE goal. A PAE outside the region of interest may happen to positively prehend a given DAE in a manner that causes its w parameter to increase. If penalties are enforced within the COPE, this increase is likely to be negated when the PAE positively prehends another DAE to which it is not sufficiently similar. However, this creates additional complexities in parameter selection, since inappropriate penalties may prevent the COPE from reaching its goal or at least slow its progress.

Both distance and grey-level constraints are important in segmentation tasks, but the latter is particularly significant when AEs in neighboring regions share similar intensity values. In this case, use of the bias measure typically leads to faster and more accurate convergence. It allows for a finer degree of control - in particular, the ability to increase the outer distance threshold without a negative impact on performance. If one AE prehends another in close $x - y$ proximity, the grey-level difference can be relatively high before it exceeds the bias threshold. However, if the distance between the two AEs is higher, only a small degree of variation can occur. The ability to weight greylevel and distance differences adds additional flexibility - though, as with penalties, bias measures complicate the process of parameter selection.

7.2 Scan Processing

7.2.1 Experiment 5: Rendering lateral ventricles

The Scan COPE framework was studied by attempting to form a 3-D rendering of the lateral ventricles, using five consecutive slices of an axial MRI brain scan. Three sets of parameters were tested; visual results for each will be presented in this section.

The first step in Scan COPE construction is to analyze individual slices and record segmentation results. Figure 7.7 illustrates the results obtained by applying SGC Slice COPE analysis to the five MRI slices. After the lateral ventricles were detected, the values in the converged Slice COPEs were used to initialize Real Slice Objects in a Scan COPE. It can be seen that segmentation of the ventricles was sufficiently accurate across all slices. Any Slice AE with weight $w > 10$ - in other words, any AE represented by a colored pixel on the right side of Fig. 7.7 - was assumed to correspond to ventricular tissue, and was therefore considered 'activated' when initializing the Scan COPE.

The real-world distance between consecutive slices was not known. However, the scan as a whole consisted of only 30 slices, suggesting that slice interpolation would be required to obtain a useful 3-D view of the ventricle structures. In constructing the Scan COPE, three Virtual Slice Objects were created in the gap between each RSO; the original slice images were each 256×256 pixels in size, meaning each VSO contained 65536 SCAEs. Unfortunately, this caused selection function involving entire slices to be costly in both memory usage and computation time. Analysis therefore focused on prehension functions involving individual SCAEs/DAEs. Memory limits also restricted analysis to a relatively low number of VSOs between each RSO; five was the maximum practical value. However,

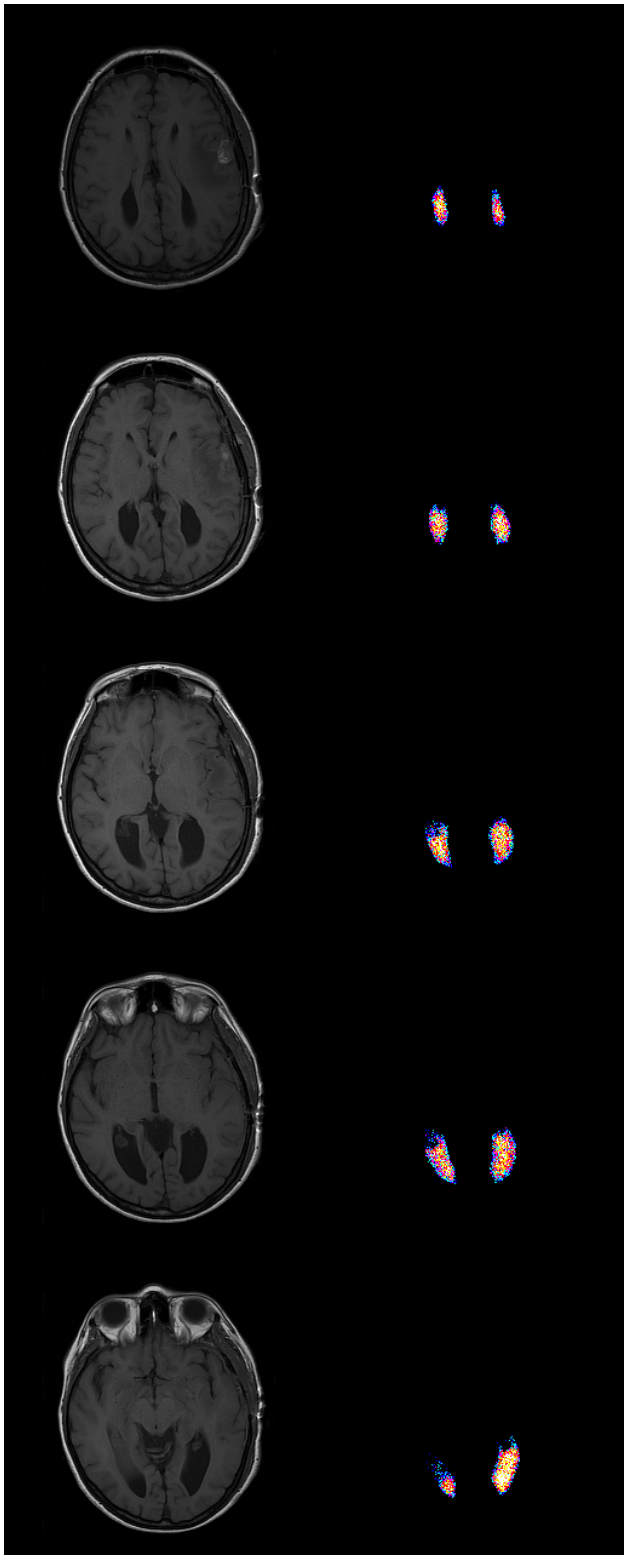


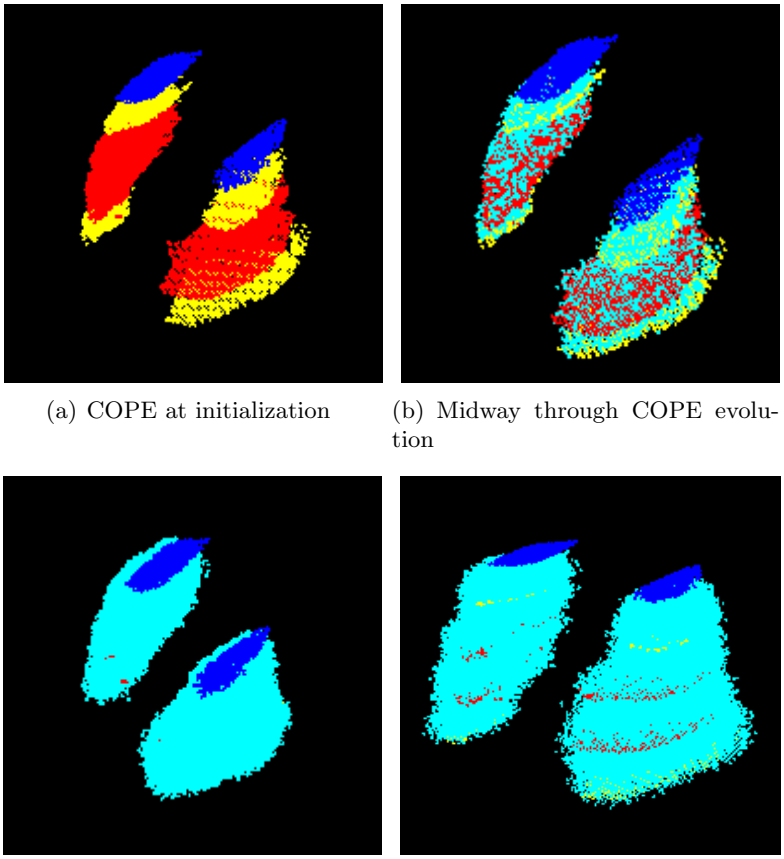
Figure 7.7: Ventricles extracted from an axial brain MRI

results at this stage provide a good overall view of Scan COPE behavior, and the potential utility in slice interpolation.

Figures 7.8, 7.9 and 7.10 illustrate the COPE's behavior using the three tested parameter sets. In the first experiment, shown in Fig. 7.8, five VSOs were created between each RSO. Note that two distance thresholds were used: intra-slice and inter-slice. Intra-slice distance involves only the $x - y$ plane - in other words, the prehending SCAE and the DAE are considered to exist in the same 2-D plane, even though they belong to different slices. Inter-slice distance is measured only along the z-axis. In this case, the intra-slice distance threshold was set to 10 units, and the inter-slice threshold to 3 units. Due to the weaker threshold, the COPE evolved relatively quickly, even though the number of VSOs was set at the practical maximum. Fig. 8(c) and Fig. 8(d) illustrate the COPE state after approximately 6.5 minutes. Unfortunately, the parameter choice negatively affected the rendering of the ventricular volume, as more prehensions occurred outside the region of interest. This caused VSOs to occupy a larger area of the $x - y$ plane and leading to 'fuzzy' edges.

The parameters were then altered to use a stricter $x - y$ distance threshold (6 units), and the number of VSOs between each RSO was reduced to four. The behavior of the resulting COPE is illustrated in Fig. 7.9. In this experiment, the ventricle structures appear smoother and the VSO areas more closely correspond to those of the closest RSOs. However, there is still some edge pixelation. COPE evolution was also slower; the results shown reflect its state after approximately eleven minutes.

The third experiment restricted the $x - y$ threshold to only 3 units, the strictest value



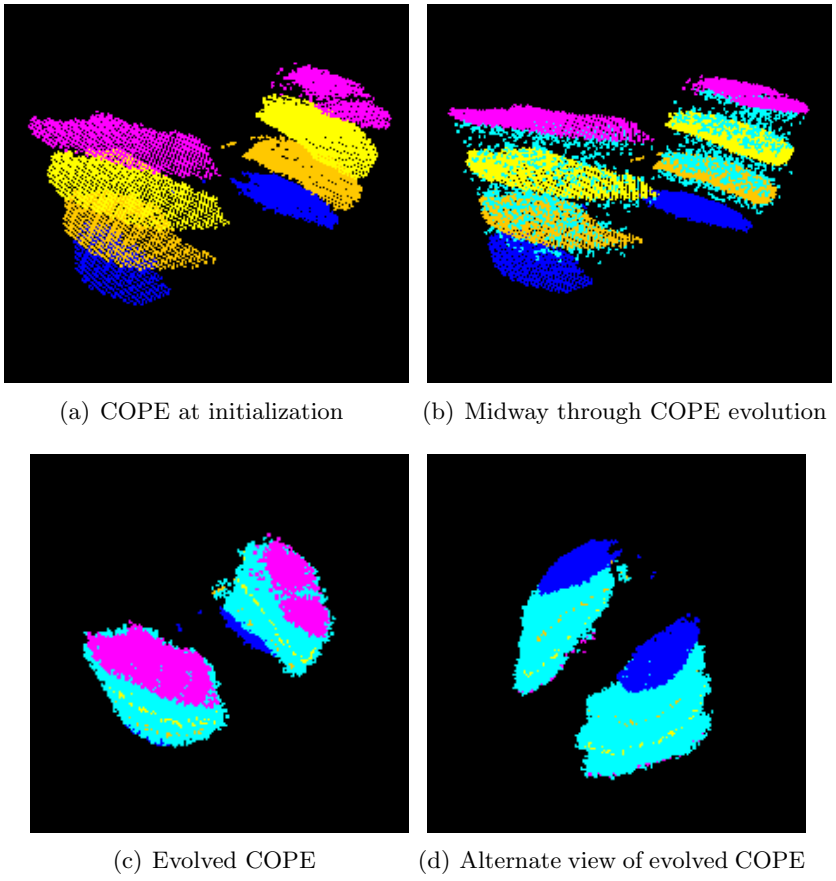
(a) COPE at initialization

(b) Midway through COPE evolution

(c) Evolved COPE

(d) Closer view of evolved COPE

Figure 7.8: Example evolution of a Scan COPE. Five VSOs were created between each RSO and the distance threshold was relatively weak (10 unit maximum). This caused the structure edges to appear 'fuzzy'.



(a) COPE at initialization (b) Midway through COPE evolution

(c) Evolved COPE (d) Alternate view of evolved COPE

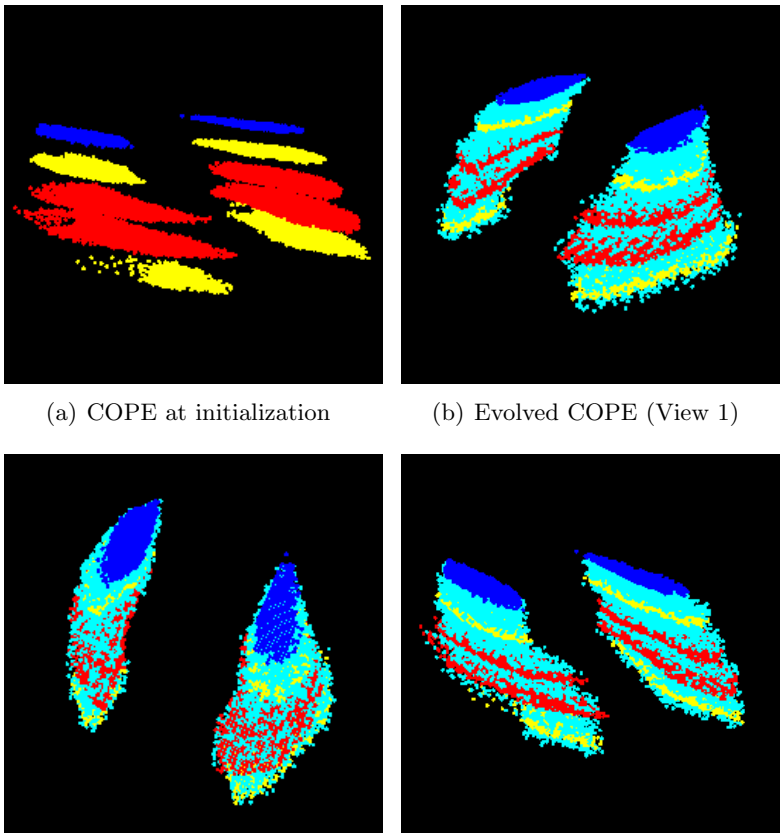
Figure 7.9: Example evolution of a Scan COPE. Four VSOs were created between each RSO and the distance threshold was slightly stricter (6 unit maximum). Edges are slightly sharper than in Fig. 7.8 but remain pixelated.

tested. The number of VSOs between each RSO was lowered to three. Figure 7.10 provides three views of the evolved COPE. It should be noted that evolution was manually halted after twenty minutes, prior to convergence, as the COPE was evolving very slowly and memory limitations were becoming an issue. The rendered volume therefore contains holes. Despite this, it can be seen that interpolation was progressing effectively, with the ventricular structures better capturing the variation in shape across the MRI slices. For example, Fig. 10(b) shows that the COPE has captured the slight abnormality in the leftmost structure (which corresponds to the left ventricle in Fig. 7.7). This suggests that in future tests, ideally in a distributed computing environment, heavy restrictions on the $x - y$ distance threshold are likely to produce smoother rendering results.

In the final experiment, the $x-y$ threshold was set to 4 units, and the number of VSOs to 6. Unlike the other experiments, SCAEs were allowed to prehend any DAE, instead of only the nearest. This increased the number of negative prehensions and therefore significantly increased convergence time; however, the resulting volume had smoother edges. Allowing prehension of any DAE reduces the 'step' effect that occurs when SCAEs only ever prehend DAEs in one specific slice (i.e. the nearest). The benefits are likely to be even more noticeable when using an higher number of VSOs. Results are presented in 7.11.

7.2.2 Impact of Prehension Parameters and Selection Functions

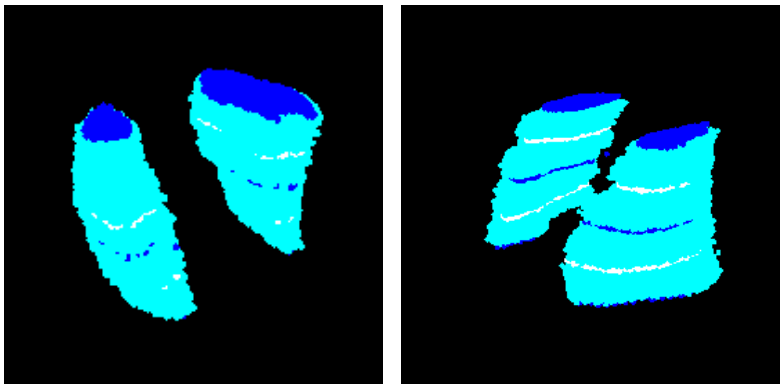
In the experiment above, incorporating the 3-D Euclidean distance between AEs into the prehension function did not significantly affect COPE behavior compared to the use of separate intra-slice and inter-slice distance thresholds. The $x - y$ distance threshold dominated the evolution of the COPE. This is intuitive given the high density of DAEs



(a) COPE at initialization (b) Evolved COPE (View 1)

(c) Evolved COPE (View 2) (d) Evolved COPE (View 3)

Figure 7.10: Example evolution of a Scan COPE. Three VSOs were created between each RSO and the distance threshold was reasonably strict (3 unit maximum). Even in close-up, the edges are sharper than in the previous two experiments using weaker distance thresholds. The difference between the two ventricular structures is more easily visible. However, evolution was very slow, and so the images were captured before convergence could complete.



(a) Evolved COPE (View 1) (b) Evolved COPE (View 2)

Figure 7.11: Example evolution of a Scan COPE. Six VSOs were created between each RSO and the distance threshold was reasonably strict (4 unit maximum). SCAEs were permitted to prehend any DAE in the COPE, leading to a smoother volume rendering.

in each RSO and the low ratio of VSOs to RSOs in each Scan COPE. The 3-D measure is expected to have more impact in situations where more than one structure is to be segmented and/or the Scan COPE contains many more VSOs between each RSO.

The same is true of the selection function allowing both SCAEs and DAEs to be prehended. Initially, it was expected that this would increase the speed of convergence - but in practice the number of VSOs between each slice was too low for such a weakening of prehension restrictions to have any obvious effect. However, it is likely to prove more useful in increasing the rate of convergence in Scan COPEs containing a higher number of VSOs, as it will increase the number of positive prehensions in those VSOs furthest from an RSO. Current results also suggest that smoother renderings can be obtained by allowing SCAEs to prehend any DAE, as opposed to only those in the nearest slice. Convergence time is increased due to the higher number of negative prehensions, but in an environment with sufficient processing power this issue is likely to be less significant.

CHAPTER 8 Conclusions and Future Work

8.1 Conclusions

This thesis describes an application of the COPE framework in medical image segmentation, focusing specifically on scans consisting of multiple cross sectional images. The developed system has shown potential in both detecting structures in 2-D images and rendering these structures as 3-D volumes across sequential slices of the scan. All objects used were derived from the COPE framework established by Primeaux, with various restrictions being used to implement the concept of a COPE 'goal' (in this case, segmenting a region or structure of interest). By incorporating dynamic 3-D visualization of COPE evolution, the system also offers a means to observe the emergent behavior resulting from the actions of individual AEs. One particular benefit of this visualization module is its portability; with a few alterations, it can be used in studying COPE behavior in other practical applications or in purely theoretical environments. The framework has also been designed for a high degree of flexibility, particularly in segmentation of 2-D images, and is not specific to application in the medical field. Based on preliminary results, the Seeded Greyscale COPE may have the most potential in practical segmentation applications involving structures which occupy a large percentage of the original image. However, the Distinguished Greyscale COPE currently offers better performance in segmenting smaller regions, at least in terms of computation speed. The use of emergent behavior in image segmentation tasks has been the subject of many other studies, and the COPE framework

could prove a valuable addition to the field, largely due to the flexibility it provides.

8.2 Future Work

The Slice/Scan COPE framework described in this thesis offers a vast range of potential avenues for future development, both practical and theoretical. This section will focus on select examples which are likely to assist in the proposed segmentation task and are natural extensions of the existing approach. First, the Slice and Scan COPEs could be extended to incorporate additional prehension and selection functions. These might be designed for specific image analysis goals, or to allow further study of the emergent behavior exhibited by the COPE. An important requirement of the second task would be to implement the dynamic 3D visualization in parallel, as this would allow a user to observe the evolution of the COPE in a manner closely approximating real time. Slice COPE visualization presently relies on updates after a set number of iterations, typically between 500 and 1000. Meanwhile, 3D visualization of Scan COPE evolution is too computationally intensive to be practical on a standard desktop computer. As stated in Section 5.1, implementing the COPE in a distributed computing environment would address this issue. Speed could also be increased by restricting the AEs which can be selected as prehenders - i.e., constructing a window enclosing the structure of interest in the nearest RSO, and only allowing VSO AEs within that window to prehend. However, this arguably contradicts the theory behind the COPE framework by placing severe constraints on the randomness of evolution. While the SGC slice COPE also has the capability to control which AEs act as prehenders, at the time of initialization any AE is either permitted to prehend or has the potential to become so in the future. In practical applications, the Scan COPE could prove useful in

estimating the volumes of the detected structures - such as identifying the size of a tumor to determine an appropriate radiation dosage for treatment. This would be relatively simple to implement, but the accuracy of the interpolation/rendering process must be verified first.

Bibliography

Bibliography

[1] G. Beni and J. Wang. Swarm intelligence in cellular robotic systems. In *Proceed. NATO Advanced Workshop on Robots and Biological Systems*, 1989.

[2] Leonardo Bocchi, Lucia Ballerini, and Signe Hssler. A new evolutionary algorithm for image segmentation. In *EvoWorkshops*, volume 3449 of *Lecture Notes in Computer Science*, pages 264–273. Springer, 2005.

[3] A.C. Rosa C. Fernandes, V. Ramos. Self-regulated artificial ant colonies on digital image habitats. *Int. Journal of Lateral Computing*, 2(1):1–8, 2005.

[4] S. Narayan C.E. White, G.A. Tagliarini. An algorithm for swarm-based color image segmentation. In *Proceed. 2004 IEEE SouthEast Conference*, pages 84–89, 2004.

[5] J.S. Duncan and N. Ayache. Medical image analysis: progress over two decades and the challenges ahead. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(1):85–106, 2000.

[6] B. B. Ertl-Wagner, J. D. Blume, D. Peck, J. K. Udupa, B. Herman, A. Levering, and I. M. Schmalfuss. Reliability of tumor volume estimation from MR images in patients with malignant glioma. Results from the American College of Radiology Imaging Network (ACRIN) 6662 Trial. *Eur Radiol*, 19:599–609, Mar 2009.

[7] G.C. Henry. *Forms of Concrescence: Alfred North Whitehead's Philosophy and Computer Programming Structures*. Bucknell University Press, 1993.

[8] J. Liu and Y.Y. Tang. Adaptive image segmentation with distributed behavior-based agents. *IEEE Trans. Pattern Anal. Mach. Intell.*, 21(6):544–551, 1999.

[9] S. Ouadfel, M. Batouche, and C. Garbay. Ant colony system for image segmentation using markov random field. In *Ant Algorithms*, volume 2463 of *Lecture Notes in Computer Science*, pages 294–295. Springer, 2002.

[10] N. Panteli and M. R. Dibben. Revisiting the nature of virtual organizations: reflections on mobile communication systems. *Futures*, 33(5):379 – 391, 2001.

[11] R.M. Pidaparti, D. Primeaux, and B. Saunders. Modeling and simulation of nanoscale self-assembly structures. In *Proceed. Sixth IEEE Conference on Nanotechnology (IEEE-NANO 2006.)*, 2006.

[12] D. Primeaux. Trust-based learning of data characteristics by an actual entity. In *Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, 2000. SNPD 2000. First ACIS International Conference on*, 2000.

[13] D. Primeaux. Experimentally determining a good threshold value for learning by an actual entity agent. In *Proceed. Second Int. Conf. on Computer and Information Science (ICIS 2006.)*, pages 55–58, 2002.

[14] D. Primeaux and B. Saunders. Finding global and local maxima in 3d space using swarm-like behavior in a colony of prehending entities. In *Proceed. 2006 IEEE Swarm Intelligence Symposium (IEEE-SIS 2006.)*, 2006.

[15] S. Saatchi and C. Hung. Using ant colony optimization and self-organizing map for image segmentation. In *MICAI*, pages 570–579, 2007.

[16] A.N. Whitehead. *Process and Reality: Corrected Edition*. Free Press, New York, 2 edition, 1979.

VITA

Rebecca Smith was born on July 21, 1982, in London, England, and holds United Kingdom citizenship. She received her Bachelor of Engineering in Computing from Imperial College London in 2003. Between January and December 2007 she was employed as a Graduate Teaching Assistant at Virginia Commonwealth University, and then as a Graduate Research Assistant between January 2008 and May 2010.